5300-project-group-5

April 27, 2024

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

[2]:
```
# Step 1: Data Collection

# Loading datasets
data_science_salaries = pd.read_csv("data_science_salaries.csv")
countries_table = pd.read_csv("countries-table.csv")
```

[3]:
```
# Step 2: Data Preparation

# Checking for missing values, outliers, and inconsistencies
print("Data Science Salaries Dataset:")
print(data_science_salaries.info())
print(data_science_salaries.describe())
```

```
Data Science Salaries Dataset:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6599 entries, 0 to 6598
Data columns (total 11 columns):
 #   Column              Non-Null Count  Dtype 
---  ------              --------------  ----- 
 0   job_title           6599 non-null   object
 1   experience_level    6599 non-null   object
 2   employment_type     6599 non-null   object
 3   work_models         6599 non-null   object
 4   work_year           6599 non-null   int64 
 5   employee_residence  6599 non-null   object
 6   salary              6599 non-null   int64 
 7   salary_currency     6599 non-null   object
 8   salary_in_usd       6599 non-null   int64 
 9   company_location    6599 non-null   object
 10  company_size        6599 non-null   object
dtypes: int64(3), object(8)
memory usage: 567.2+ KB
None
```

```
         work_year        salary  salary_in_usd
count  6599.000000  6.599000e+03    6599.000000
mean   2022.818457  1.792833e+05  145560.558569
std       0.674809  5.263722e+05   70946.838070
min    2020.000000  1.400000e+04   15000.000000
25%    2023.000000  9.600000e+04   95000.000000
50%    2023.000000  1.400000e+05  138666.000000
75%    2023.000000  1.875000e+05  185000.000000
max    2024.000000  3.040000e+07  750000.000000
```

[4]:
```
print("\nCountries Table Dataset:")
print(countries_table.info())
print(countries_table.describe())
```

```
Countries Table Dataset:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 234 entries, 0 to 233
Data columns (total 19 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   country          234 non-null    object
 1   rank             234 non-null    int64
 2   area             234 non-null    float64
 3   landAreaKm       234 non-null    float64
 4   cca2             233 non-null    object
 5   cca3             234 non-null    object
 6   netChange        226 non-null    float64
 7   growthRate       234 non-null    float64
 8   worldPercentage  228 non-null    float64
 9   density          234 non-null    float64
 10  densityMi        234 non-null    float64
 11  place            234 non-null    int64
 12  pop1980          234 non-null    int64
 13  pop2000          234 non-null    int64
 14  pop2010          234 non-null    int64
 15  pop2022          234 non-null    int64
 16  pop2023          234 non-null    int64
 17  pop2030          234 non-null    int64
 18  pop2050          234 non-null    int64
dtypes: float64(7), int64(9), object(3)
memory usage: 34.9+ KB
None
             rank          area    landAreaKm   netChange  growthRate  \
count  234.000000  2.340000e+02  2.340000e+02  226.000000  234.000000
mean   117.500000  5.814500e+05  5.571123e+05    0.010306    0.009737
std     67.694165  1.761841e+06  1.689972e+06    0.034774    0.012350
min      1.000000  4.400000e-01  4.400000e-01   -0.028600   -0.074500
```

```
25%     59.250000  2.650000e+03  2.625875e+03    0.000000    0.002325
50%    117.500000  8.119950e+04  7.568925e+04    0.000900    0.008200
75%    175.750000  4.304258e+05  4.047876e+05    0.008000    0.016850
max    234.000000  1.709824e+07  1.637687e+07    0.418400    0.049800

       worldPercentage       density     densityMi       place       pop1980  \
count       228.000000    234.000000    234.000000  234.000000  2.340000e+02
mean          0.004407    451.288182   1168.836388  439.085470  1.898462e+07
std           0.017375   1979.362419   5126.548664  253.295484  8.178519e+07
min           0.000000      0.138000      0.357400    4.000000  7.330000e+02
25%           0.000100     39.747650    102.946450  223.000000  2.296142e+05
50%           0.000750     97.481000    252.475800  439.000000  3.141146e+06
75%           0.002925    242.928650    629.185350  659.750000  9.826054e+06
max           0.178500  21402.705200  55433.006400  894.000000  9.823725e+08

            pop2000       pop2010       pop2022       pop2023       pop2030  \
count  2.340000e+02  2.340000e+02  2.340000e+02  2.340000e+02  2.340000e+02
mean   2.626947e+07  2.984524e+07  3.407441e+07  3.437442e+07  3.651461e+07
std    1.116982e+08  1.242185e+08  1.367664e+08  1.373864e+08  1.417827e+08
min    6.510000e+02  5.960000e+02  5.100000e+02  5.180000e+02  5.610000e+02
25%    3.272420e+05  3.931490e+05  4.197385e+05  4.225982e+05  4.561490e+05
50%    4.292907e+06  4.942770e+06  5.559944e+06  5.643895e+06  6.178231e+06
75%    1.576230e+07  1.915957e+07  2.247650e+07  2.324537e+07  2.616311e+07
max    1.264099e+09  1.348191e+09  1.425887e+09  1.428628e+09  1.514994e+09

            pop2050
count  2.340000e+02
mean   4.148628e+07
std    1.481676e+08
min    7.310000e+02
25%    5.466058e+05
50%    6.352397e+06
75%    3.568614e+07
max    1.670491e+09
```

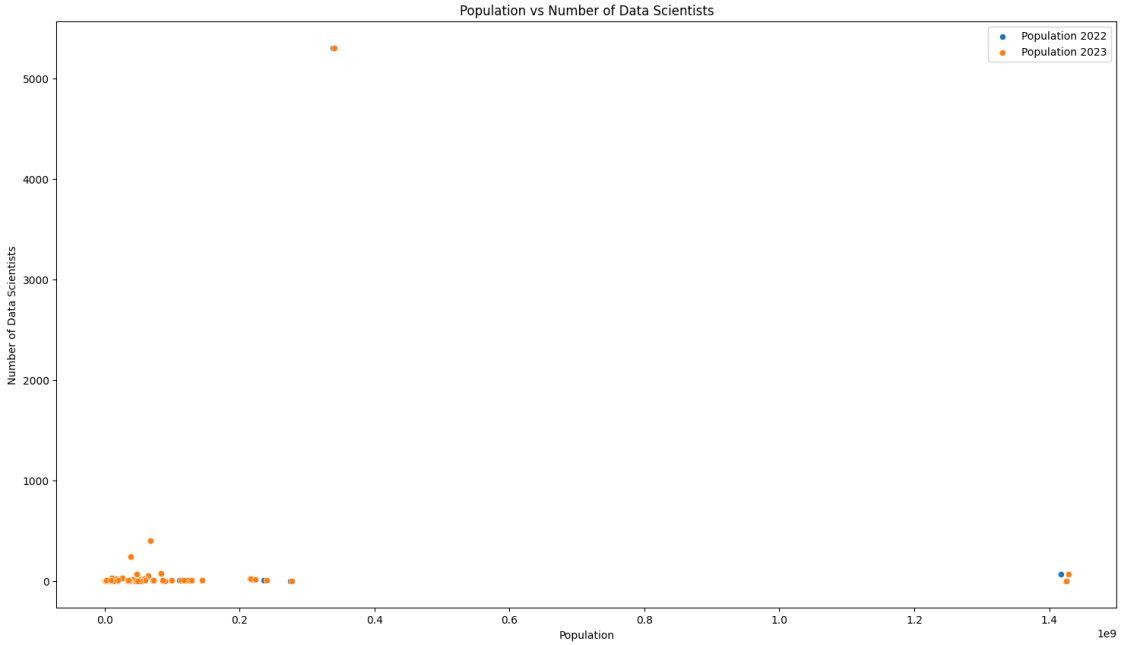
[5]:
```python
# Step 3: Merge Datasets

# Merging datasets based on common attributes
merged_data = pd.merge(data_science_salaries, countries_table,␣
 ↪left_on='employee_residence', right_on='country')
```

[6]:
```python
# Step 4: Data Analysis

# Investigate the relationship between population size and the number of data␣
 ↪scientists employed in each country
country_data_scientists = merged_data.groupby('country')['job_title'].count().
 ↪reset_index()
```

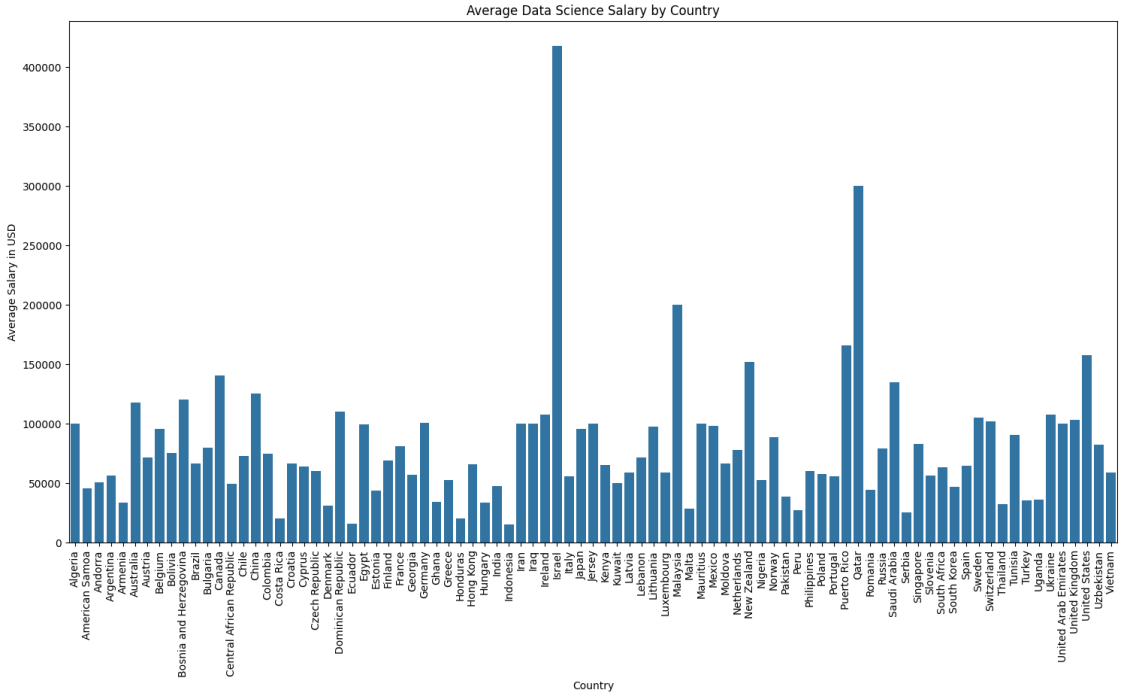
```
country_population_2022 = merged_data[['country', 'pop2022']].drop_duplicates()
country_population_2023 = merged_data[['country', 'pop2023']].drop_duplicates()
country_analysis = pd.merge(country_data_scientists, country_population_2022,␣
 ↪on='country')
country_analysis = pd.merge(country_analysis, country_population_2023,␣
 ↪on='country')
```

[7]:
```
# Analyze variations in data science salaries across different countries and␣
 ↪their populations
salary_by_country = merged_data.groupby('country')['salary_in_usd'].mean().
 ↪reset_index()
salary_population_analysis = pd.merge(salary_by_country,␣
 ↪country_population_2023, on='country')

# Explore local trends in data science demand and salary structures
local_trends = merged_data.groupby(['country', 'work_year'])['salary_in_usd'].
 ↪mean().reset_index()
```

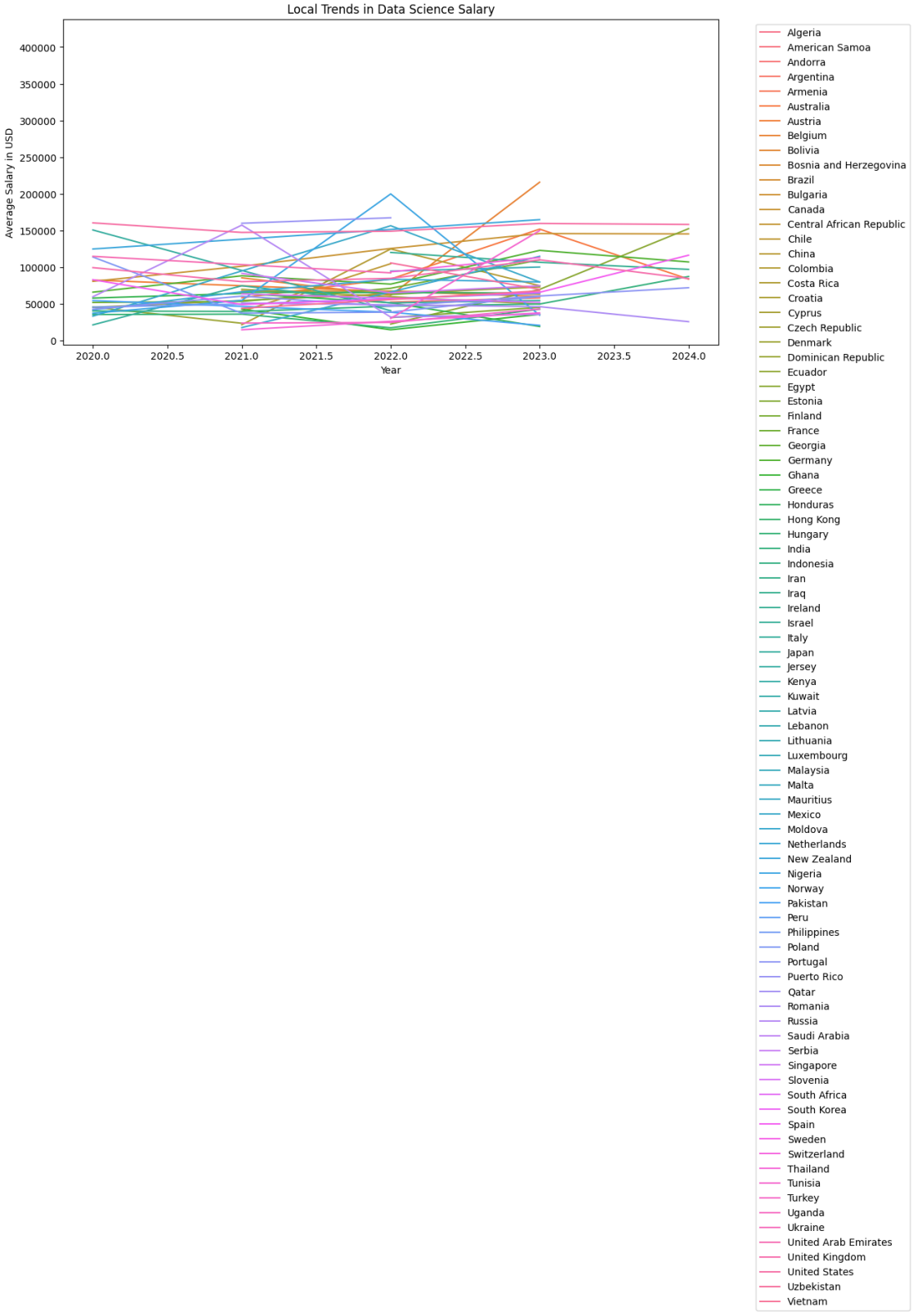
[8]:
```
# Step 5: Data Visualization

# Investigating the relationship between population size and the number of data␣
 ↪scientists employed in each country
plt.figure(figsize=(18, 10))
sns.scatterplot(x='pop2022', y='job_title', data=country_analysis,␣
 ↪label='Population 2022')
sns.scatterplot(x='pop2023', y='job_title', data=country_analysis,␣
 ↪label='Population 2023')
plt.xlabel('Population')
plt.ylabel('Number of Data Scientists')
plt.title('Population vs Number of Data Scientists')
plt.legend()
plt.show()
```



[9]:
```
# Analyze variations in data science salaries across different countries and
  ↪their populations
plt.figure(figsize=(18, 9))
sns.barplot(x='country', y='salary_in_usd', data=salary_population_analysis)
plt.xlabel('Country')
plt.ylabel('Average Salary in USD')
plt.title('Average Data Science Salary by Country')
plt.xticks(rotation=90)
plt.show()
```



[10]:
```
# Exploring local trends in data science demand and salary structures
plt.figure(figsize=(12, 6))
sns.lineplot(x='work_year', y='salary_in_usd', hue='country', data=local_trends)
plt.xlabel('Year')
plt.ylabel('Average Salary in USD')
plt.title('Local Trends in Data Science Salary')
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')
plt.show()
```


Local Trends in Data Science Salary
Average Salary in USD
Year
0
50000
100000
150000
200000
250000
300000
350000
400000
2020.0
2020.5
2021.0
2021.5
2022.0
2022.5
2023.0
2023.5
2024.0
Algeria
American Samoa
Andorra
Argentina
Armenia
Australia
Austria
Belgium
Bolivia
Bosnia and Herzegovina
Brazil
Bulgaria
Canada
Central African Republic
Chile
China
Colombia
Costa Rica
Croatia
Cyprus
Czech Republic
Denmark
Dominican Republic
Ecuador
Egypt
Estonia
Finland
France
Georgia
Germany
Ghana
Greece
Honduras
Hong Kong
Hungary
India
Indonesia
Iran
Iraq
Ireland
Israel
Italy
Japan
Jersey
Kenya
Kuwait
Latvia
Lebanon
Lithuania
Luxembourg
Malaysia
Malta
Mauritius
Mexico
Moldova
Netherlands
New Zealand
Nigeria
Norway
Pakistan
Peru
Philippines
Poland
Portugal
Puerto Rico
Qatar
Romania
Russia
Saudi Arabia
Serbia
Singapore
Slovenia
South Africa
South Korea
Spain
Sweden
Switzerland
Thailand
Tunisia
Turkey
Uganda
Ukraine
United Arab Emirates
United Kingdom
United States
Uzbekistan
Vietnam

[11]:
```
# Saving merged data to a new file
merged_data.to_csv("merged_data.csv", index=False)
```