DSE5002-Project2

May 3, 2024

You are a data scientist and would like to know where the top 5 places in the world (country or city) where your salary (in USD) will go the farthest with respect to each individual index within the cost_of_living.csv file. Provide a simple statistical analysis in a Jupyter Notebook file and provide visualizations to support your analysis (I am looking for data wrangling more than anything)

[170]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from itertools import chain
```

[268]:
```
## Data Importing and Orgnanising

#Import Cost of Living dataset
cost_of_living = pd.read_csv('/Users/nathanmonges/Documents/DSE5002/
 ↪cost_of_living.csv')

#Rename columns
cost_of_living.columns = ["rank", "city", "col_index", "rent_index",␣
 ↪"col_plus_rent_index",
                          "groceries_index",␣
 ↪"restaurant_price_index","local_purchasing_power_index"]
cost_of_living = cost_of_living.drop("rank", axis=1) #remove rank column full␣
 ↪of "nan"
cost_of_living.head()

## Filtering only US cities from Cost of Living Data
cost_of_living[['City', 'State', 'Country']] = cost_of_living['city'].str.
 ↪split(', ', expand=True)

us_cost_of_living = cost_of_living[cost_of_living["Country"] == "United States"]

us_cost_of_living.drop("city", axis=1)
us_cost_of_living.insert(0,"City", us_cost_of_living.pop("City"))
us_cost_of_living.insert(1,"State", us_cost_of_living.pop("State"))

us_cost_of_living = us_cost_of_living.drop(["city","Country"], axis=1)
```

```
us_cost_of_living = us_cost_of_living.reset_index().drop("index", axis=1)
us_cost_of_living

## Filtering only International Countries/Cities from C.O.L dataset
col_international = cost_of_living[cost_of_living["Country"] != "United States"]
col_international = col_international.drop(["City", "State", "Country"], axis=1)

split_cities = col_international['city'].str.split(", ", n=1, expand=True) #␣
 ↪Split "city" column into separate columns for country and city

col_international['Country'] = split_cities[1]
col_international['City'] = split_cities[0]
col_international.insert(0,"Country", col_international.pop("Country"))
col_international.insert(1,"City", col_international.pop("City"))

col_international = col_international.reset_index().drop(["index", "city"],␣
 ↪axis=1)
#-------------------------------------

#Import DS Salaries dataset
salaries = pd.read_csv('/Users/nathanmonges/Documents/DSE5002/ds_salaries.csv')
#Fix columns
salaries = salaries.drop("Ramk", axis=1)
salaries.head()

#-------------------------------------

#Import Salaries Levels dataset
levels = pd.read_csv('/Users/nathanmonges/Documents/DSE5002/
 ↪Levels_Fyi_Salary_Data (2).csv')
#Remove unnessecary columns
levels_columns = [col for col in levels.columns if 'Degree' not in col and␣
 ↪'Race' not in col] #loop to remove cols with specific word
levels = levels[levels_columns]
levels = levels.drop(["bonus", "stockgrantvalue", "totalyearlycompensation",␣
 ↪"tag",
                      "otherdetails", "gender", "Highschool", "Some_College",␣
 ↪"Education"], axis=1)
#Rename columns
levels.columns = ["time_stamp",        "company", "level",        "title",␣
 ↪"location",
                  "years_of_experience", "years_at_company", "base_salary",␣
 ↪"city_id", "dma_id", "row_number"]
```

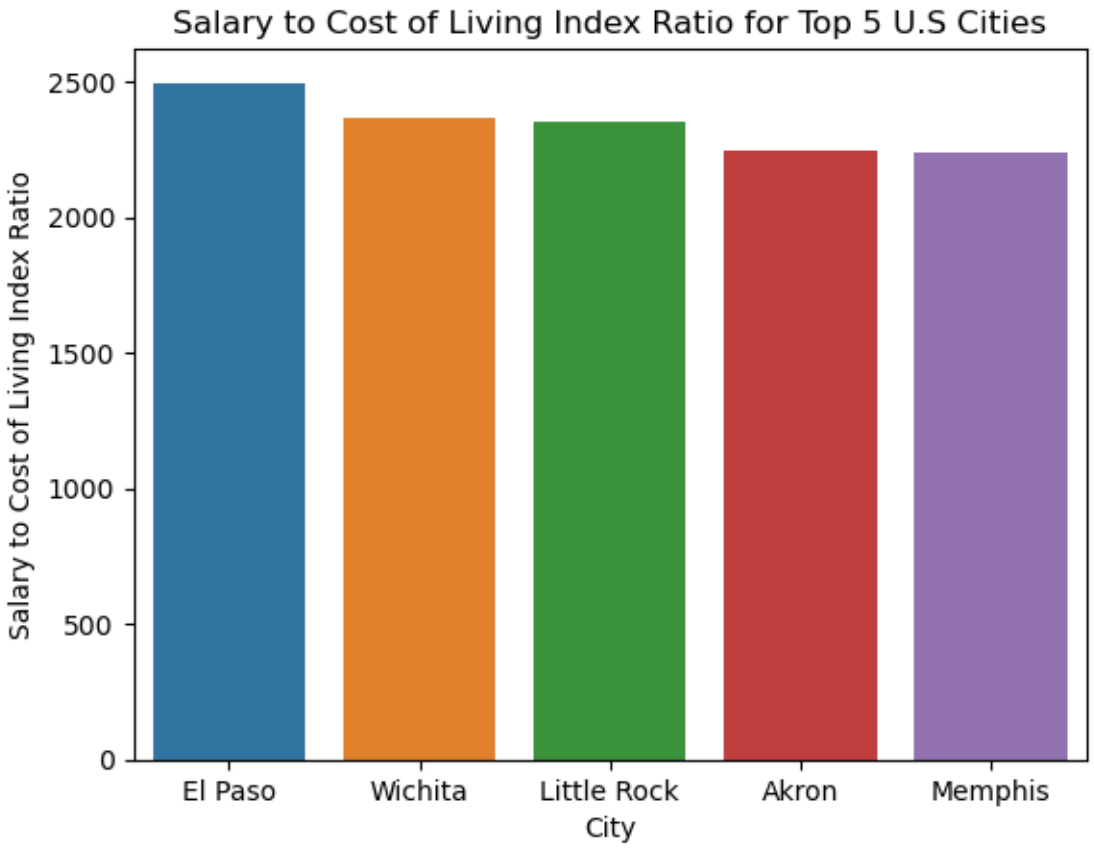
[251]:
```
## Data Filtering - Determinining my salary as a Data Scientist for comparison␣
 ↪of cost of living indexes
```

```
## Join both salary datasets and get mean of data scientist role salary - average will be my salary for comparison

#Filter salaries to only data scientists
salaries = salaries[salaries.job_title == "Data Scientist"]

#Filter levels data
levels = levels.rename(columns = {"base_salary":"salary_in_usd"}) #match col names to salaries
levels = levels[levels.title == "Data Scientist"] #filter levels to only data scientist roles

#Join levels to salaries - remove columns
ds_salaries = salaries.merge(levels, how="left", on="salary_in_usd").drop(["time_stamp", "company", "level", "location",
 "years_of_experience", "years_at_company", "city_id",
 "dma_id", "row_number", "title"], axis=1)
#Determine my salary by computing avg of ds salaries
my_salary = round(ds_salaries.salary_in_usd.mean()) #139601
print("My salary as a Data Scientist is $",my_salary)
```

```
My salary as a Data Scientist is $ 139601
```

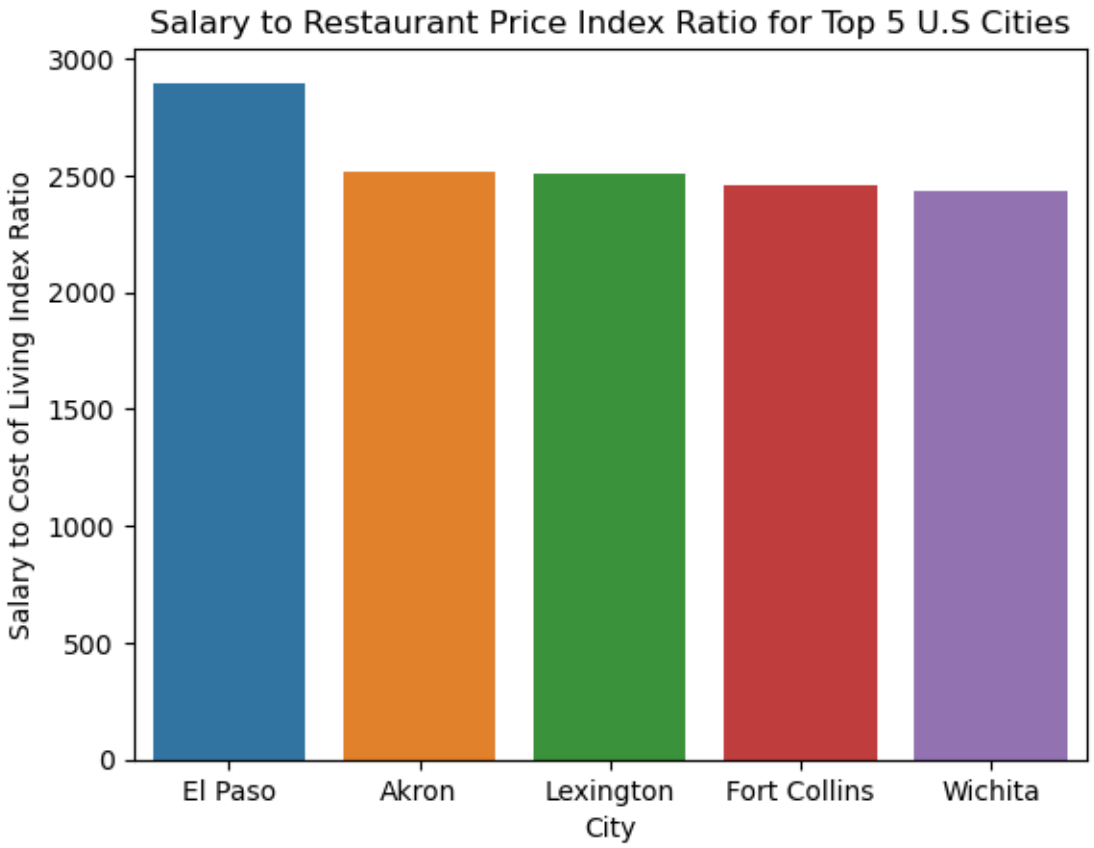
Now that my salary is determined, I am interested in viewing the top 5 cities where my salary will take my the furthest. To do so I will determine the ratio of my salary to the cost of living index and chart the 5 cities with the largest ratio.

[252]:
```
## Cost of Living Index - US - Top 5 Citites

us_cost_of_living["salary_to_col_index_ratio"] = my_salary / us_cost_of_living.col_index
# Top 5 US Cities from salary to C.O.L Index Ratio
top_5_cities_us = us_cost_of_living.nlargest(5, "salary_to_col_index_ratio")
top_5_cities_us


sns.barplot(data=top_5_cities_us, y="salary_to_col_index_ratio", x="City")
plt.title('Salary to Cost of Living Index Ratio for Top 5 U.S Cities')
plt.xlabel('City')
plt.ylabel('Salary to Cost of Living Index Ratio')
plt.show()
```



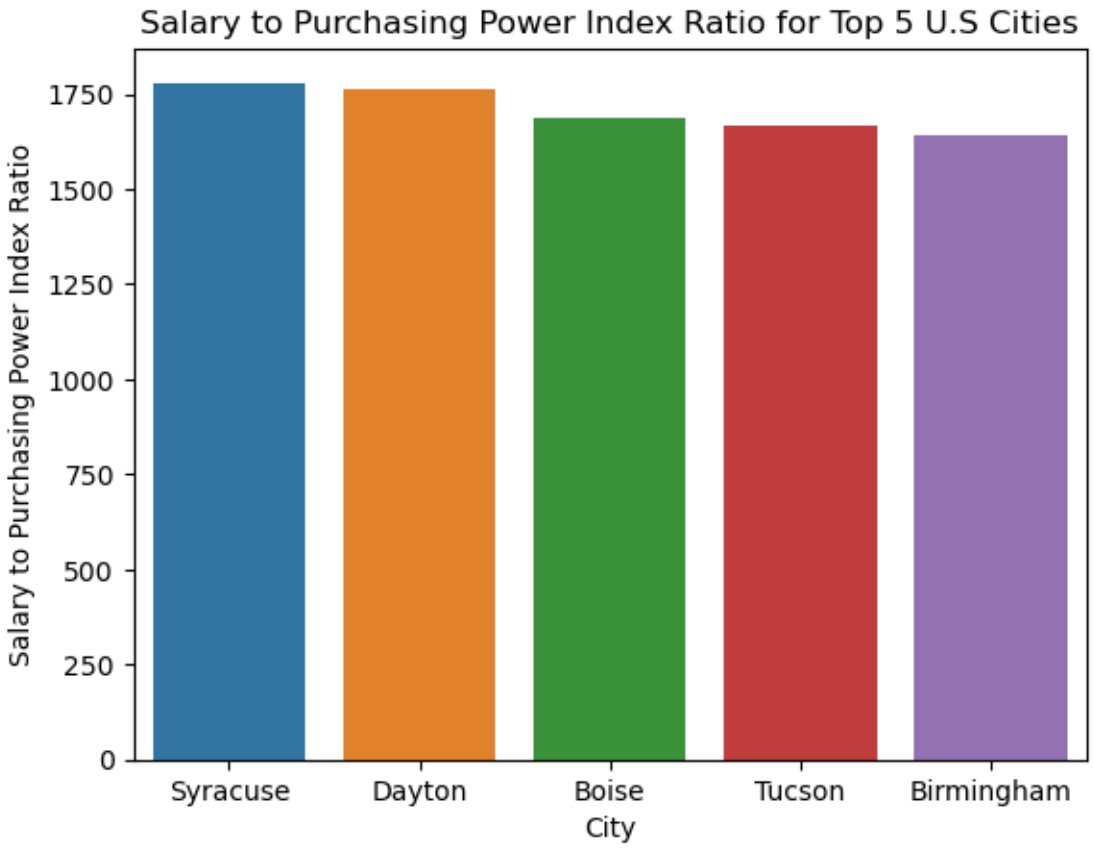
Now that I know the top 5 cities for me based of cost of living index. Since I love ordering take-out and going to restaurants, I am interested in seeing the top 5 cities where my salary will work the best based on the restaurant index of these cities.

[250]:
```
## Restaurant Index - Top 5 US Cities

us_cost_of_living["salary_to_restaurant_index_ratio"] = my_salary /␣
 ↪us_cost_of_living.restaurant_price_index
# Top 5 US Cities from salary to Restaurant Index Ratio
top_5_cities_us_rent = us_cost_of_living.nlargest(5,␣
 ↪"salary_to_restaurant_index_ratio")
top_5_cities_us_rent

sns.barplot(data=top_5_cities_us_rent, y="salary_to_restaurant_index_ratio",␣
 ↪x="City")
plt.title('Salary to Restaurant Price Index Ratio for Top 5 U.S Cities')
plt.xlabel('City')
plt.ylabel('Salary to Cost of Living Index Ratio')
plt.show()
```



Now that I can see the top U.S cities for me realtive to cost of living index and restaurant index. I want to view the top U.S cities for my salary relative to the local purchasing power index.

[260]:
```
## Purchasing Power Index - Top 5 US Cities

us_cost_of_living["salary_to_purchasing_power_index_ratio"] = my_salary /␣
 ↪us_cost_of_living.local_purchasing_power_index
# Top 5 US Cities from salary to Purchasing Power Index Ratio
top_5_cities_us_pp = us_cost_of_living.nlargest(5,␣
 ↪"salary_to_purchasing_power_index_ratio")
top_5_cities_us_pp

sns.barplot(data=top_5_cities_us_pp,␣
 ↪y="salary_to_purchasing_power_index_ratio", x="City")
plt.title('Salary to Purchasing Power Index Ratio for Top 5 U.S Cities')
plt.xlabel('City')
plt.ylabel('Salary to Purchasing Power Index Ratio')
plt.show()
```



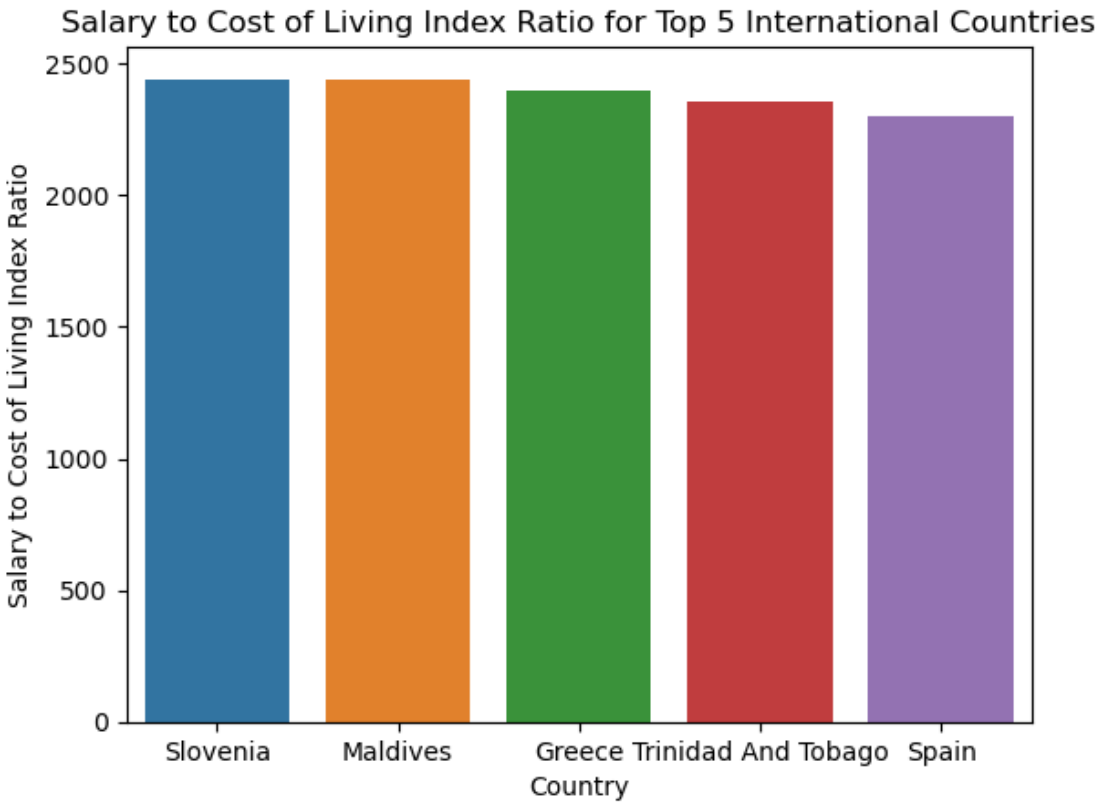
Now, I have narrowed down my search to the top 5 cities in the United States where my salary will go the furtherst when comapred to the index I am most interested in (Cost of Living, Restaruant, and Purcashing Power).

It is a good idea to view how far my salary will take me when compared to cost of living indexes in countries outside of the United States. But to this, I beleive I should disclude countries that are too "cheap" for me, or countries where I am already sure that I can afford. To do this, I will first find the lowest index for the categories im interested in, for citites in the U.S. This minimum index value will be the threshold value for the data in my international cost of living dataset. So, I will be excluding the countries that are beneath the lowest index value related to U.S cities for the index categories I am interested to chart.

[331]:
```
## Cost of Living Index - Top 5 International Countries

#Filter data to set baseline index
us_cost_of_living.col_index.min() #minimum of 55.92
col_international_filtered = col_international[col_international.col_index >=␣
↪55.92]

#Calulating ratio and adding to df
```

```
col_international_filtered["salary_to_col_index_ratio"] = my_salary /␣
 ↪col_international_filtered.col_index

#Create new df of averages cost of living index for each country
country_mean_ratios_col = col_international_filtered.
 ↪groupby("Country")["salary_to_col_index_ratio"].mean().reset_index()
top_5_countries_col = country_mean_ratios_col.
 ↪nlargest(5,"salary_to_col_index_ratio") #filter down top 5

sns.barplot(data=top_5_countries_col, y="salary_to_col_index_ratio",␣
 ↪x="Country")
plt.title("Salary to Cost of Living Index Ratio for Top 5 International␣
 ↪Countries")
plt.xlabel("Country")
plt.ylabel("Salary to Cost of Living Index Ratio")
plt.show()
```

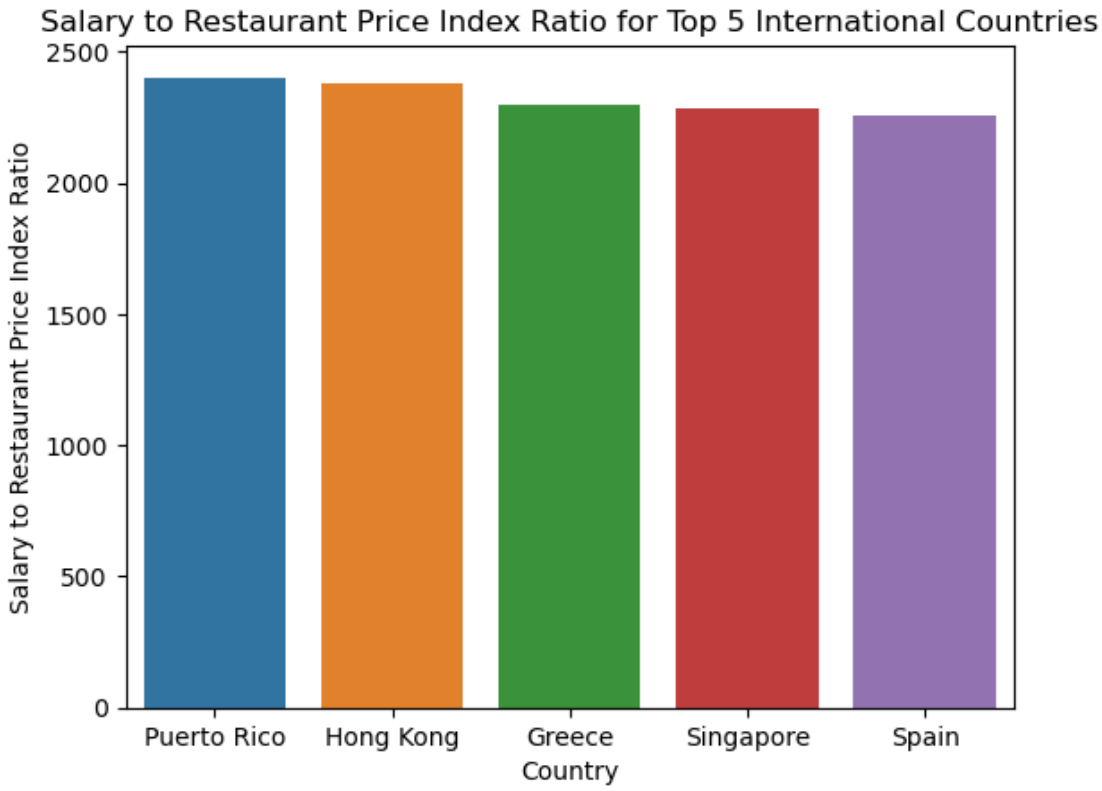
```
/var/folders/f4/pcf9r9h91lbgszy1pxgprndr0000gn/T/ipykernel_52113/2137850536.py:8
: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  col_international_filtered["salary_to_col_index_ratio"] = my_salary /
col_international_filtered.col_index
```



[346]:
```
#Filter data to set baseline index
col_international_filtered_rest = col_international[col_international.
 ↪restaurant_price_index >= 55.92]

#Calulating ratio and adding to df
col_international_filtered_rest["salary_to_restaurant_index_ratio"] = my_salary␣
 ↪/ col_international_filtered.restaurant_price_index

#Create new df of averages cost of living index for each country
country_mean_ratios_rest = col_international_filtered_rest.
 ↪groupby("Country")["salary_to_restaurant_index_ratio"].mean().reset_index()
top_5_countries_rest = country_mean_ratios_rest.
 ↪nlargest(5,"salary_to_restaurant_index_ratio") #filter down top 5
top_5_countries_rest#


sns.barplot(data=top_5_countries_rest, y="salary_to_restaurant_index_ratio",␣
 ↪x="Country")
plt.title("Salary to Restaurant Price Index Ratio for Top 5 International␣
 ↪Countries")
```

```
plt.xlabel("Country")
plt.ylabel("Salary to Restaurant Price Index Ratio")
plt.show()
```

```
/var/folders/f4/pcf9r9h91lbgszy1pxgprndr0000gn/T/ipykernel_52113/1987675042.py:5
: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  col_international_filtered_rest["salary_to_restaurant_index_ratio"] =
my_salary / col_international_filtered.restaurant_price_index
```

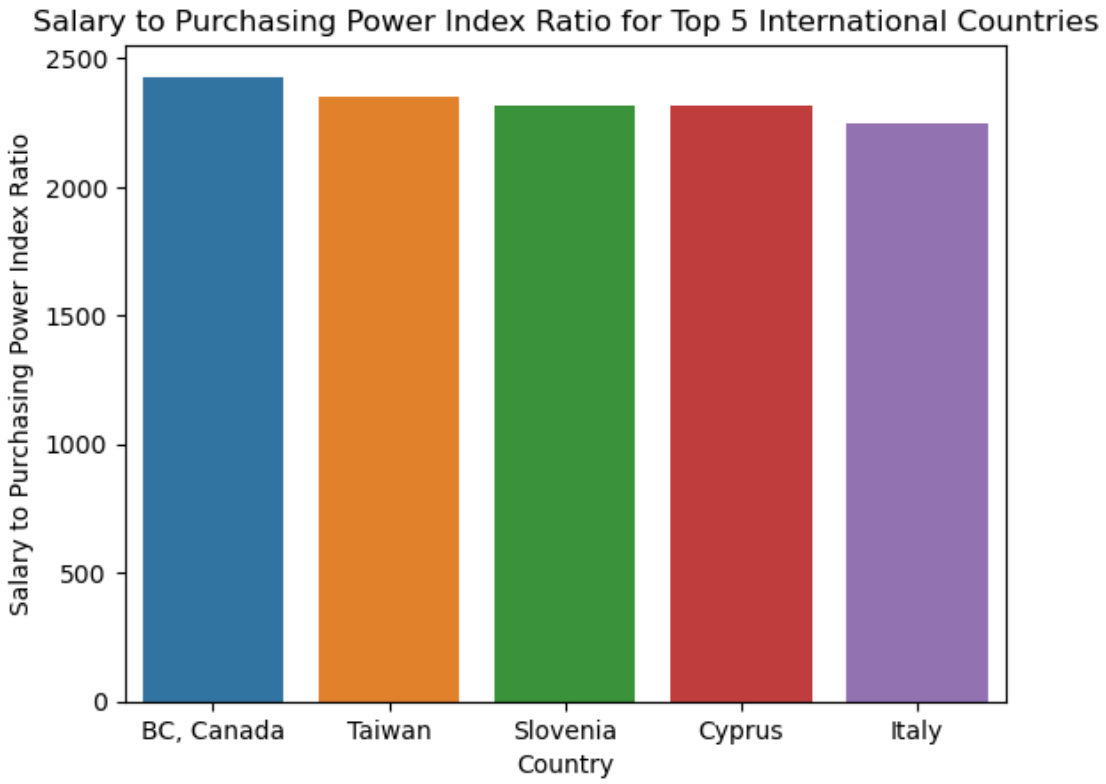


Finally, I will rank the top 5 internatioinal countries where my salary will take me the farthest in terms of the relative local purchasing pwoer.

[349]:
```
## ## Purchasing Power Index - Top 5 US Cities
col_international_filtered_purch = col_international[col_international.
  ↪local_purchasing_power_index >= 55.92]

#Calulating ratio and adding to df
```

```
col_international_filtered_purch["salary_to_purchasing_power_index_ratio"] =␣
 ↪my_salary / col_international_filtered.local_purchasing_power_index

#Create new df of averages cost of living index for each country
country_mean_ratios_purch = col_international_filtered_purch.
 ↪groupby("Country")["salary_to_purchasing_power_index_ratio"].mean().
 ↪reset_index()
top_5_countries_purch = country_mean_ratios_purch.
 ↪nlargest(5,"salary_to_purchasing_power_index_ratio") #filter down top 5
top_5_countries_purch


sns.barplot(data=top_5_countries_purch,␣
 ↪y="salary_to_purchasing_power_index_ratio", x="Country")
plt.title("Salary to Purchasing Power Index Ratio for Top 5 International␣
 ↪Countries")
plt.xlabel("Country")
plt.ylabel("Salary to Purchasing Power Index Ratio")
plt.show()
```

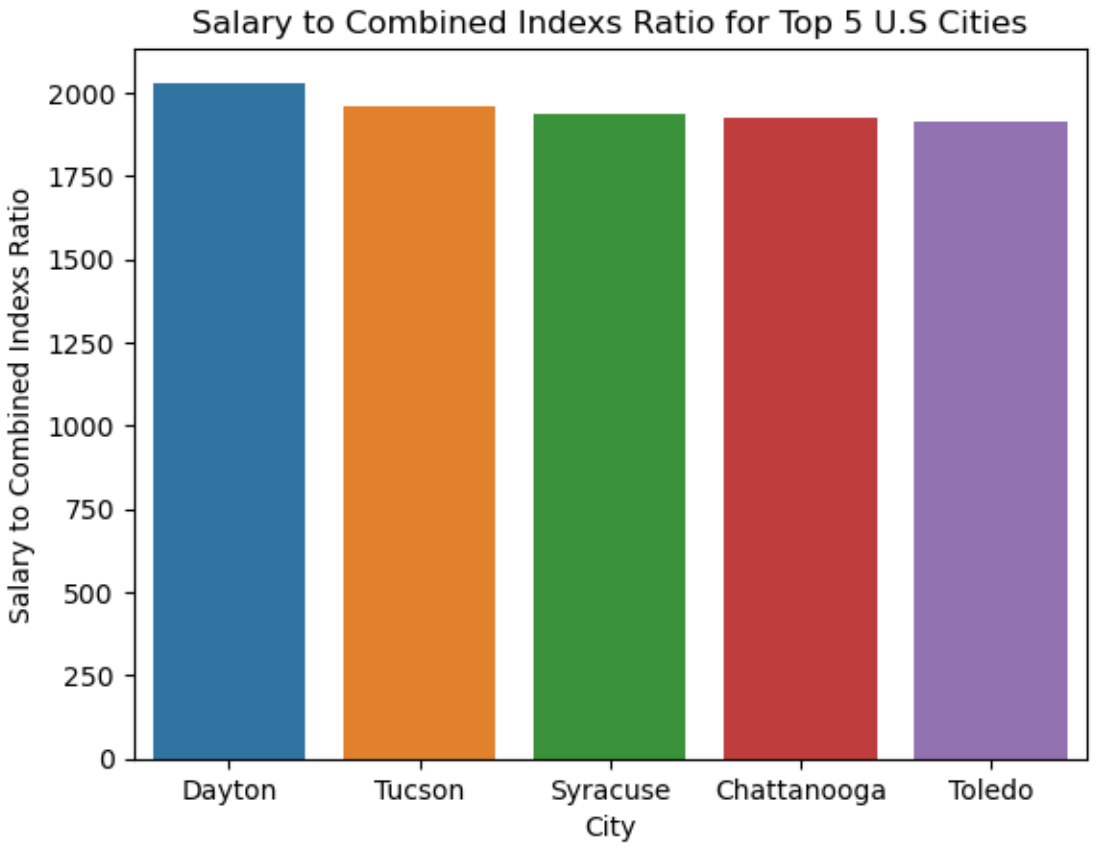
```
/var/folders/f4/pcf9r9h91lbgszy1pxgprndr0000gn/T/ipykernel_52113/1345716635.py:5
: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  col_international_filtered_purch["salary_to_purchasing_power_index_ratio"] =
my_salary / col_international_filtered.local_purchasing_power_index
```



To conclude my analysis, I will be combining the cost of living indexes I am most interested in, calcluating the average and calculating the ratio of the value to my salary for U.S cities and international countries, in order to determine which are the top 5 cities and countries for with my salary.

[381]:
```
## Indexes Combined - Top 5 U.S Citites
grouped_mean_us = us_cost_of_living.groupby("City")[["col_index",␣
 ↪"restaurant_price_index", "local_purchasing_power_index"]].mean()
grouped_mean_us["index_means_combined"] = grouped_mean_us.mean(axis=1)
grouped_mean_us["salary_to_combined_indexs_ratio"] = my_salary /grouped_mean_us.
 ↪index_means_combined
grouped_mean_us.reset_index(inplace=True)

top_5_cities_for_me = grouped_mean_us.nlargest(5,␣
 ↪"salary_to_combined_indexs_ratio")
top_5_cities_for_me

sns.barplot(data=top_5_cities_for_me, y="salary_to_combined_indexs_ratio",␣
 ↪x="City")
plt.title("Salary to Combined Indexs Ratio for Top 5 U.S Cities")
plt.xlabel("City")
plt.ylabel("Salary to Combined Indexs Ratio")
```

```
plt.show()
```



[401]:
```
## Indexes Combined - International Countries

filtered_countries = col_international[
    (col_international["col_index"] > 55.92) &
    (col_international["restaurant_price_index"] > 55.92) &
    (col_international["local_purchasing_power_index"] > 55.92)]

grouped_mean_int = filtered_countries.groupby("Country")[["col_index",␣
 ↪"restaurant_price_index", "local_purchasing_power_index"]].mean()
grouped_mean_int["index_means_combined"] = grouped_mean_int.mean(axis=1)
grouped_mean_int["salary_to_combined_indexs_ratio"] = my_salary /␣
 ↪grouped_mean_int.index_means_combined
grouped_mean_int.reset_index(inplace=True)

top_5_countries_for_me = grouped_mean_int.nlargest(5,␣
 ↪"salary_to_combined_indexs_ratio")
top_5_countries_for_me
```

```
sns.barplot(data=top_5_countries_for_me, y="salary_to_combined_indexs_ratio",␣
 ↪x="Country")
plt.title("Salary to Combined Indexs Ratio for Top 5 International Countries")
plt.xlabel("Country")
plt.ylabel("Salary to Combined Indexs Ratio")
plt.show()
```