In [146…

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
import seaborn as sns
```

In [104…

```
#loading the dataset
df = pd.read_csv("C:\TE sem 7\data modeling and visualization\sales_data_sampl
df
```

```
<>:2: SyntaxWarning: invalid escape sequence '\T'
<>:2: SyntaxWarning: invalid escape sequence '\T'
C:\Users\vidhi\AppData\Local\Temp\ipykernel_26944\3890246303.py:2: SyntaxWarnin
g: invalid escape sequence '\T'
  df = pd.read_csv("C:\TE sem 7\data modeling and visualization\sales_data_samp
le.csv", encoding='latin1')
```

Out[104…

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SA
0	10107	30	95.70	2	287
1	10121	34	81.35	5	276
2	10134	41	94.74	2	388
3	10145	45	83.26	6	374
4	10159	49	100.00	14	520
...	...	...	...	...	
2818	10350	20	100.00	15	224
2819	10373	29	100.00	1	397
2820	10386	43	100.00	4	541
2821	10397	34	62.24	1	211
2822	10414	47	65.52	9	307

2823 rows × 25 columns

In [106…

```
j_file = pd.read_json("C:\TE sem 7\data modeling and visualization\customers.j
j_file
```

```
<>:1: SyntaxWarning: invalid escape sequence '\T'
<>:1: SyntaxWarning: invalid escape sequence '\T'
C:\Users\vidhi\AppData\Local\Temp\ipykernel_26944\4144724392.py:1: SyntaxWarnin
g: invalid escape sequence '\T'
  j_file = pd.read_json("C:\TE sem 7\data modeling and visualization\customer
s.json")
```

Out[106…

	id	email	first	last	company	
0	1	isidro_von@hotmail.com	Torrey	Veum	Hilll, Mayert and Wolf	04:06:
1	2	frederique19@gmail.com	Micah	Sanford	Stokes-Reichel	16:08:
2	3	fredy54@gmail.com	Hollis	Swift	Rodriguez, Cartwright and Kuhn	06:15:
3	4	braxton29@hotmail.com	Perry	Leffler	Sipes, Feeney and Hansen	11:31:
4	5	turner59@gmail.com	Janelle	Hagenes	Lesch and Daughters	15:05:
...	...	...	...	...	...	
9994	9995	delores_cruickshank@gmail.com	Robert	Batz	Carter-Tillman	19:13:
9995	9996	marley_brown32@hotmail.com	Leone	Reinger	Smitham and Daughters	18:45:
9996	9997	raymond68@hotmail.com	Clementina	Bode	VonRueden LLC	18:38:
9997	9998	juston_powlowski@hotmail.com	Yvonne	Prosacco	Green Inc	18:54:
9998	9999	orion.senger72@yahoo.com	Darrin	Connelly	Funk and Daughters	11:20:

9999 rows × 7 columns

In [108…

```
xl_data = pd.read_excel(r"C:\TE sem 7\data modeling and visualization\Sample-S
xl_data
```

Out[108…

	Postcode	Sales_Rep_ID	Sales_Rep_Name	Year	Value
0	2121	456	Jane	2011	84219.497311
1	2092	789	Ashish	2012	28322.192268
2	2128	456	Jane	2013	81878.997241
3	2073	123	John	2011	44491.142121
4	2134	789	Ashish	2012	71837.720959
...	...	...	...	...	...
385	2164	123	John	2012	88884.535217
386	2193	456	Jane	2013	79440.290813
387	2031	123	John	2011	65643.689454
388	2130	456	Jane	2012	66247.874869
389	2116	456	Jane	2013	3195.699054

390 rows × 5 columns

In [110…

```
# Finding missing data
df.isnull().sum()
```

Out[110…

```
ORDERNUMBER            0
QUANTITYORDERED        0
PRICEEACH              0
ORDERLINENUMBER        0
SALES                  0
ORDERDATE              0
STATUS                 0
QTR_ID                 0
MONTH_ID               0
YEAR_ID                0
PRODUCTLINE            0
MSRP                   0
PRODUCTCODE            0
CUSTOMERNAME           0
PHONE                  0
ADDRESSLINE1           0
ADDRESSLINE2        2521
CITY                   0
STATE               1486
POSTALCODE            76
COUNTRY                0
TERRITORY           1074
CONTACTLASTNAME        0
CONTACTFIRSTNAME       0
DEALSIZE               0
dtype: int64
```

In [112…

```
df.dropna()
```

Out[112…

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	S
10	10223	37	100.00	1	39
21	10361	20	72.55	13	14
40	10270	21	100.00	9	49
47	10347	30	100.00	1	39
51	10391	24	100.00	4	24
...	...	...	...	...	
2667	10120	43	76.00	14	32
2673	10223	26	67.20	15	17
2685	10361	44	100.00	10	50
2764	10361	35	100.00	11	42
2791	10361	23	95.20	12	21

147 rows × 25 columns

In [114…

```
df.drop_duplicates()
```

Out[114…

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SA
0	10107	30	95.70	2	28
1	10121	34	81.35	5	276
2	10134	41	94.74	2	388
3	10145	45	83.26	6	374
4	10159	49	100.00	14	520
...	...	...	...	...	
2818	10350	20	100.00	15	224
2819	10373	29	100.00	1	39
2820	10386	43	100.00	4	54
2821	10397	34	62.24	1	21
2822	10414	47	65.52	9	30

2823 rows × 25 columns

In [116…

```
# Finding duplicates
df.duplicated().sum()
```

Out[116…

```
0
```

In [118…

```
j_file.isnull().sum()
```

Out[118…

```
id            0
email         0
first         0
last          0
company       0
created_at    0
country       0
dtype: int64
```

In [120…

```
xl_data.isnull().sum()
```

Out[120…

```
Postcode          0
Sales_Rep_ID      0
Sales_Rep_Name    0
Year              0
Value             0
dtype: int64
```

In [122…

```
j_file.duplicated().sum()
```

Out[122…

```
0
```

In [124…

```
xl_data.duplicated().sum()
```

Out[124…

```
0
```

In [126…

```
# Concat all three files
concat_df = pd.concat([j_file,df,xl_data], ignore_index=True)
concat_df
```

Out[126…

	id	email	first	last	company	creat
0	1.0	isidro_von@hotmail.com	Torrey	Veum	Hilll, Mayert and Wolf	2014- 04:06:27.981000+
1	2.0	frederique19@gmail.com	Micah	Sanford	Stokes-Reichel	2014- 16:08:17.044000+
2	3.0	fredy54@gmail.com	Hollis	Swift	Rodriguez, Cartwright and Kuhn	2014- 06:15:16.731000+
3	4.0	braxton29@hotmail.com	Perry	Leffler	Sipes, Feeney and Hansen	2014- 11:31:40.235000+
4	5.0	turner59@gmail.com	Janelle	Hagenes	Lesch and Daughters	2014- 15:05:43.229000+
...	...	...	...	...	...	
13207	NaN	NaN	NaN	NaN	NaN	
13208	NaN	NaN	NaN	NaN	NaN	
13209	NaN	NaN	NaN	NaN	NaN	
13210	NaN	NaN	NaN	NaN	NaN	
13211	NaN	NaN	NaN	NaN	NaN	

13212 rows × 37 columns

In [128…

```
concat_df.describe()
```

Out[128…

	id	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLIN
count	9999.000000	2823.000000	2823.000000	2823.000000	2
mean	5000.000000	10258.725115	35.092809	83.658544	
std	2886.607005	92.085478	9.741443	20.174277	
min	1.000000	10100.000000	6.000000	26.880000	
25%	2500.500000	10180.000000	27.000000	68.860000	
50%	5000.000000	10262.000000	35.000000	95.700000	
75%	7499.500000	10333.500000	43.000000	100.000000	
max	9999.000000	10425.000000	97.000000	100.000000	

In [130…

```
concat_df.columns
```

Out[130…

```
Index(['id', 'email', 'first', 'last', 'company', 'created_at', 'country',
       'ORDERNUMBER', 'QUANTITYORDERED', 'PRICEEACH', 'ORDERLINENUMBER',
       'SALES', 'ORDERDATE', 'STATUS', 'QTR_ID', 'MONTH_ID', 'YEAR_ID',
       'PRODUCTLINE', 'MSRP', 'PRODUCTCODE', 'CUSTOMERNAME', 'PHONE',
       'ADDRESSLINE1', 'ADDRESSLINE2', 'CITY', 'STATE', 'POSTALCODE',
       'COUNTRY', 'TERRITORY', 'CONTACTLASTNAME', 'CONTACTFIRSTNAME',
       'DEALSIZE', 'Postcode', 'Sales_Rep_ID', 'Sales_Rep_Name', 'Year',
       'Value'],
      dtype='object')
```

In [132…

```
total_sales = concat_df['SALES'].sum()
print("Total Sales:", total_sales)
```

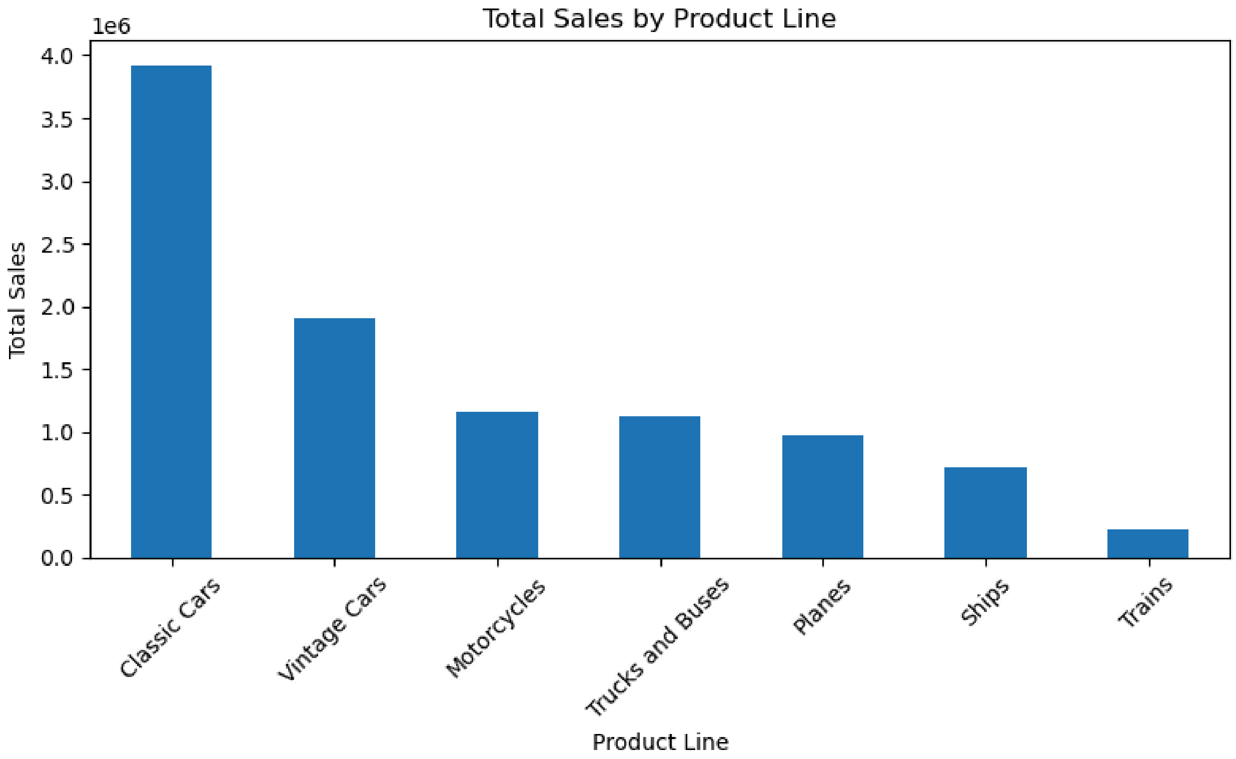
```
Total Sales: 10032628.850000001
```

In [134…

```
category_sales = concat_df.groupby('ORDERNUMBER')['SALES'].mean()
```

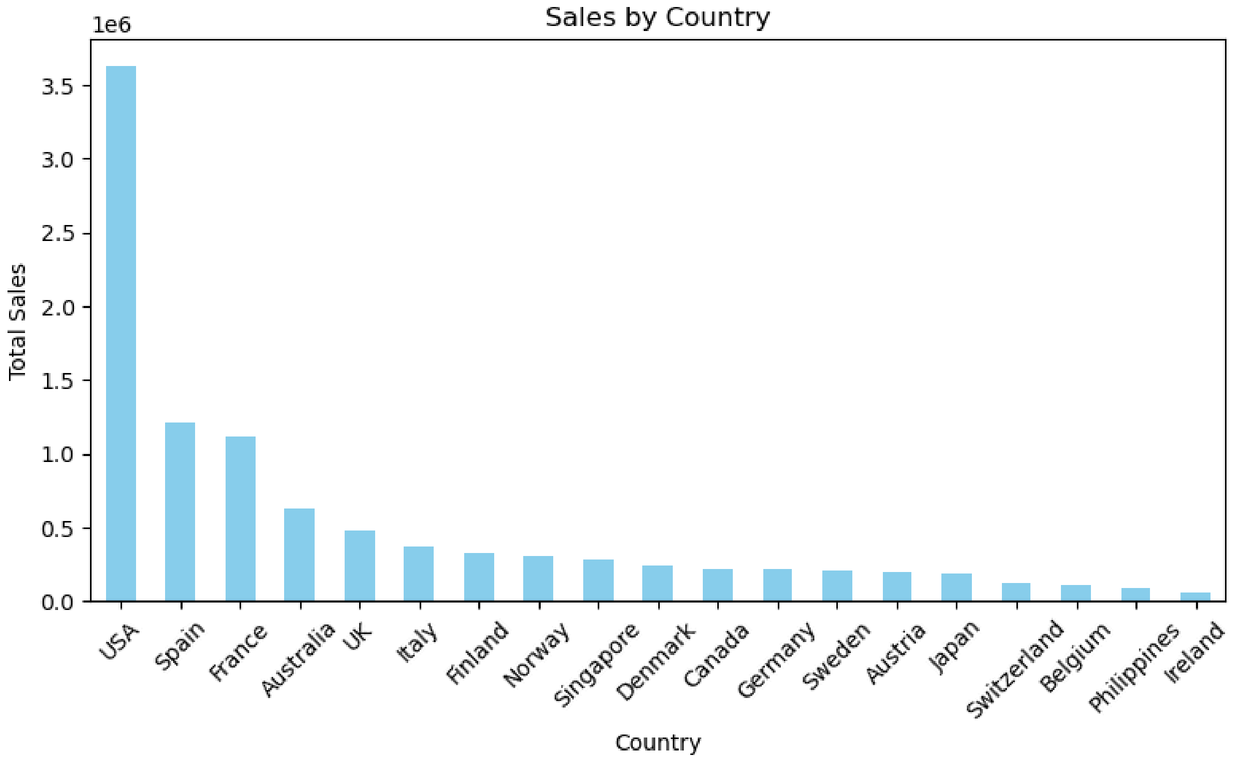
In [136…

```
#This shows which product category contributes most to revenue.
sales_by_product = concat_df.groupby('PRODUCTLINE')['SALES'].sum().sort_values

plt.figure(figsize=(8,5))
sales_by_product.plot(kind='bar')
plt.title('Total Sales by Product Line')
plt.xlabel('Product Line')
plt.ylabel('Total Sales')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



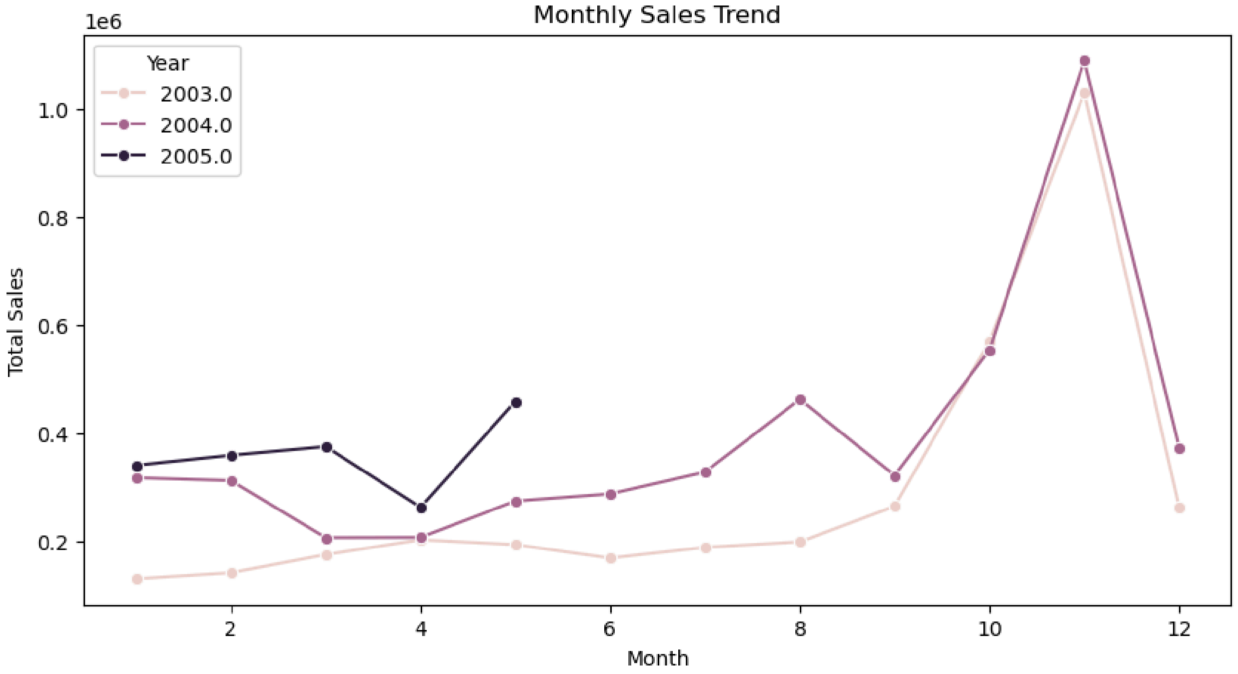
In [140…

```
sales_by_country = concat_df.groupby('COUNTRY')['SALES'].sum().sort_values(asc

plt.figure(figsize=(8,5))
sales_by_country.plot(kind='bar', color='skyblue')
plt.title('Sales by Country')
plt.xlabel('Country')
plt.ylabel('Total Sales')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



In [148…

```
monthly_sales = concat_df.groupby(['YEAR_ID', 'MONTH_ID'])['SALES'].sum().rese

plt.figure(figsize=(10,5))
sns.lineplot(data=monthly_sales, x='MONTH_ID', y='SALES', hue='YEAR_ID', marke
plt.title('Monthly Sales Trend')
plt.xlabel('Month')
plt.ylabel('Total Sales')
plt.legend(title='Year')
plt.show()
```



In [ ]: