Analyzing Sales Performance by Region in a Retail Company

July 24, 2024

[6]:
```
pip install dask
```

```
Note: you may need to restart the kernel to use updated packages.
Collecting dask
  Downloading dask-2024.7.1-py3-none-any.whl.metadata (3.8 kB)
Requirement already satisfied: click>=8.1 in c:\users\mubashir
khan\appdata\local\programs\python\python312\lib\site-packages (from dask)
(8.1.7)
Collecting cloudpickle>=1.5.0 (from dask)
  Downloading cloudpickle-3.0.0-py3-none-any.whl.metadata (7.0 kB)
Requirement already satisfied: fsspec>=2021.09.0 in c:\users\mubashir
khan\appdata\local\programs\python\python312\lib\site-packages (from dask)
(2024.2.0)
Requirement already satisfied: packaging>=20.0 in c:\users\mubashir
khan\appdata\local\programs\python\python312\lib\site-packages (from dask)
(23.2)
Collecting partd>=1.4.0 (from dask)
  Downloading partd-1.4.2-py3-none-any.whl.metadata (4.6 kB)
Requirement already satisfied: pyyaml>=5.3.1 in c:\users\mubashir
khan\appdata\local\programs\python\python312\lib\site-packages (from dask)
(6.0.1)
Requirement already satisfied: toolz>=0.10.0 in c:\users\mubashir
khan\appdata\local\programs\python\python312\lib\site-packages (from dask)
(0.12.1)
Requirement already satisfied: colorama in c:\users\mubashir
khan\appdata\local\programs\python\python312\lib\site-packages (from
click>=8.1->dask) (0.4.6)
Collecting locket (from partd>=1.4.0->dask)
  Downloading locket-1.0.0-py2.py3-none-any.whl.metadata (2.8 kB)
Downloading dask-2024.7.1-py3-none-any.whl (1.2 MB)
   ---------------------------------------- 0.0/1.2 MB ? eta -:--:--
   ---------------- ----------------------- 0.5/1.2 MB 10.7 MB/s eta 0:00:01
   --------------------------------- ------ 1.0/1.2 MB 10.9 MB/s eta 0:00:01
   ---------------------------------------- 1.2/1.2 MB 8.7 MB/s eta 0:00:00
Downloading cloudpickle-3.0.0-py3-none-any.whl (20 kB)
Downloading partd-1.4.2-py3-none-any.whl (18 kB)
Downloading locket-1.0.0-py2.py3-none-any.whl (4.4 kB)
Installing collected packages: locket, cloudpickle, partd, dask
```

```
Successfully installed cloudpickle-3.0.0 dask-2024.7.1 locket-1.0.0 partd-1.4.2
```

```
DEPRECATION: Loading egg at c:\users\mubashir
khan\appdata\local\programs\python\python312\lib\site-
packages\spylon_kernel-0+unknown-py3.12.egg is deprecated. pip 24.3 will enforce
this behaviour change. A possible replacement is to use pip for package
installation. Discussion can be found at
https://github.com/pypa/pip/issues/12330

[notice] A new release of pip is available: 24.1.1 -> 24.1.2
[notice] To update, run: python.exe -m pip install --upgrade pip
```

[9]:

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

# File paths
sales_file_path = r'C:\Users\MUBASHIR KHAN\Desktop\jupyter\DMV\sales.csv'
product_file_path = r'C:\Users\MUBASHIR␣
  ↪KHAN\Desktop\jupyter\DMV\product_hierarchy.csv'
store_file_path = r'C:\Users\MUBASHIR KHAN\Desktop\jupyter\DMV\store_cities.csv'

# Load datasets with proper column types
dtype_dict_sales = {'Product ID': 'str', 'Store ID': 'str', 'Sales Amount':␣
  ↪'float64'}
dtype_dict_product = {'Product ID': 'str'}
dtype_dict_store = {'Store ID': 'str'}

def load_data(file_path, dtype_dict=None):
    try:
        return pd.read_csv(file_path, dtype=dtype_dict, low_memory=False)
    except MemoryError:
        print("MemoryError: Unable to load the file.")
        return None

# Load datasets
sales_df = load_data(sales_file_path, dtype_dict_sales)
product_df = load_data(product_file_path, dtype_dict_product)
store_df = load_data(store_file_path, dtype_dict_store)

# Print the first few rows and column names
print("Sales DataFrame columns:", sales_df.columns)
print("Product DataFrame columns:", product_df.columns)
print("Store DataFrame columns:", store_df.columns)

# Print the first few rows of each DataFrame
print(sales_df.head())
print(product_df.head())
```

```python
print(store_df.head())

# Print dataset info
print(sales_df.info())
print(product_df.info())
print(store_df.info())

# Check if the required columns are present
required_columns_sales = {'Product ID', 'Store ID', 'Sales Amount'}
required_columns_product = {'Product ID'}
required_columns_store = {'Store ID'}

print("Sales DataFrame columns missing:", required_columns_sales - set(sales_df.columns))
print("Product DataFrame columns missing:", required_columns_product - set(product_df.columns))
print("Store DataFrame columns missing:", required_columns_store - set(store_df.columns))

# Adjust column names if necessary (example)
# sales_df.rename(columns={'Product ID ': 'Product ID'}, inplace=True)  # Adjust if needed

# Merge datasets
try:
    sales_product_df = pd.merge(sales_df, product_df, on='Product ID', how='left')
    sales_product_store_df = pd.merge(sales_product_df, store_df, on='Store ID', how='left')

    # Check the merged dataset
    print(sales_product_store_df.head())
    print(sales_product_store_df.info())

    # Group by region and calculate total sales amount
    sales_by_region = sales_product_store_df.groupby('Region')['Sales Amount'].sum().reset_index()
    sales_by_region = sales_by_region.sort_values(by='Sales Amount', ascending=False)

    # Bar plot for sales distribution by region
    plt.figure(figsize=(10, 6))
    sns.barplot(x='Region', y='Sales Amount', data=sales_by_region, palette='viridis')
    plt.title('Total Sales Amount by Region')
    plt.xlabel('Region')
    plt.ylabel('Total Sales Amount')
    plt.xticks(rotation=45)
    plt.show()

    # Pie chart for sales distribution by region
    plt.figure(figsize=(8, 8))
    plt.pie(sales_by_region['Sales Amount'], labels=sales_by_region['Region'],␣
 ↪autopct='%1.1f%%', colors=sns.color_palette('viridis', len(sales_by_region)))
    plt.title('Sales Distribution by Region')
    plt.show()

    # Identify top-performing regions
    top_regions = sales_by_region.head(5)
    print("Top Performing Regions:")
    print(top_regions)

    # Group by region and product category
    sales_by_region_category = sales_product_store_df.groupby(['Region',␣
 ↪'Product Category'])['Sales Amount'].sum().reset_index()

    # Pivot the data for better visualization
    sales_pivot = sales_by_region_category.pivot(index='Region',␣
 ↪columns='Product Category', values='Sales Amount').fillna(0)
    print(sales_pivot)

    # Stacked bar plot
    sales_pivot.plot(kind='bar', stacked=True, figsize=(12, 8),␣
 ↪colormap='viridis')
    plt.title('Sales Amount by Region and Product Category (Stacked)')
    plt.xlabel('Region')
    plt.ylabel('Total Sales Amount')
    plt.xticks(rotation=45)
    plt.legend(title='Product Category')
    plt.show()

    # Grouped bar plot
    sales_pivot.plot(kind='bar', figsize=(12, 8), colormap='viridis')
    plt.title('Sales Amount by Region and Product Category (Grouped)')
    plt.xlabel('Region')
    plt.ylabel('Total Sales Amount')
    plt.xticks(rotation=45)
    plt.legend(title='Product Category')
    plt.show()

except KeyError as e:
    print(f"KeyError: {e}")
```

```
Sales DataFrame columns: Index(['product_id', 'store_id', 'date', 'sales',
'revenue', 'stock', 'price',
       'promo_type_1', 'promo_bin_1', 'promo_type_2', 'promo_bin_2',
       'promo_discount_2', 'promo_discount_type_2'],
      dtype='object')
Product DataFrame columns: Index(['product_id', 'product_length',
'product_depth', 'product_width',
       'cluster_id', 'hierarchy1_id', 'hierarchy2_id', 'hierarchy3_id',
       'hierarchy4_id', 'hierarchy5_id'],
      dtype='object')
Store DataFrame columns: Index(['store_id', 'storetype_id', 'store_size',
'city_id'], dtype='object')
  product_id store_id        date  sales  revenue  stock  price promo_type_1  \
0      P0001    S0002  2017-01-02    0.0     0.00    8.0   6.25         PR14
1      P0001    S0012  2017-01-02    1.0     5.30    0.0   6.25         PR14
2      P0001    S0013  2017-01-02    2.0    10.59    0.0   6.25         PR14
3      P0001    S0023  2017-01-02    0.0     0.00    6.0   6.25         PR14
4      P0001    S0025  2017-01-02    0.0     0.00    1.0   6.25         PR14

  promo_bin_1 promo_type_2 promo_bin_2  promo_discount_2 promo_discount_type_2
0         NaN         PR03         NaN               NaN                   NaN
1         NaN         PR03         NaN               NaN                   NaN
2         NaN         PR03         NaN               NaN                   NaN
3         NaN         PR03         NaN               NaN                   NaN
4         NaN         PR03         NaN               NaN                   NaN
  product_id  product_length  product_depth  product_width cluster_id  \
0      P0000             5.0           20.0           12.0        NaN
1      P0001            13.5           22.0           20.0  cluster_5
2      P0002            22.0           40.0           22.0  cluster_0
3      P0004             2.0           13.0            4.0  cluster_3
4      P0005            16.0           30.0           16.0  cluster_9

  hierarchy1_id hierarchy2_id hierarchy3_id hierarchy4_id hierarchy5_id
0           H00         H0004       H000401     H00040105   H0004010534
1           H01         H0105       H010501     H01050100   H0105010006
2           H03         H0315       H031508     H03150800   H0315080028
3           H03         H0314       H031405     H03140500   H0314050003
4           H03         H0312       H031211     H03121109   H0312110917
  store_id storetype_id  store_size city_id
0    S0091         ST04          19    C013
1    S0012         ST04          28    C005
2    S0045         ST04          17    C008
3    S0032         ST03          14    C019
4    S0027         ST04          24    C022
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19454838 entries, 0 to 19454837
Data columns (total 13 columns):
 #   Column                 Dtype
```

```
---  ------                 -----
 0   product_id             object
 1   store_id               object
 2   date                   object
 3   sales                  float64
 4   revenue                float64
 5   stock                  float64
 6   price                  float64
 7   promo_type_1           object
 8   promo_bin_1            object
 9   promo_type_2           object
 10  promo_bin_2            object
 11  promo_discount_2       float64
 12  promo_discount_type_2  object
dtypes: float64(5), object(8)
memory usage: 1.9+ GB
None
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 699 entries, 0 to 698
Data columns (total 10 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   product_id      699 non-null    object
 1   product_length  681 non-null    float64
 2   product_depth   683 non-null    float64
 3   product_width   683 non-null    float64
 4   cluster_id      649 non-null    object
 5   hierarchy1_id   699 non-null    object
 6   hierarchy2_id   699 non-null    object
 7   hierarchy3_id   699 non-null    object
 8   hierarchy4_id   699 non-null    object
 9   hierarchy5_id   699 non-null    object
dtypes: float64(3), object(7)
memory usage: 54.7+ KB
None
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 144 entries, 0 to 143
Data columns (total 4 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   store_id      144 non-null    object
 1   storetype_id  144 non-null    object
 2   store_size    144 non-null    int64
 3   city_id       144 non-null    object
dtypes: int64(1), object(3)
memory usage: 4.6+ KB
None
Sales DataFrame columns missing: {'Product ID', 'Sales Amount', 'Store ID'}
```

```
Product DataFrame columns missing: {'Product ID'}
Store DataFrame columns missing: {'Store ID'}
KeyError: 'Product ID'
```

[ ]: