exp_3_vis_titanic

February 17, 2025

1 Experiment 3: Data Visualization with Seaborn and Matplotlib

- Name: **Anas Muhammmed Sahil**
- Date: 23-01-2025
- Roll Number: 20242AIE0010

[1]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

2 Titanic dataset with Pandas

[2]:
```python
titanic = pd.read_csv('data/titanic.csv')
```

[3]:
```python
titanic.shape
```

[3]:
```
(891, 12)
```

[4]:
```python
titanic.head()
```

[4]:
```
   PassengerId  Survived  Pclass  \
0            1         0       3
1            2         1       1
2            3         1       3
3            4         1       1
4            5         0       3

                                                Name     Sex   Age  SibSp  \
0                            Braund, Mr. Owen Harris    male  22.0      1
1  Cumings, Mrs. John Bradley (Florence Briggs Th…  female  38.0      1
2                             Heikkinen, Miss. Laina  female  26.0      0
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
4                           Allen, Mr. William Henry    male  35.0      0

   Parch            Ticket     Fare Cabin Embarked
```

```
0      0          A/5 21171    7.2500    NaN        S
1      0           PC 17599   71.2833    C85        C
2      0   STON/O2. 3101282    7.9250    NaN        S
3      0             113803   53.1000   C123        S
4      0             373450    8.0500    NaN        S
```

[5]:
```
list(titanic)
```

[5]:
```
['PassengerId',
 'Survived',
 'Pclass',
 'Name',
 'Sex',
 'Age',
 'SibSp',
 'Parch',
 'Ticket',
 'Fare',
 'Cabin',
 'Embarked']
```

[6]:
```
titanic.dtypes
```

[6]:
```
PassengerId      int64
Survived         int64
Pclass           int64
Name            object
Sex             object
Age            float64
SibSp            int64
Parch            int64
Ticket          object
Fare           float64
Cabin           object
Embarked        object
dtype: object
```

2.1 Objective of this Study

This dataset has the following categorical features: * Survived: 1 = Yes, 0= No * Pclass (Passenger Class): 1,2,3 * Sex: Male, Female * Embarked (Port of Embarkation): C = Cherbourg, Q = Queenstown, S = Southampton

The objective is to undertand the relationship between the the above variables. The following questions will be answered:

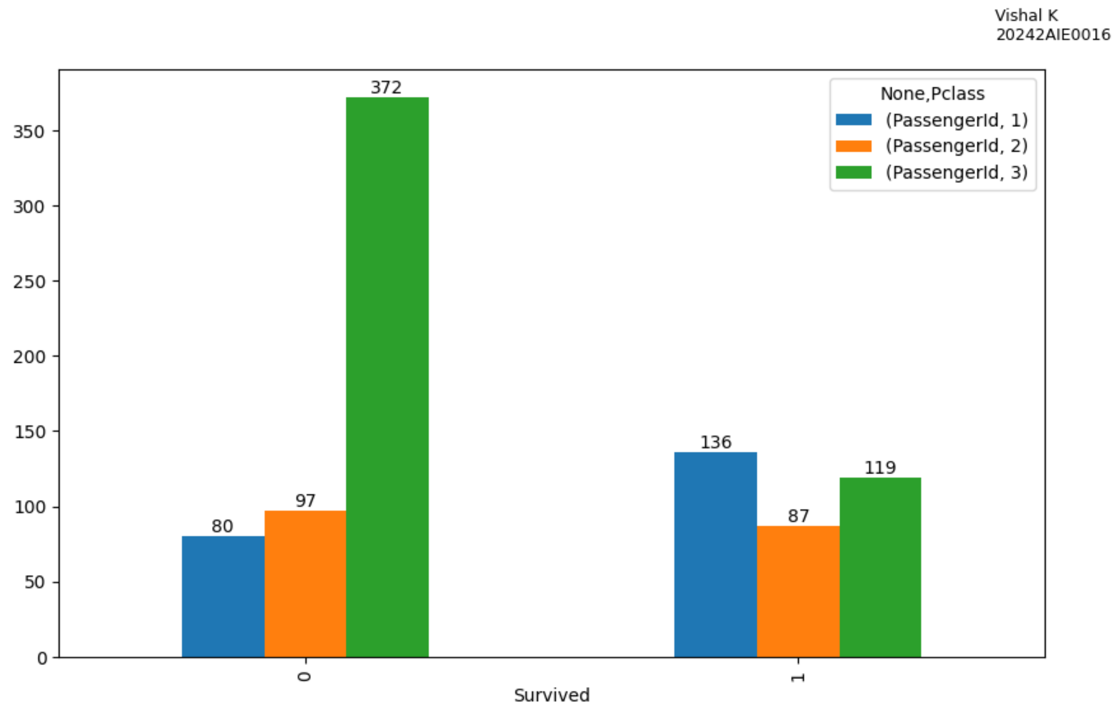
[7]:
```
def make_pivot(param1, param2, name):
    df_slice = titanic[[param1, param2, 'PassengerId']]
```

```python
    slice_pivot = df_slice.pivot_table(index=[param1], columns=[param2],␣
↪aggfunc=np.size, fill_value=0)

    p_chart = slice_pivot.plot.bar(figsize=(10, 6))

    # Annotate bars with values
    for p in p_chart.patches:
        p_chart.annotate(
            str(p.get_height()),
            (p.get_x() + p.get_width() / 2, p.get_height()),
            ha='center', va='bottom', fontsize=10, color='black'
        )

    # Add name label at the top right corner
    max_height = max([p.get_height() for p in p_chart.patches])
    p_chart.text(
        0.95, 1.05, name,
        transform=p_chart.transAxes,
        fontsize=9, color="black",
    )

    return slice_pivot, p_chart
```

2.2 1) Relation between passengers' survival and booking class

[8]:
```python
make_pivot ('Survived','Pclass', "Vishal K\n20242AIE0016")
```

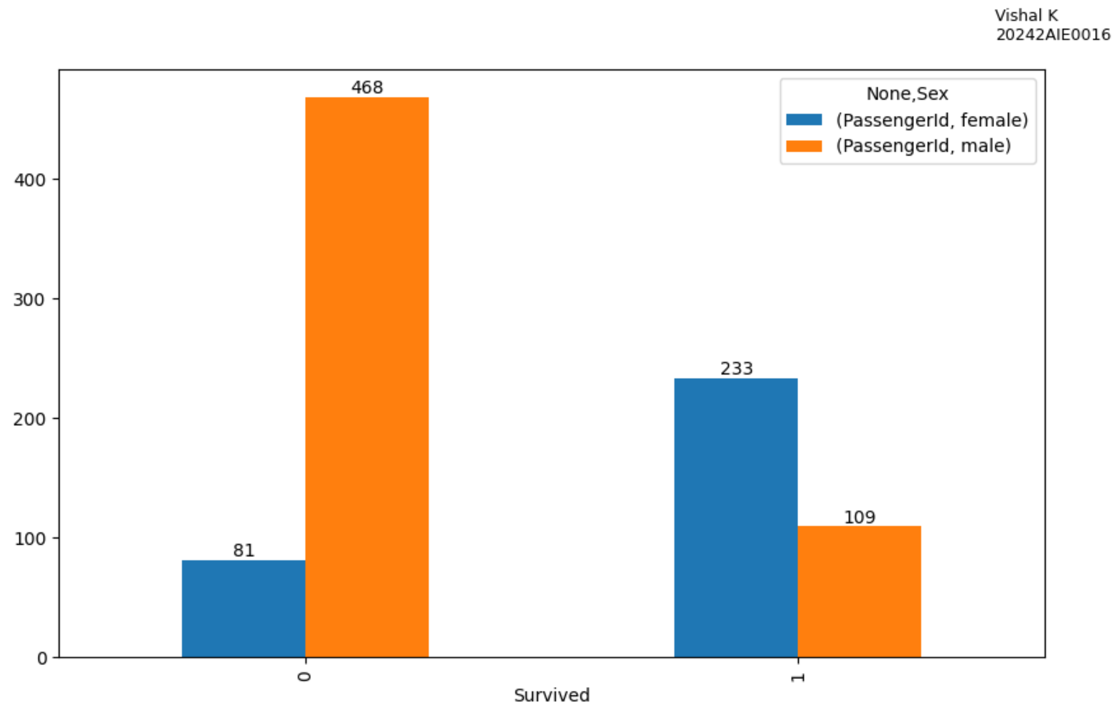
[8]:
```
(         PassengerId
 Pclass             1   2    3
 Survived
 0                 80  97  372
 1                136  87  119,
 <Axes: xlabel='Survived'>)
```



2.3 2) Relation between passengers' survival and their sex

[9]:
```
make_pivot ('Survived','Sex', "Vishal K\n20242AIE0016")
```

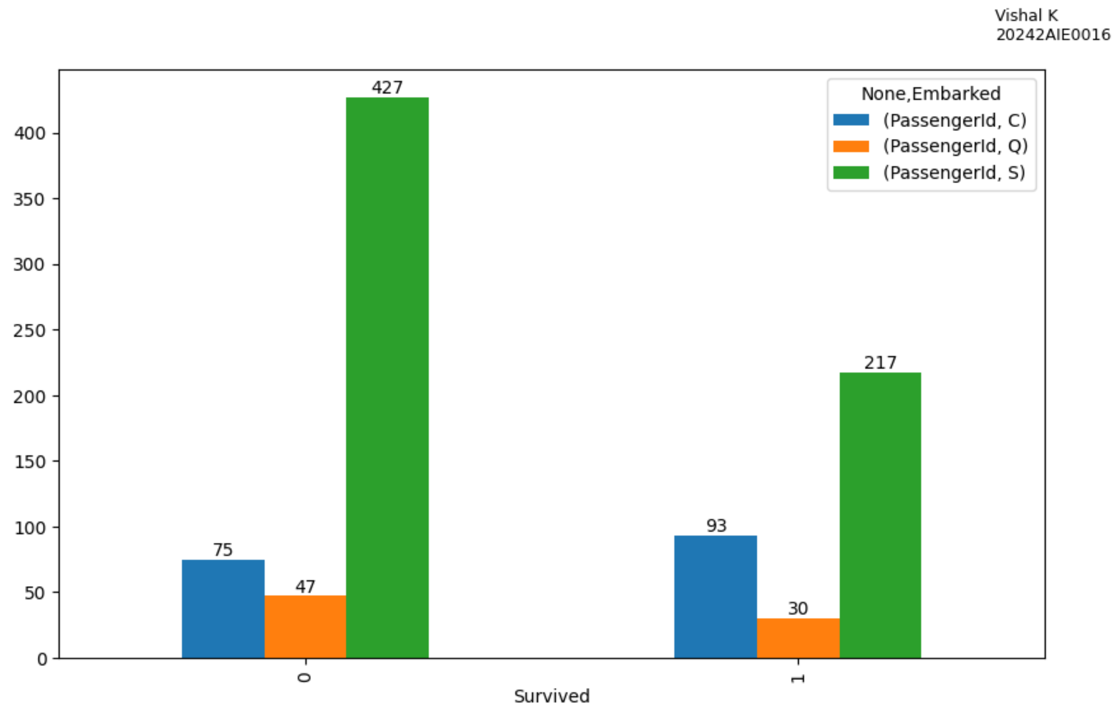
[9]:
```
(         PassengerId
 Sex            female male
 Survived
 0                  81  468
 1                 233  109,
 <Axes: xlabel='Survived'>)
```



2.4 3) Relation between passengers' survival and port of embarkation

[10]:
```
make_pivot ('Survived','Embarked', "Vishal K\n20242AIE0016")
```

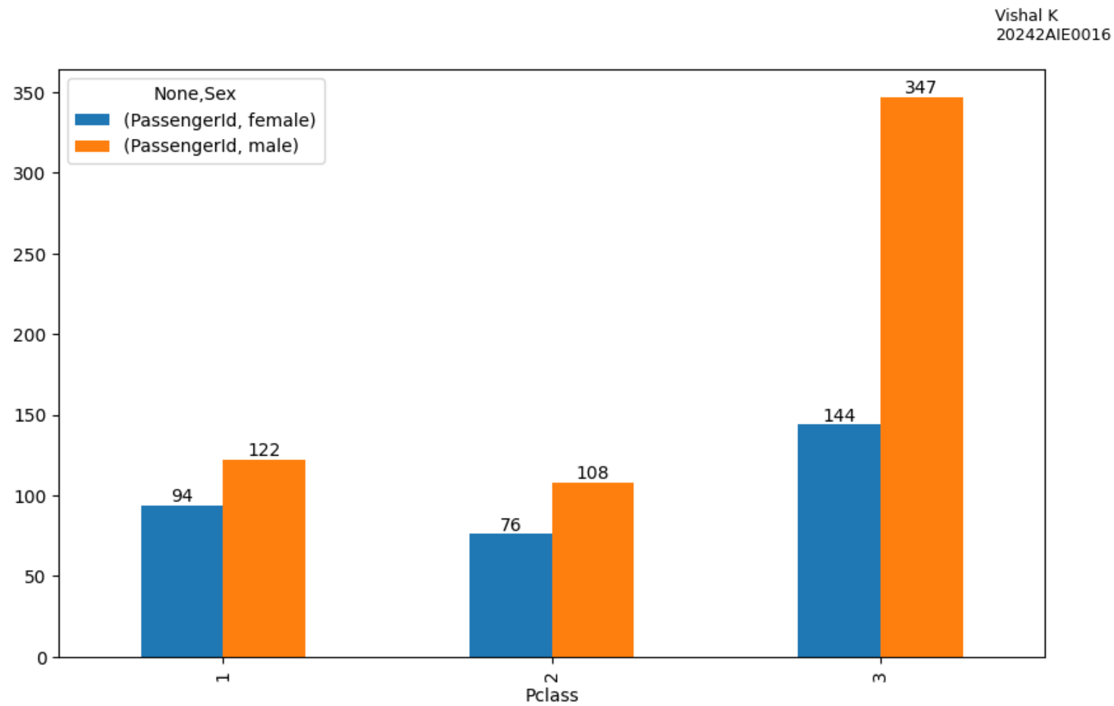
[10]:
```
(          PassengerId
 Embarked            C   Q    S
 Survived
 0                  75  47  427
 1                  93  30  217,
 <Axes: xlabel='Survived'>)
```



2.5 4) Relation between passengers' booking class and their sex

[11]:
```
make_pivot ('Pclass','Sex', "Vishal K\n20242AIE0016")
```

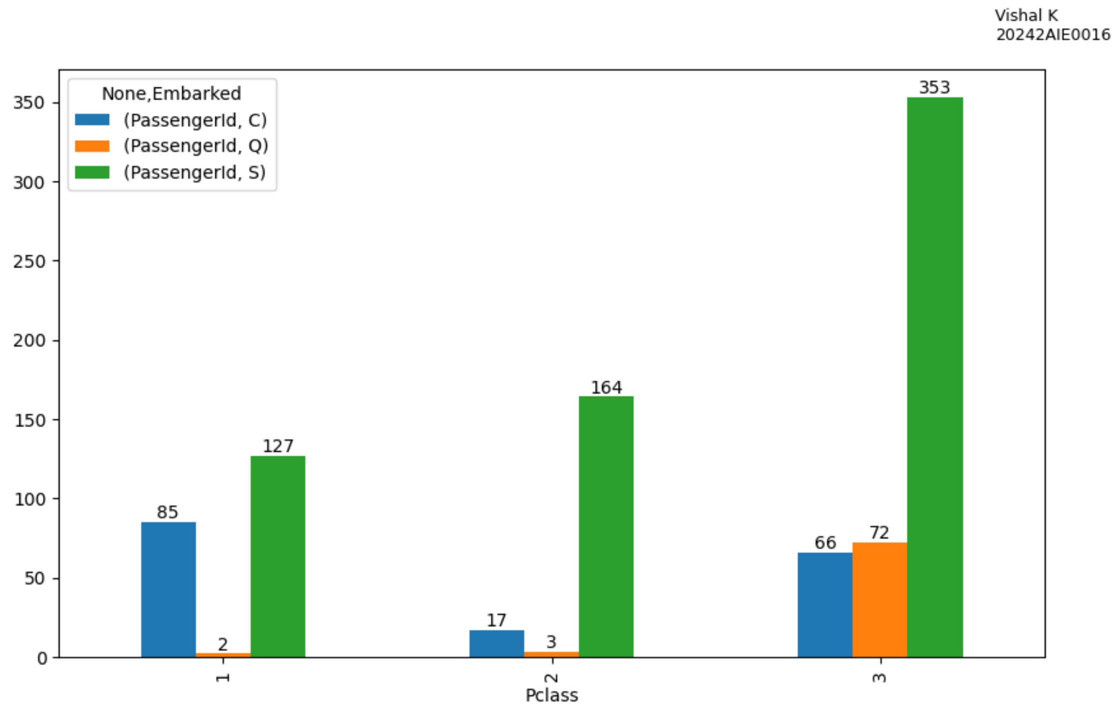
[11]:
```
(       PassengerId
 Sex          female male
 Pclass
 1                94   122
 2                76   108
 3               144   347,
 <Axes: xlabel='Pclass'>)
```



2.6 5) Relation between passengers' booking class and port of embarkation

[12]:
```
make_pivot ('Pclass','Embarked', "Vishal K\n20242AIE0016")
```

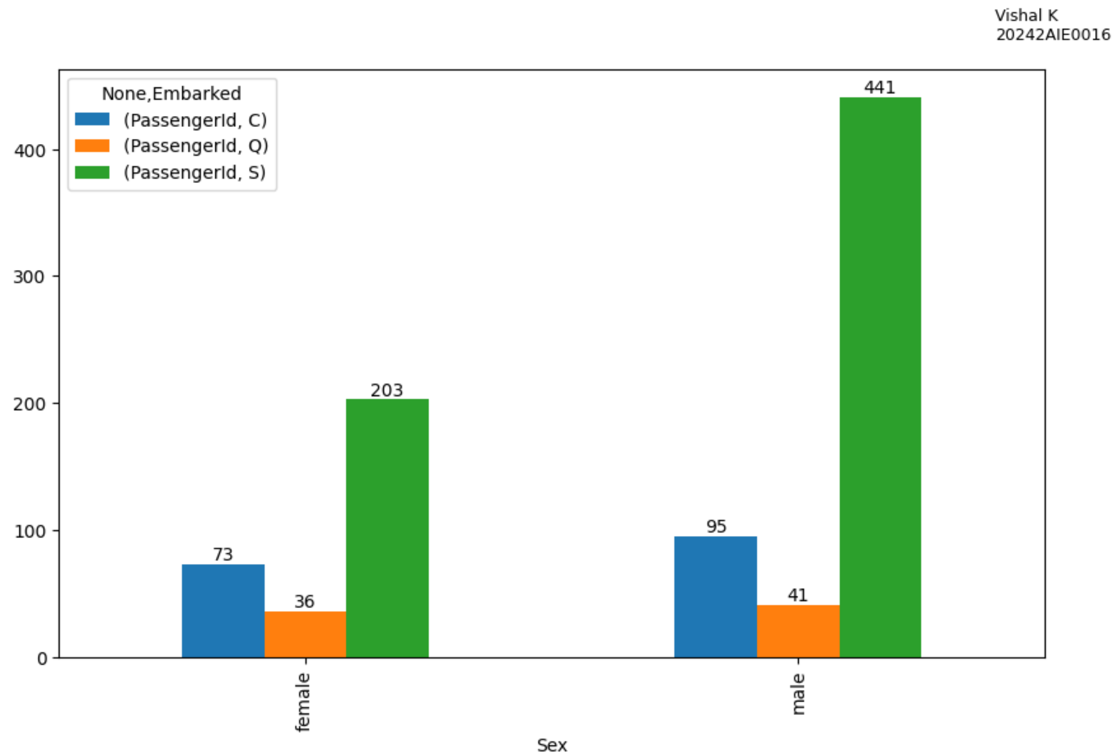
[12]:
```
(          PassengerId
 Embarked            C   Q    S
 Pclass
 1                  85   2  127
 2                  17   3  164
 3                  66  72  353,
 <Axes: xlabel='Pclass'>)
```



2.7 6) Relation between passengers' sex and port of embarkation

[13]:
```
make_pivot ('Sex','Embarked', "Vishal K\n20242AIE0016")
```

[13]:
```
(          PassengerId
 Embarked            C   Q    S
 Sex
 female             73  36  203
 male               95  41  441,
 <Axes: xlabel='Sex'>)
```



3 Canada dataset with Seaborn

[14]:
```
canada = pd.read_csv('data/canada.csv')
canada.describe()
```

[14]:

	1980	1981	1982	1983	1984 \
count	195.000000	195.000000	195.000000	195.000000	195.000000
mean	508.394872	566.989744	534.723077	387.435897	376.497436
std	1949.588546	2152.643752	1866.997511	1204.333597	1198.246371
min	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000	0.000000	0.000000
50%	13.000000	10.000000	11.000000	12.000000	13.000000
75%	251.500000	295.500000	275.000000	173.000000	181.000000
max	22045.000000	24796.000000	20620.000000	10015.000000	10170.000000

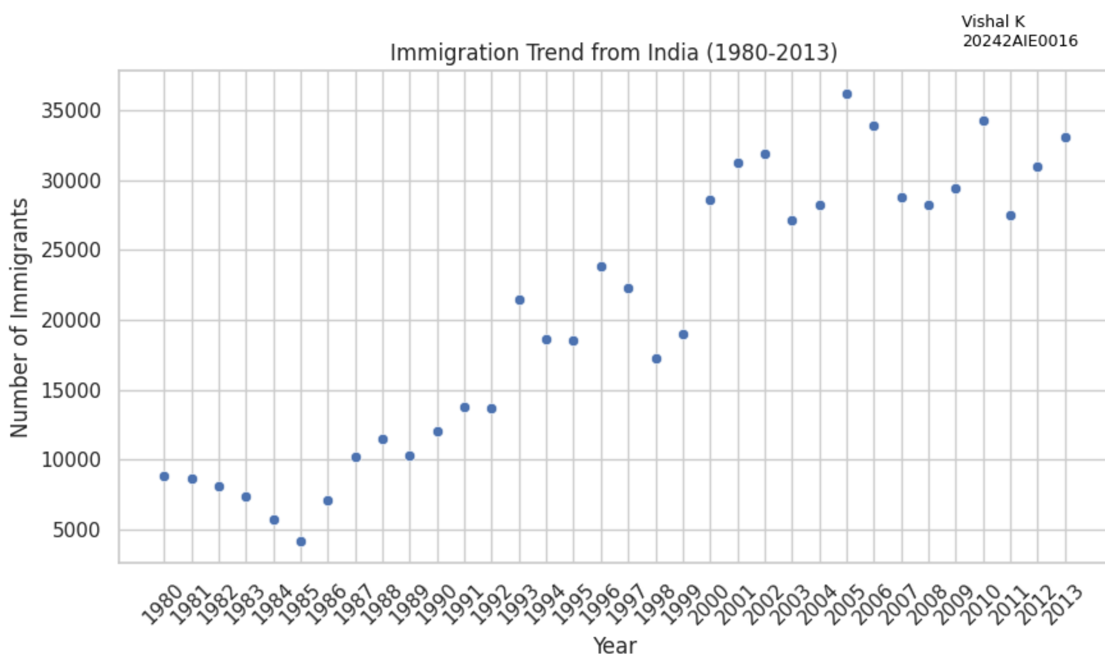
	1985	1986	1987	1988	1989 \
count	195.000000	195.000000	195.000000	195.000000	195.000000
mean	358.861538	441.271795	691.133333	714.389744	843.241026
std	1079.309600	1225.576630	2109.205607	2443.606788	2555.048874
min	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.500000	0.500000	1.000000	1.000000
50%	17.000000	18.000000	26.000000	34.000000	44.000000

```
75%     197.000000   254.000000    434.000000    409.000000    508.500000
max    9564.000000  9470.000000  21337.000000  27359.000000  23795.000000

       …          2005          2006          2007          2008  \
count  …    195.000000    195.000000    195.000000    195.000000
mean   …   1320.292308   1266.958974   1191.820513   1246.394872
std    …   4425.957828   3926.717747   3443.542409   3694.573544
min    …      0.000000      0.000000      0.000000      0.000000
25%    …     28.500000     25.000000     31.000000     31.000000
50%    …    210.000000    218.000000    198.000000    205.000000
75%    …    832.000000    842.000000    899.000000    934.500000
max    …  42584.000000  33848.000000  28742.000000  30037.000000

               2009          2010          2011          2012          2013  \
count    195.000000    195.000000    195.000000    195.000000    195.000000
mean    1275.733333   1420.287179   1262.533333   1313.958974   1320.702564
std     3829.630424   4462.946328   4030.084313   4247.555161   4237.951988
min        0.000000      0.000000      0.000000      0.000000      0.000000
25%       36.000000     40.500000     37.500000     42.500000     45.000000
50%      214.000000    211.000000    179.000000    233.000000    213.000000
75%      888.000000    932.000000    772.000000    783.000000    796.000000
max    29622.000000  38617.000000  36765.000000  34315.000000  34129.000000

               Total
count     195.000000
mean    32867.451282
std     91785.498686
min         1.000000
25%       952.000000
50%      5018.000000
75%     22239.500000
max    691904.000000

[8 rows x 35 columns]
```

3.1 1. Scatter plot: Immigration trend for a single country

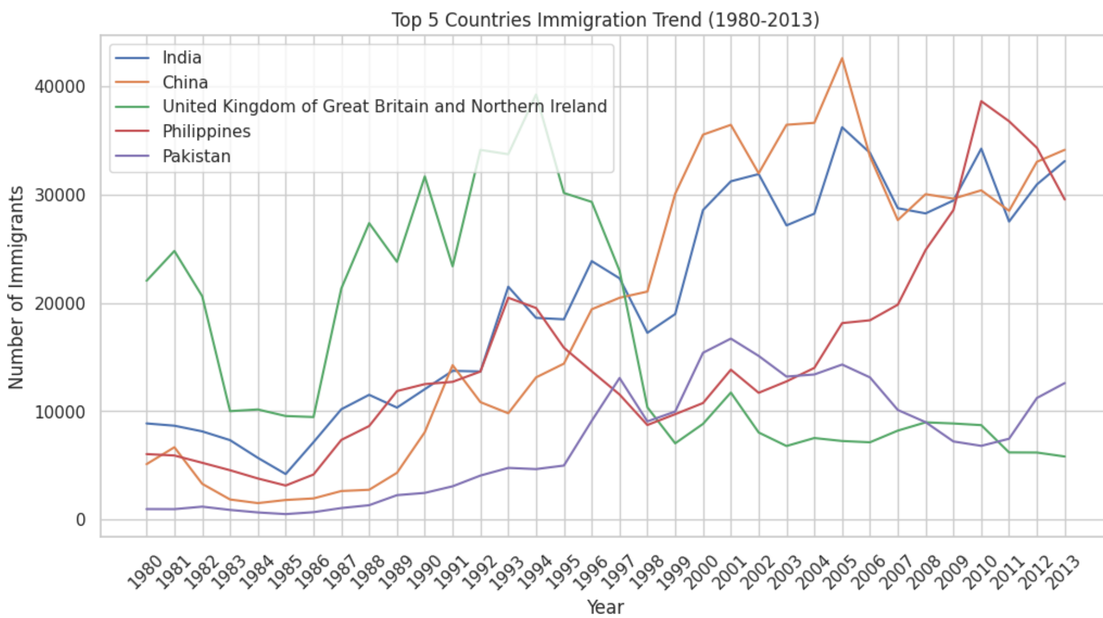
[15]:
```python
years = [str(year) for year in range(1980, 2014)]
canada[years] = canada[years].apply(pd.to_numeric)
canada["Total"] = canada[years].sum(axis=1)

# Set Seaborn style
sns.set(style="whitegrid")

# Scatter plot: Immigration trend for a single country (India)
plt.figure(figsize=(10, 5))
ax = sns.scatterplot(
```

```
    x=years,
    y=canada[canada["Country"] == "India"].iloc[:, 4:-1].values.flatten(),
    marker="o"
)

plt.xticks(rotation=45)
plt.xlabel("Year")
plt.ylabel("Number of Immigrants")
plt.title("Immigration Trend from India (1980-2013)")

# Add textbox at top right corner
ax.text(
    0.85, 1.05, "Vishal K\n20242AIE0016",  # Normalized coordinates
    transform=ax.transAxes,
    fontsize=9, color="black",
    bbox=dict(facecolor="white", alpha=0.8)
)

plt.show()
```



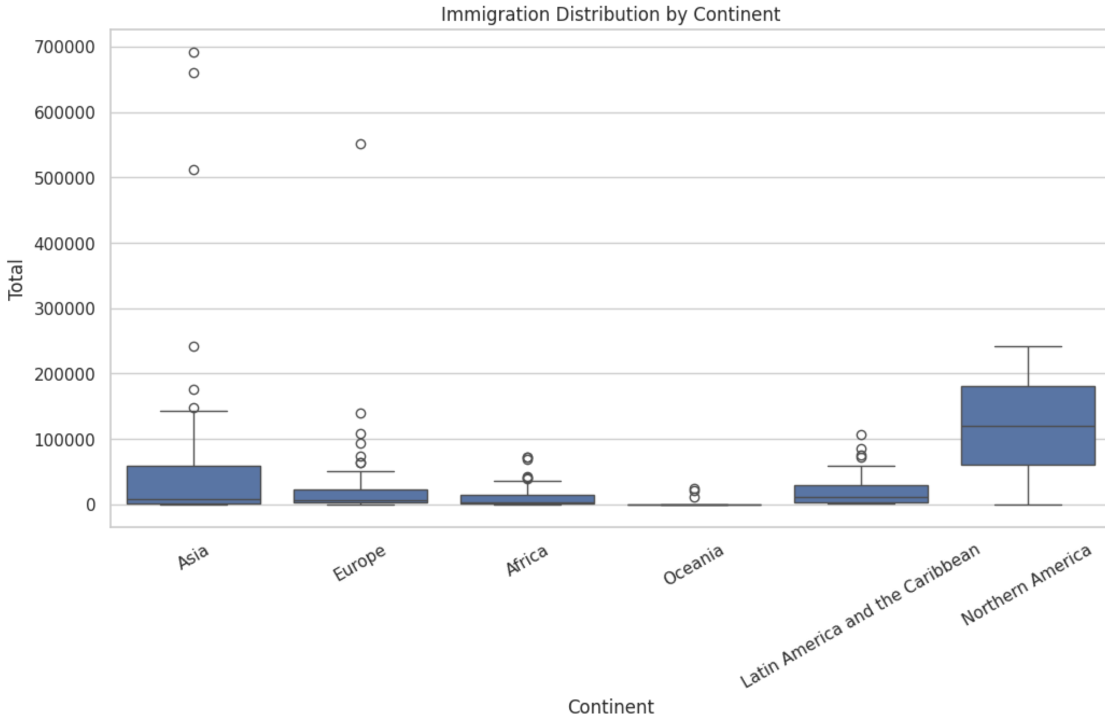
3.2 2. Line plot: Top 5 countries with highest immigration

[16]:
```
top_countries = canada.nlargest(5, "Total")
plt.figure(figsize=(12, 6))
for country in top_countries["Country"]:
    sns.lineplot(x=years, y=canada[canada["Country"] == country].iloc[:, 4:-1].
 ↪values.flatten(), label=country)
plt.xticks(rotation=45)
plt.xlabel("Year")
plt.ylabel("Number of Immigrants")
plt.title("Top 5 Countries Immigration Trend (1980-2013)")
plt.legend()
plt.show()
```



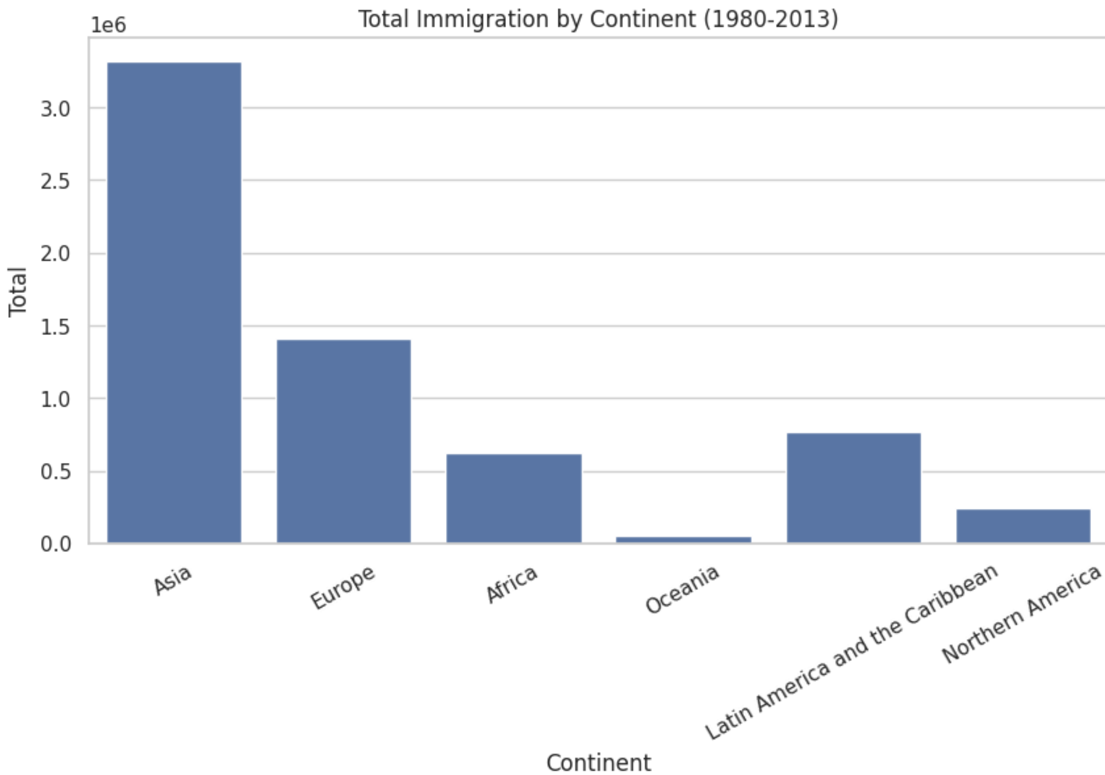
3.3 3. Box plot: Distribution of immigration per continent

[17]:
```
plt.figure(figsize=(12, 6))
sns.boxplot(x="Continent", y="Total", data=canada)
plt.xticks(rotation=30)
plt.title("Immigration Distribution by Continent")
plt.show()
```



3.4 4. Bar plot: Total immigration by continent

[18]:
```
plt.figure(figsize=(10, 5))
sns.barplot(x="Continent", y="Total", data=canada, estimator=sum, errorbar=None)
plt.xticks(rotation=30)
plt.title("Total Immigration by Continent (1980-2013)")
plt.show()
```


Total Immigration by Continent (1980-2013)
1e6
3.0
2.5
2.0
1.5
1.0
0.5
0.0
Total
Asia
Europe
Africa
Oceania
Latin America and the Caribbean
Northern America
Continent

```
[ ]:
```