Job Compensation Analysis

Overview

Our group decided to analyze the relationship between job compensation and country/region, looking primarily at the cost of living. Our models explore the relationship between the average job salary of the country/region with that country/region's cost of living (looking for trends, outliers, linear relationships, etc.). Our results showed that lower-cost countries generally have shallower slopes (in relation to salary vs. cost of living), and larger country economies tend to fare steeer slopes. Additionally, we were able to identify outlying countries in the data, which often indicated to unique economic structures within that country (unique job markets/opportunities).

Names

- Sascha Stevens
- Arnav Kamra
- Brian Asami
- Yuhe Tian
- Ahti Hensel

Research Question

What cities around the globe are people getting the most bang for their buck? That is where around the world are people getting apid the most relative to their expenditures? Are there any reason's why some cities have a higher income to expenditures ratio? Are there any regions that consistantly have higher earnings to cost of living ratios?

Background and Prior Work

Exploring the complex relationships between job compensation, Gross Domestic Product (GDP), and the cost of living in different geographical areas reveals an engaging economic narrative. GDP serves as a macroeconomic measure of a nation's total economic activity, offering a broad view of its economic health. At a more granular level, the Consumer Price Index (CPI) provides insights into the average price changes for a set of consumer goods and services, crucial for measuring inflation and making international comparisons. These indicators, along with detailed cost of living analyses like those provided by the Economist Intelligence Unit's (EIU) Cost of Living Index [Ref 3], are foundational in understanding how living expenses align with job compensation in various locales.

Delving into specific job-based analyses adds depth to this exploration. Studies such as the Economic Policy Institute's examination of teacher compensation across countries relative to GDP per capita highlight significant disparities in earnings. For example, teachers in South Korea and Germany earn substantially higher percentages of the per capita GDP compared to their counterparts in the United States [Ref 1]. Similarly, the Hamilton Project at the Brookings Institution's report 'Salary versus Cost of Living: A New Report' sheds light on the variances in earnings for specific job roles across the U.S., demonstrating how cost of living factors can influence the value of salaries in different regions and how companies use location data to balance workforce affordability with quality of life [Ref 2].

However, the accessibility of data from sources like EIU and Mercer, considered industry standards in academia and corporate sectors for their comprehensive and rigorous data, poses a challenge due to their paywalled nature. Our study addresses this by employing Numbeo's Cost of Living Index [Ref 4]. Numbeo offers a free, accessible alternative, sourcing its data through crowdsourcing methods. While this approach may introduce variability and less

precision compared to EIU or Mercer, Numbeo's relative accuracy and extensive data coverage make it a practical choice for our research.

By opting for Numbeo, we navigate the limitations of resource accessibility while maintaining a broad scope in our comparative analysis. This approach enables us to conduct comprehensive research within our constraints, focusing on the critical evaluation of data reliability and the accurate representation of our findings. Understanding these economic indicators and their interplay is crucial for deciphering the broader narrative of job compensation across different geographical and economic landscapes.

Sources:

Teacher Pay around the World: https://urldefense.com/v3/__https://www.epi.org/publication/webfeatures_snapshots_20080402/**B2**B.*5Cn__;4oCL4E-UBO9oY2ogAmEyXEUVG1kmJy7Tq4gCQ$ Salary versus Cost of Living: A New Report: https://urldefense.com/v3/__https://www.esri.com/about/newsroom/publications/wherenext/salary-versus-cost-of-living/**B3**B.*5Cn__;4oCL4oCLJQ!!Mih3wA!B0SPTVuChwGBqCNka_bTd4kYvYUEsO02IR8GqrLgFxf1CvyO4-E-UBO9oY2ogAmEyXEUVG1kmJwMAKji9Q$ Economist Intelligence Unit (EIU): https://urldefense.com/v3/__https://www.investopedia.com/terms/e/economist-intelligence-unit.asp.*5Cn__;JQ!!Mih3wA!B0SPTVuChwGBqCNka_bTd4kYvYUEsO02IR8GqrLgFxf1CvyO4-E-UBO9oY2ogAmEyXEUVG1kmJxQIOanYQ$ Numbeo: https://urldefense.com/v3/__https://www.numbeo.com/common/__;!!Mih3wA!B0SPTVuChwGBqCNka_bTd4kYvYUEsO02E-UBO9oY2ogAmEyXEUVG1kmJyTr8cwdw$

Background and Explanation of Numbeo Data Utilization in Our Project

As part of our project's foundation, we have integrated comprehensive data from Numbeo, a renowned global database for cost of living statistics. Numbeo's approach to data collection and analysis is particularly relevant to our project's goals. Here, I'll provide an overview of how Numbeo's data is structured and processed, which is crucial for understanding the insights we derive for our analysis.

Numbeo's data is centered around cost of living indices, with a specific focus on making relative comparisons to New York City, which serves as a baseline with an index value of 100%. The indices cover various aspects of living costs, from consumer goods prices to rent and local purchasing power. These indices include:

Cost of Living Index (Excluding Rent): Measures the price of consumer goods excluding housing costs. Rent Index: Focuses on rental costs in comparison to New York City. Groceries Index: Estimates grocery prices relative to those in New York. Restaurants Index: Compares dining costs to those in New York City. Cost of Living Plus Rent Index: A comprehensive index covering both consumer goods and rent. Local Purchasing Power Index: Reflects the relative buying power of average net salaries. The methodology behind Numbeo's data collection is a blend of user-generated input and information gathered from authoritative sources. This approach ensures a balance between the ground reality as reported by residents and standardized data from official sources. The data is updated semi-annually, with a higher weighting given to the manually collected information, enhancing its reliability.

In terms of data processing, Numbeo employs a robust system of over 30 filters, both automatic and semi-automatic, to ensure data accuracy and integrity. These filters are designed to identify and correct any anomalies or biases, including the exclusion of entries from identified spam IP addresses. Such stringent filtering is vital to maintaining the objectivity and credibility of the data.

Numbeo’s algorithmic capabilities are an integral part of its data processing. The platform uses advanced algorithms to sift through and validate the data, including a re-evaluation of previously discarded inputs. This ensures that the data is not only current but also accurate and representative of the real-world scenarios.

For our project, we leverage Numbeo's approach to aggregating country data, where city-level inputs are weighted by contributor numbers to form a national average. This method is particularly useful for our analysis, which often requires a holistic view of a country's cost of living situation.

Numbeo's handling of currencies is another aspect we utilize. The platform updates its internal currency exchange rates almost every hour and stores values in multiple currencies. This is particularly beneficial for our project, which often requires converting and comparing costs across different countries.

Finally, Numbeo's approach to incorporating taxes into its data provides us with a more realistic picture of the cost of living. The inclusion of sales taxes and the consideration of post-income tax salary figures allow us to better understand the actual purchasing power in different regions.

In summary, Numbeo's data structure and processing techniques are integral to our project. By understanding the nuances of Numbeo's methodology, we can more effectively interpret and utilize this data to draw meaningful conclusions for our analysis.

Hypothesis

We hypothesize that more developed regions will have a higher earnings to cost of living ratio than less developed countries. We also expect there to be a heavy boost to certain tech cities around the world beacuse of the way the cost of living index measures cost of living.

Datasets

- Cost of Living index
  - Link to the dataset: https://www.kaggle.com/datasets/kkhandekar/cost-of-living-index-by-city-2022
  - Number of cities: 578
  - Number of variables: 8
- Average Salary in USD
  - Link to the dataset: https://www.numbeo.com/cost-of-living/prices_by_city.jsp?displayCurrency=USD&itemId=105
  - Number of observations: 427
  - Number of variables: 3

Data Cleaning

The numbeo datasets are as discussed earlier beneficial from a data cleaning perspective. All of the city names are formatted the same and the cities already include the names of their respective countries. The main difficulty with using the numbeo dataset is that it is difficult to get the Average Salary data formatted in a way that allows pandas to parse through it easily. This is beacause the numbeo data is copy pasted from the chart on the website. There is no download link. These are the steps taken to obtain and clean the Average Salary numbeo data:

Step 1: Go to the website https://www.numbeo.com/cost-of-living/prices_by_city.jsp?displayCurrency=USD&itemId=105 and copy the data from the chart.

Step 2: Paste the data into the IDE of your choosing. The data should come out formatted like this:

1 Zug, Switzerland 7718.50 2 Zurich, Switzerland 7708.67 3 San Francisco, CA, United States 7668.71 4 San Jose, CA, United States 7179.72 5 Basel, Switzerland 7147.45 6 New York, NY, United States 7123.14

Step 3: Paste the data into the following script (replace the example format text). The program will create a csv file for the data. Make sure to enter in a filepath for the script. This is where the CSV will be saved.

```
import pandas as pd # Filepath to save the CSV filepath = '' # Your data string goes here data_str = """ 1 Zug, Switzerland 7718.50 2 Zurich, Switzerland 7708.67 3 San Francisco, CA, United States 7668.71 4 San Jose, CA, United States 7179.72 5 Basel, Switzerland 7147.45 6 New York, NY, United States 7123.14 """ def split_data(s): # Split the string into parts parts = s.split() # The rank is the first element rank = parts[0] # The value is the last element value = parts[-1] # The place is everything in between place = ' '.join(parts[1:-1]) return rank, place, value # Prepare a list to hold all the rows as dictionaries data_list = [] # Split the data into lines lines = data_str.strip().split('\n') # Process each line and add to the list for line in lines: rank, place, value = split_data(line) data_list.append({'Rank': rank, 'Place': place, 'Average Earnings Per Month USD': value}) # Create the DataFrame from the list of dictionaries processed_data = pd.DataFrame(data_list) # Now processed_data contains the desired output processed_data processed_data.to_csv(filepath, index=False)
```

Cleaning the data now that they are in the same database:

In [136…
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from adjustText import adjust_text
import warnings


# Suppress specific FutureWarnings from libraries like Seaborn or Pandas
warnings.simplefilter("ignore", category=FutureWarning)

cost_of_living = pd.read_csv('/Users/saschastevens/Downloads/costofliving.csv')
average_salary = pd.read_csv('/Users/saschastevens/Downloads/Average_Salary.csv')
```

Cleaning Cost of Living and Average Salary Data

In [137…
```
# Preprocessing Cost of Living data
cost_of_living['Rank'] = cost_of_living.index + 1
cost_of_living.rename(columns={'Rank': 'Cost of Living Rank'}, inplace=True)


# Preprocessing Average Salary data
average_salary.rename(columns={'Rank': 'Average Salary Rank'}, inplace=True)
average_salary.rename(columns={'Place': 'City'}, inplace=True)
```

Creating the Merged Dataframe

In [138…
```
# Merging Dataframes on Place column
merged_df = pd.merge(cost_of_living, average_salary, on='City')
merged_df


# Getting the Country from 'City' and making a new Country Column
merged_df['Country'] = merged_df['City'].apply(lambda x: x.split(', ')[-1] if ', ' in x else None)

# Creating the adjusted average salary column
merged_df['Adjust Average Monthly Earnings USD'] = merged_df['Average Earnings Per Month USD'] * me

# Making EU or Not EU Column

# EU Country Link: https://www.gov.uk/eu-eea#:~:text=The%20EU%20countries%20are%3A,%2C%20Slovenia%2
# List of EU member countries as of your current date
eu_countries = ['Austria', 'Belgium', 'Bulgaria', 'Croatia', 'Cyprus', 'Czech Republic', 'Denmark',

# Mapping function
def check_eu_status(country):
    return 'EU' if country in eu_countries else 'Non-EU'

# Assuming your DataFrame has a 'Country' column
merged_df['EU Status'] = merged_df['Country'].apply(check_eu_status)


merged_df.head()
```

Out[138…

	Cost of Living Rank	City	Cost of Living Index	Rent Index	Cost of Living Plus Rent Index	Groceries Index	Restaurant Price Index	Local Purchasing Power Index	Average Salary Rank	Average Earnings Per Month USD	Country
0	1	Hamilton, Bermuda	149.02	96.10	124.22	157.89	155.22	79.43	8	6961.37	Bermuda
1	2	Zurich, Switzerland	131.24	69.26	102.19	136.14	132.52	129.79	2	7708.67	Switzerland
2	3	Basel, Switzerland	130.93	49.38	92.70	137.07	130.95	111.53	5	7147.45	Switzerland
3	4	Zug, Switzerland	128.13	72.12	101.87	132.61	130.93	143.40	1	7718.50	Switzerland
4	5	Lugano, Switzerland	123.99	44.99	86.96	129.17	119.80	111.96	15	5774.50	Switzerland

Initial Plotting and Data Analysis

Let's check out the initial plot of Cost of living vs Average Monthly Earnings in USD. This plot will not generate the greatest measure of cost to earnings but it is a base line and a great place to start.

In [139…

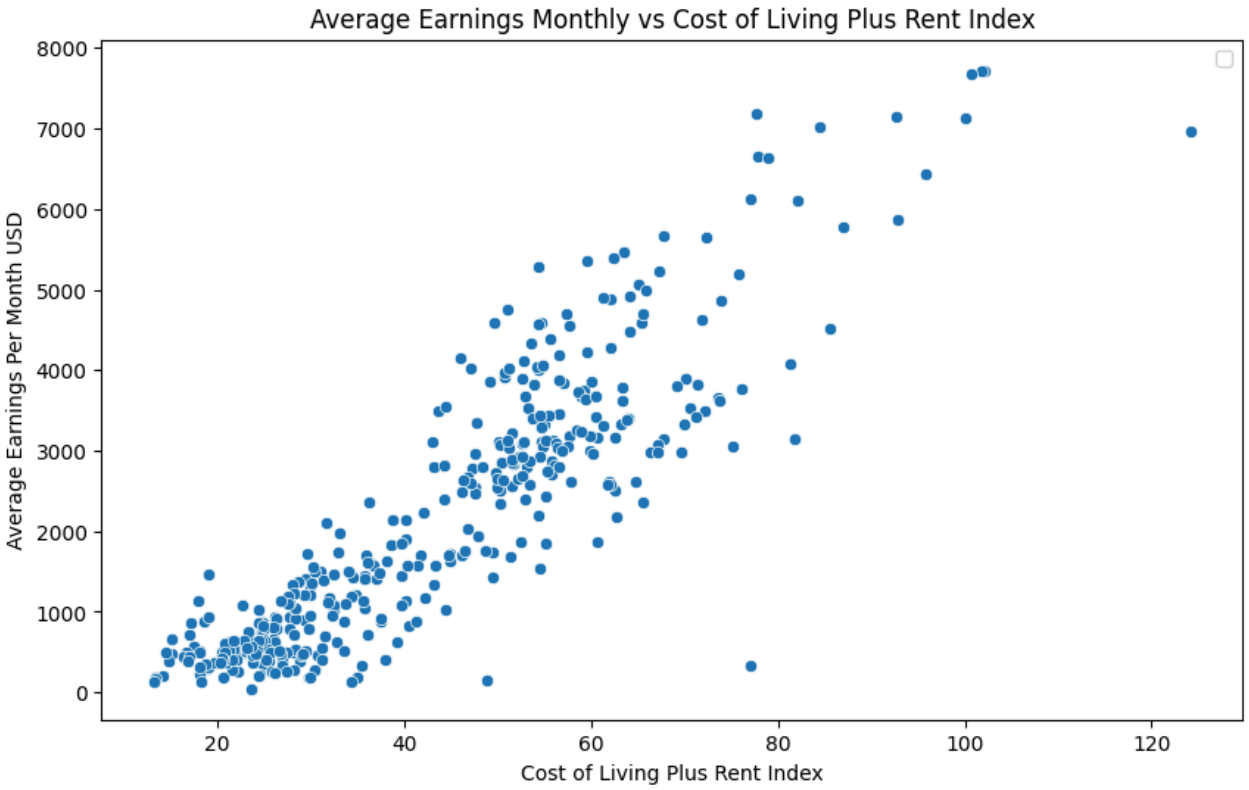
```
# Plotting Average Salary vs Cost of Living
# Create a scatter plot using seaborn for better aesthetics
plt.figure(figsize=(10, 6))
sns.scatterplot(data=merged_df, x='Cost of Living Index', y='Average Earnings Per Month USD')



# Adding title and labels
plt.title('Average Salary vs Cost of Living')
plt.xlabel('Cost of Living Index')
plt.ylabel('Average Earnings Per Month (USD)')

# Show the plot
plt.show()
```

Great. Now that we have a baseline, we should look for a way to further normalize the data. The initial cost of living metric does not include the price of rent in any given city. Let's add that metric into our plot and see how the data looks.

In [140…

```
# Plotting Average Earnings Per Month USD vs Cost of Living Plus Rent Index
plt.figure(figsize=(10, 6))
sns.scatterplot(data=merged_df, x='Cost of Living Plus Rent Index', y='Average Earnings Per Month U

# Adding title and labels
plt.title('Average Earnings Monthly vs Cost of Living Plus Rent Index')
plt.xlabel('Cost of Living Plus Rent Index')
plt.ylabel('Average Earnings Per Month USD')
plt.legend()

# Show the plot
plt.show()
```

```
No artists with labels found to put in legend.  Note that artists whose label start with an undersc
ore are ignored when legend() is called with no argument.
```



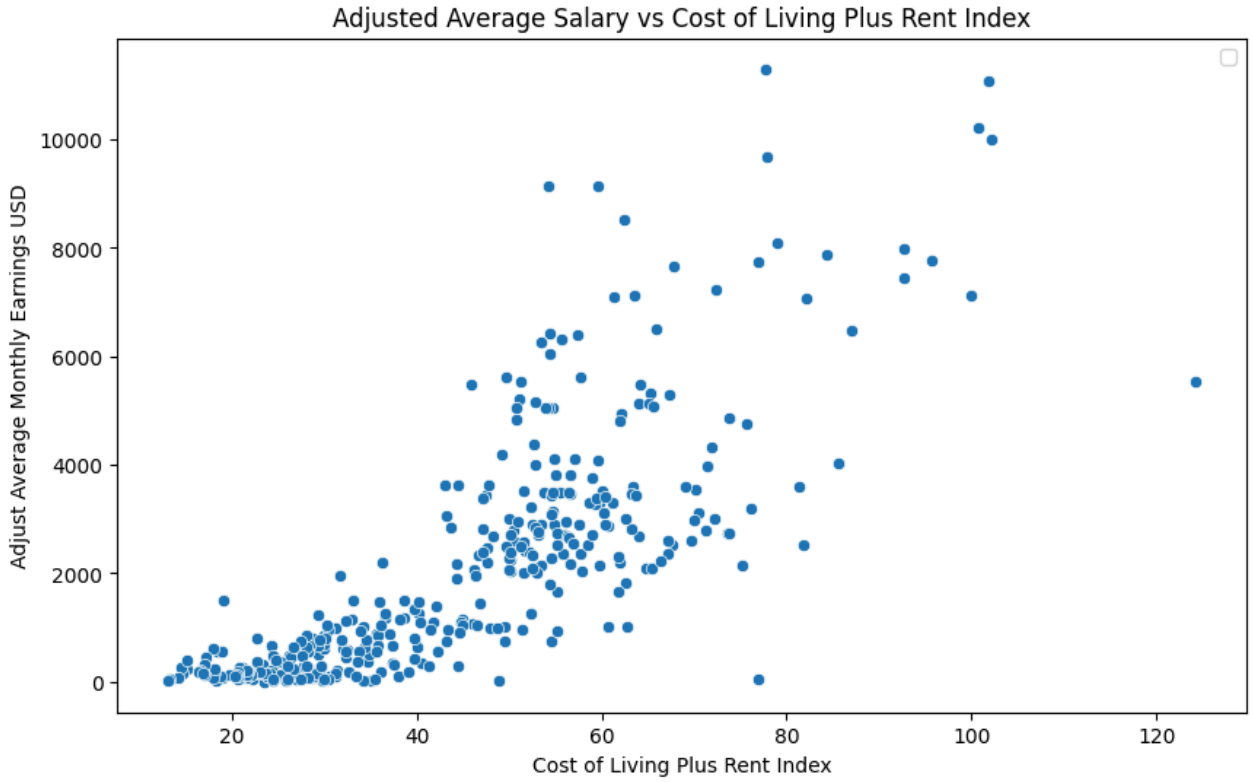
This looks great! But can we make it better? There is a local purchasing power index. Let's use that index to create a purchasing power adjusted monthly earnings by multipling the earnings by the normalize multiplier. This will compare the discretionary income of any given city to that of New York City's normalizing for products consumer plus rent.

Now we will use the adjusted monthly earnings to create a graph for cost of living with and without rent included.

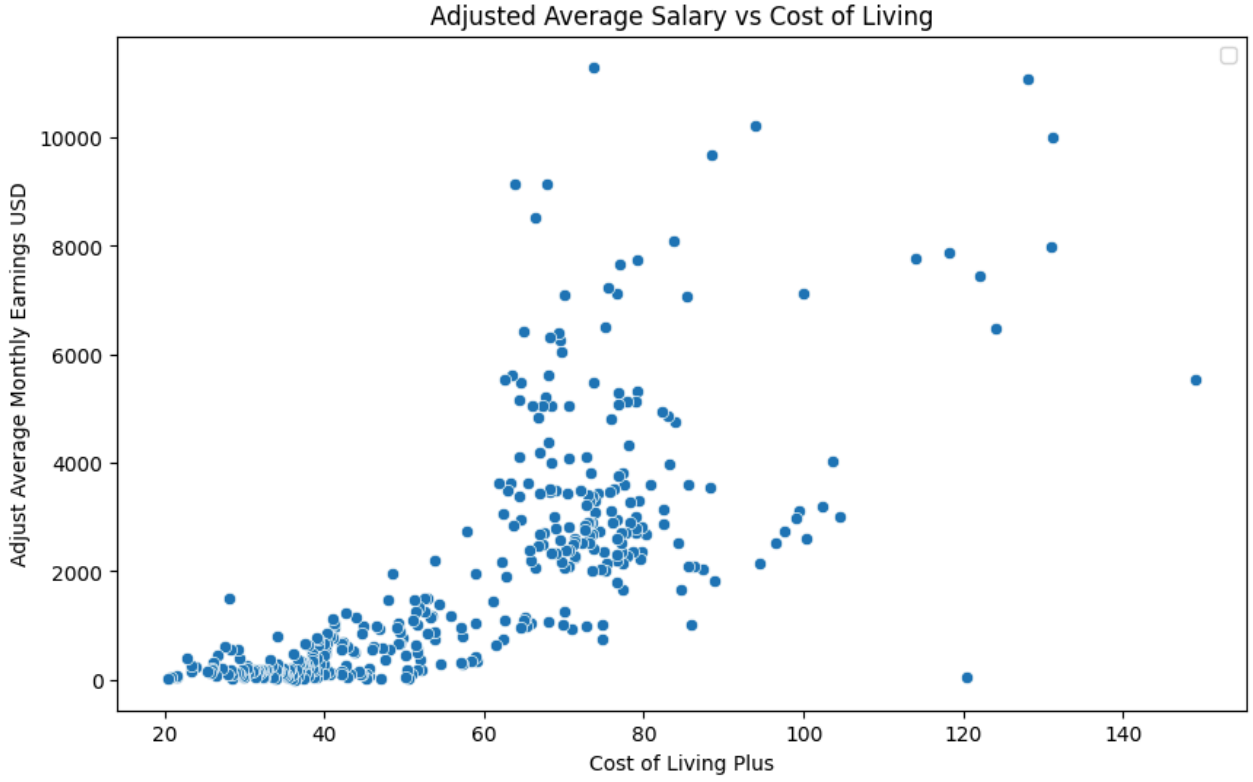
In [141…

```
# Creating Adjusted Purchasing Power
merged_df['Adjust Average Monthly Earnings USD'] = merged_df['Average Earnings Per Month USD'] * me

plt.figure(figsize=(10, 6))
sns.scatterplot(data=merged_df, x='Cost of Living Plus Rent Index', y='Adjust Average Monthly Earni

# Adding title and labels
plt.title('Adjusted Average Salary vs Cost of Living Plus Rent Index')
plt.xlabel('Cost of Living Plus Rent Index')
plt.ylabel('Adjust Average Monthly Earnings USD')
plt.legend()

# Show the plot
plt.show()

plt.figure(figsize=(10, 6))
sns.scatterplot(data=merged_df, x='Cost of Living Index', y='Adjust Average Monthly Earnings USD')

# Adding title and labels
plt.title('Adjusted Average Salary vs Cost of Living')
plt.xlabel('Cost of Living Plus')
plt.ylabel('Adjust Average Monthly Earnings USD')
plt.legend()

# Show the plot
plt.show()
```

```
No artists with labels found to put in legend.  Note that artists whose label start with an undersc
ore are ignored when legend() is called with no argument.
```




```
No artists with labels found to put in legend.  Note that artists whose label start with an undersc
ore are ignored when legend() is called with no argument.
```

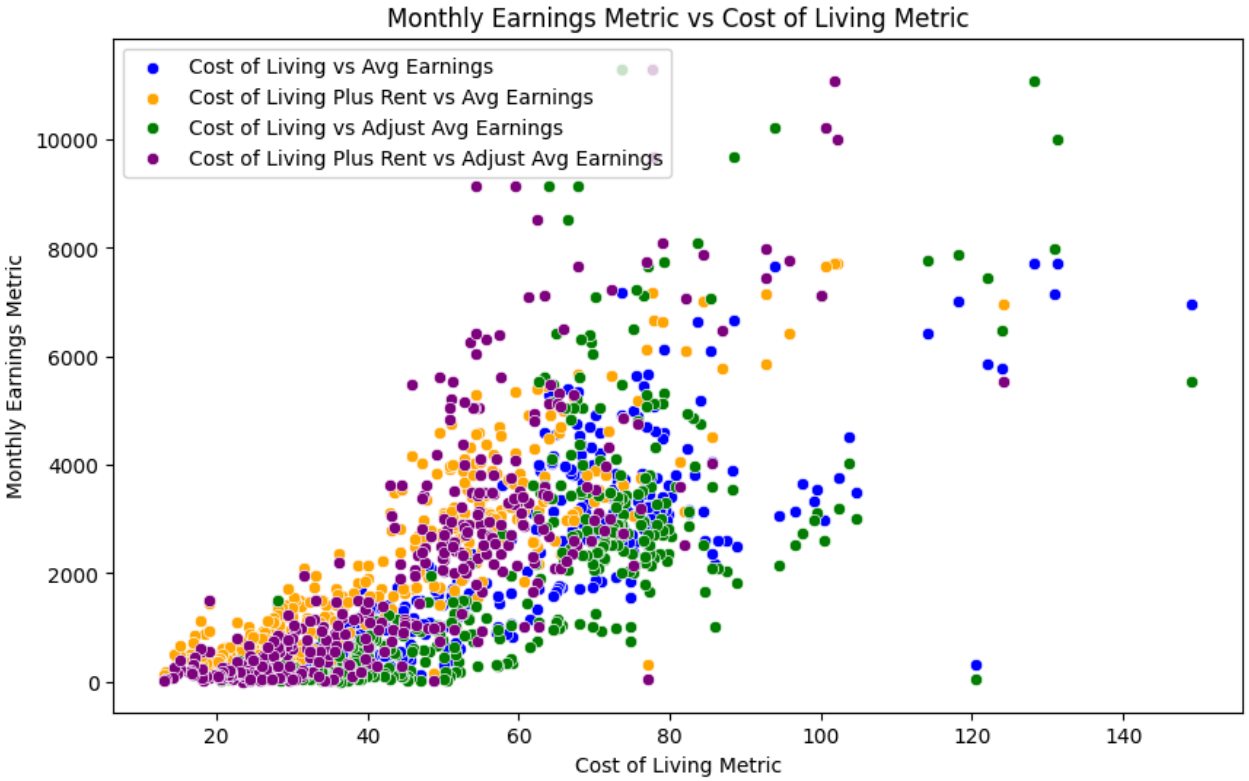


Why don't we include all of these plots together so we can compare the 4 different combinations of earnings and cost of living metric?

In [142…

```
plt.figure(figsize=(10, 6))
sns.scatterplot(data=merged_df, x='Cost of Living Index', y='Average Earnings Per Month USD', color
sns.scatterplot(data=merged_df, x='Cost of Living Plus Rent Index', y='Average Earnings Per Month U
sns.scatterplot(data=merged_df, x='Cost of Living Index', y='Adjust Average Monthly Earnings USD',
```

```
sns.scatterplot(data=merged_df, x='Cost of Living Plus Rent Index', y='Adjust Average Monthly Earni

# Adding title and labels
plt.title('Monthly Earnings Metric vs Cost of Living Metric')
plt.xlabel('Cost of Living Metric')
plt.ylabel('Monthly Earnings Metric')

# Display the legend
plt.legend()

# Show the plot
plt.show()
```



After looking at the combined graphs, we can clearly see that the orange scatter plot is tightest. Why doesn't the adjusted earnings make the plotting look more equitable? Becasue the cost of living metric is created based on the income's of the city's average citizen, when we adjust the income for purchasing power we are skewing the income away from the anticipated income use for the cost of living metric.

Also, as expected, the Cost of Living Index with rent included shows a much clearer trend than the rent excluded index. This is beacuse everyone has to pay rent! So for the rest of our data anlysis we will be using the Average Monthly Earnings and the Rent Included Cost of Living.

Regression Analysis

`Whole Data Regression`

In [143…

```
# Assuming merged_df is your DataFrame
X = merged_df['Cost of Living Plus Rent Index']
Y = merged_df['Average Earnings Per Month USD']

# Calculate the mean of X and Y
mean_X = np.mean(X)
mean_Y = np.mean(Y)

# Calculate the terms needed for the numerator and denominator of beta
merged_df['xycov'] = (merged_df['Cost of Living Plus Rent Index'] - mean_X) * (merged_df['Average E
```

```
merged_df['xvar'] = (merged_df['Cost of Living Plus Rent Index'] - mean_X)**2

# Calculate beta (slope) and alpha (intercept)
beta = merged_df['xycov'].sum() / merged_df['xvar'].sum()
alpha = mean_Y - (beta * mean_X)

# Prepare the regression line
reg_line = 'Y = ' + str(round(alpha, 2)) + ' + ' + str(round(beta, 2)) + 'X'

# Calculate predictions and residuals
merged_df['predictions'] = alpha + beta * merged_df['Cost of Living Plus Rent Index']
merged_df['residuals'] = merged_df['Average Earnings Per Month USD'] - merged_df['predictions']

# Calculate standard deviation of the residuals
std_residuals = np.std(merged_df['residuals'])

# Calculate the number of standard deviations from the regression line
merged_df['Std Deviations'] = (merged_df['Average Earnings Per Month USD'] - merged_df['predictions

# Filter the DataFrame for countries with standard deviations above 2.5 or below -2.5
extreme_countries = merged_df[merged_df['Std Deviations'].abs() > 2.5]

print(f'Whole Data Regression Line: {reg_line}')
print(f'Whole Data Standard Deviation of the Residuals: {std_residuals}')
print('\n')
```

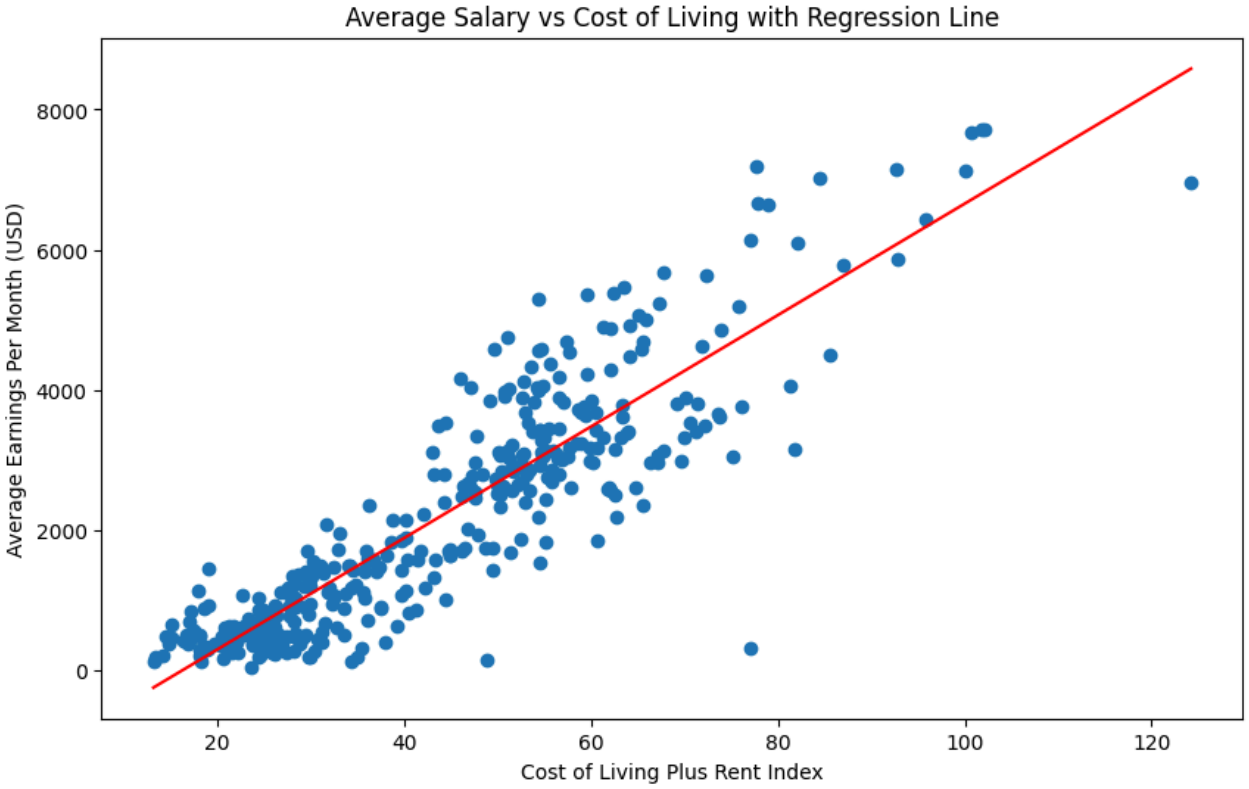
```
Whole Data Regression Line: Y = -1281.38 + 79.4X
Whole Data Standard Deviation of the Residuals: 754.4602998265716
```

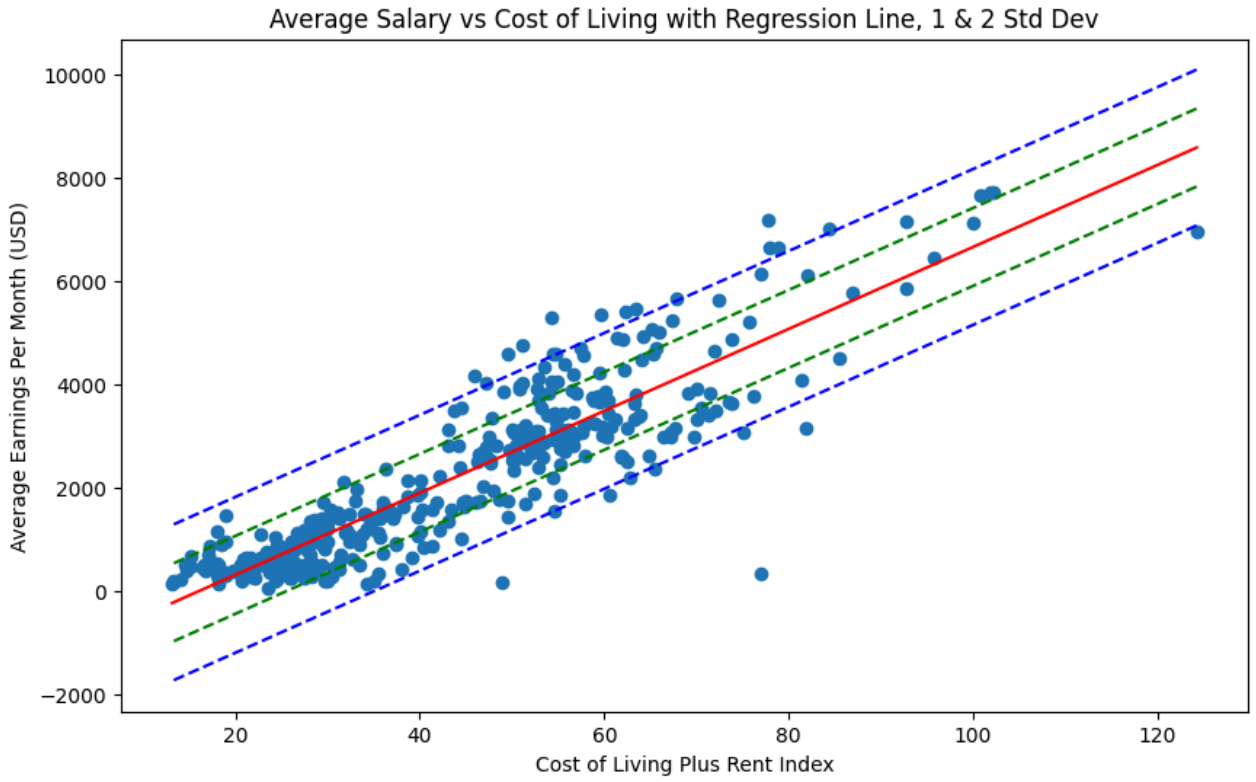
Whole Regression Plotting

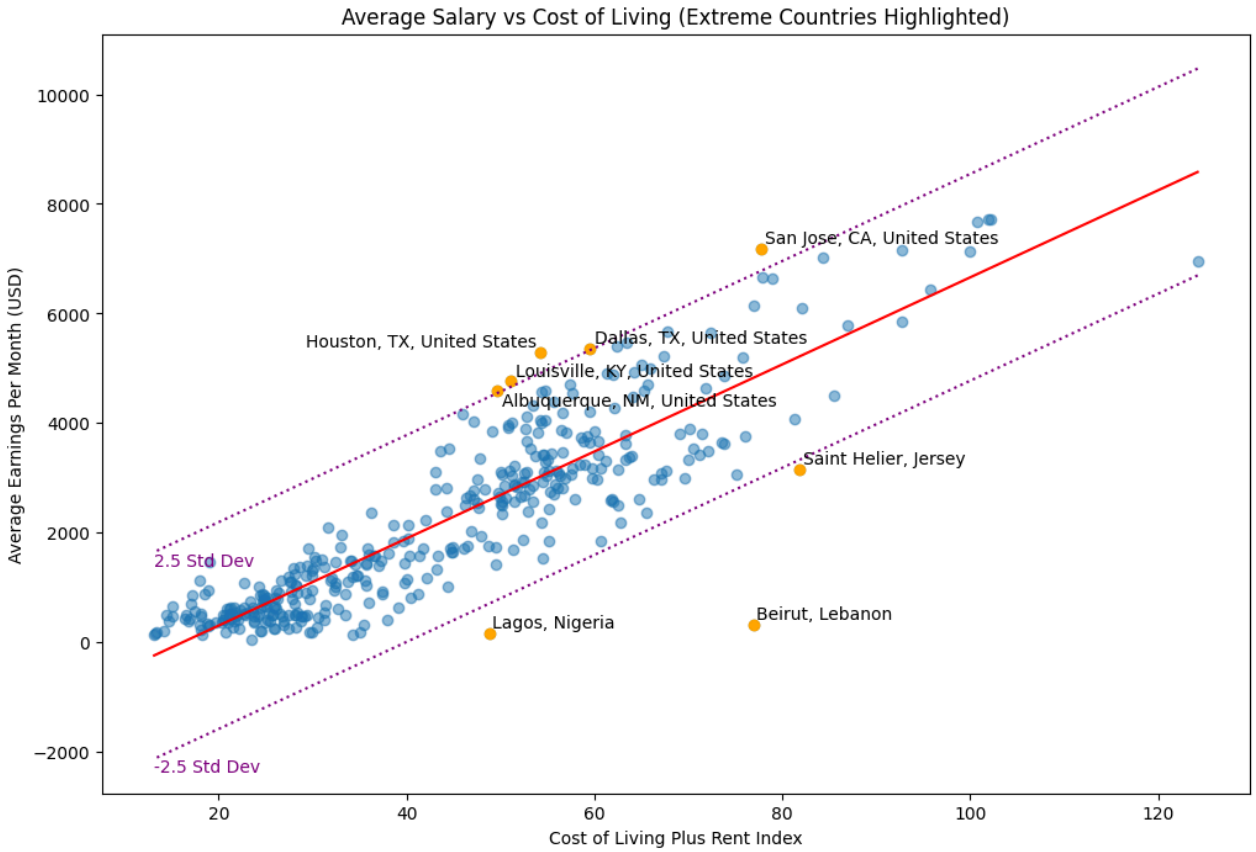
In [144…

```
# Create scatter plot
plt.figure(figsize=(10, 6))
plt.scatter(X, Y) # actual points
plt.plot(X, alpha + beta * X, color='red') # regression line

# Adding title and labels
plt.title('Average Salary vs Cost of Living with Regression Line')
plt.xlabel('Cost of Living Plus Rent Index')
plt.ylabel('Average Earnings Per Month (USD)')
plt.show()

# Plot the graph with lines one and two standard deviations away from the regression line
plt.figure(figsize=(10, 6))
plt.scatter(merged_df['Cost of Living Plus Rent Index'], merged_df['Average Earnings Per Month USD'
plt.plot(merged_df['Cost of Living Plus Rent Index'], merged_df['predictions'], color='red')  # reg

# Plot lines one standard deviation away
plt.plot(merged_df['Cost of Living Plus Rent Index'], merged_df['predictions'] + std_residuals, col
plt.plot(merged_df['Cost of Living Plus Rent Index'], merged_df['predictions'] - std_residuals, col

# Plot lines two standard deviations away
plt.plot(merged_df['Cost of Living Plus Rent Index'], merged_df['predictions'] + 2 * std_residuals,
plt.plot(merged_df['Cost of Living Plus Rent Index'], merged_df['predictions'] - 2 * std_residuals,

# Adding title and labels
plt.title('Average Salary vs Cost of Living with Regression Line, 1 & 2 Std Dev')
plt.xlabel('Cost of Living Plus Rent Index')
plt.ylabel('Average Earnings Per Month (USD)')
plt.show()

##########################################################
# Create scatter plot
plt.figure(figsize=(12, 8))
plt.scatter(merged_df['Cost of Living Plus Rent Index'], merged_df['Average Earnings Per Month USD'
plt.plot(merged_df['Cost of Living Plus Rent Index'], merged_df['predictions'], color='red')  # reg

# Plot lines 2.5 standard deviations away and label them
```

```
line_up = plt.plot(merged_df['Cost of Living Plus Rent Index'], merged_df['predictions'] + 2.5 * st
line_down = plt.plot(merged_df['Cost of Living Plus Rent Index'], merged_df['predictions'] - 2.5 *
plt.text(X.iloc[-1], merged_df['predictions'].iloc[-1] + 2.5 * std_residuals, '2.5 Std Dev', vertic
plt.text(X.iloc[-1], merged_df['predictions'].iloc[-1] - 2.5 * std_residuals, '-2.5 Std Dev', verti

# Plot and store the text objects for adjustment
texts = []
for index, row in extreme_countries.iterrows():
    plt.scatter(row['Cost of Living Plus Rent Index'], row['Average Earnings Per Month USD'], color
    texts.append(plt.text(row['Cost of Living Plus Rent Index'], row['Average Earnings Per Month US

# Adjust the text so they don't overlap
adjust_text(texts)

# Adding title and labels
plt.title('Average Salary vs Cost of Living (Extreme Countries Highlighted)')
plt.xlabel('Cost of Living Plus Rent Index')
plt.ylabel('Average Earnings Per Month (USD)')
plt.show()
```







EU Data Analysis

In [146…

```
# Create a new DataFrame for only EU countries and explicitly make a copy
eu_df = merged_df[merged_df['EU Status'] == 'EU'].copy()

# Perform linear regression
X_eu = eu_df['Cost of Living Plus Rent Index']
```

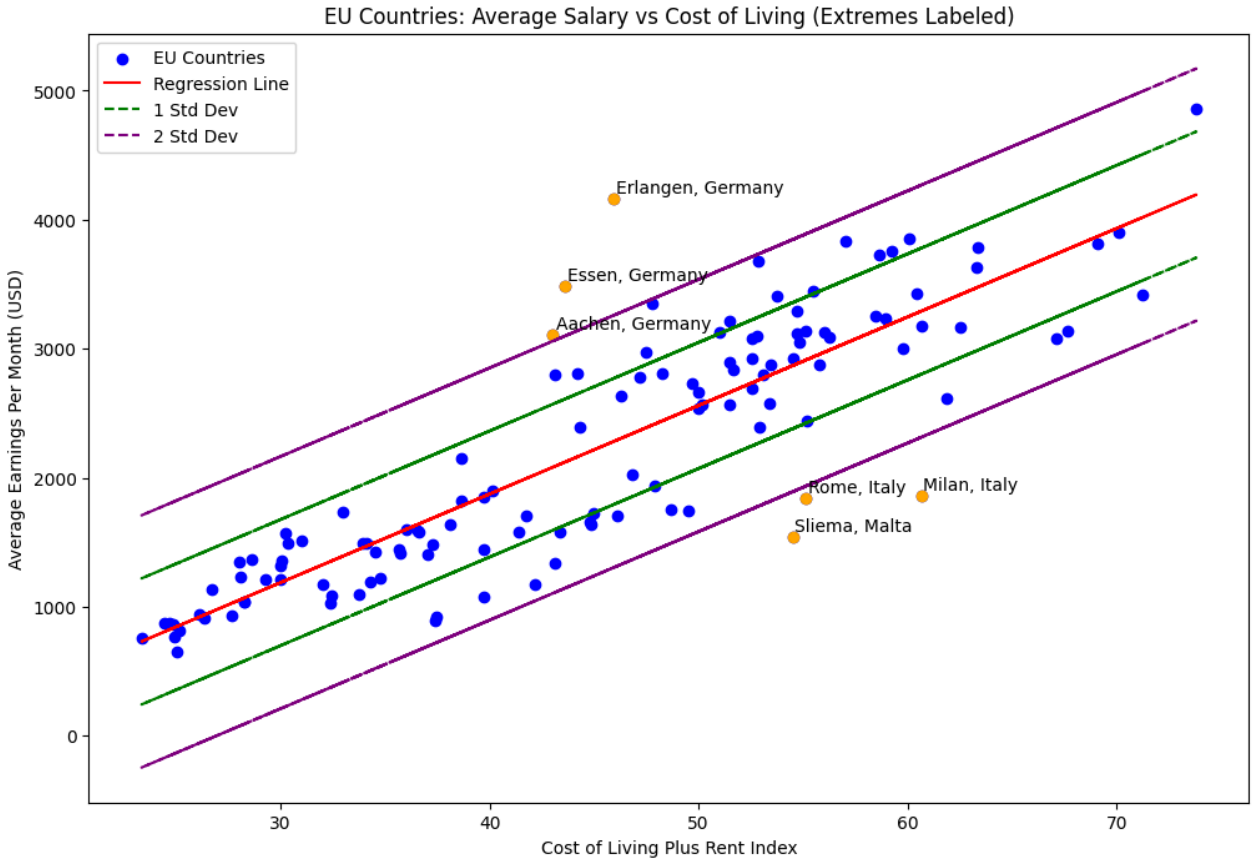
```
Y_eu = eu_df['Average Earnings Per Month USD']
mean_X_eu = X_eu.mean()
mean_Y_eu = Y_eu.mean()

# Use .loc[] for assignments
eu_df.loc[:, 'xycov_eu'] = (X_eu - mean_X_eu) * (Y_eu - mean_Y_eu)
eu_df.loc[:, 'xvar_eu'] = (X_eu - mean_X_eu)**2

# Calculate beta (slope) and alpha (intercept) for EU DataFrame
beta_eu = eu_df['xycov_eu'].sum() / eu_df['xvar_eu'].sum()
alpha_eu = mean_Y_eu - (beta_eu * mean_X_eu)

# More .loc[] for assignments
eu_df.loc[:, 'predictions_eu'] = alpha_eu + beta_eu * X_eu
eu_df.loc[:, 'residuals_eu'] = Y_eu - eu_df['predictions_eu']
std_residuals_eu = eu_df['residuals_eu'].std()

eu_reg_line = 'Y = ' + str(round(alpha_eu, 2)) + ' + ' + str(round(beta_eu, 2)) + 'X'

print(f'EU Data Regression Line: {eu_reg_line}')
print(f'EU Data Standard Deviation of the Residuals: {std_residuals_eu}')
print('\n')
```

```
Whole Data Regression Line: Y = -871.31 + 68.62X
Whole Data Standard Deviation of the Residuals: 488.66814515358976
```

EU Plotting

In [97]:

```
# First, identify EU countries that are outside of 2 standard deviations
extreme_eu = eu_df[eu_df['residuals_eu'].abs() > 2 * std_residuals_eu]

# Create the scatter plot with regression and std deviation lines for EU countries
plt.figure(figsize=(12, 8))
plt.scatter(eu_df['Cost of Living Plus Rent Index'], eu_df['Average Earnings Per Month USD'], color
plt.plot(eu_df['Cost of Living Plus Rent Index'], eu_df['predictions_eu'], color='red', label='Regr
plt.plot(eu_df['Cost of Living Plus Rent Index'], eu_df['predictions_eu'] + 1 * std_residuals_eu, c
plt.plot(eu_df['Cost of Living Plus Rent Index'], eu_df['predictions_eu'] - 1 * std_residuals_eu, c
plt.plot(eu_df['Cost of Living Plus Rent Index'], eu_df['predictions_eu'] + 2 * std_residuals_eu, c
plt.plot(eu_df['Cost of Living Plus Rent Index'], eu_df['predictions_eu'] - 2 * std_residuals_eu, c

# Plot and store the text objects for adjustment
texts = []
for index, row in extreme_eu.iterrows():
    plt.scatter(row['Cost of Living Plus Rent Index'], row['Average Earnings Per Month USD'], color
    texts.append(plt.text(row['Cost of Living Plus Rent Index'], row['Average Earnings Per Month US

# Adjust the text so they don't overlap
adjust_text(texts)

# Adding title, labels, and legend
plt.title('EU Countries: Average Salary vs Cost of Living (Extremes Labeled)')
plt.xlabel('Cost of Living Plus Rent Index')
plt.ylabel('Average Earnings Per Month (USD)')
plt.legend()

# Show the plot
plt.show()
```



USA Data Analysis

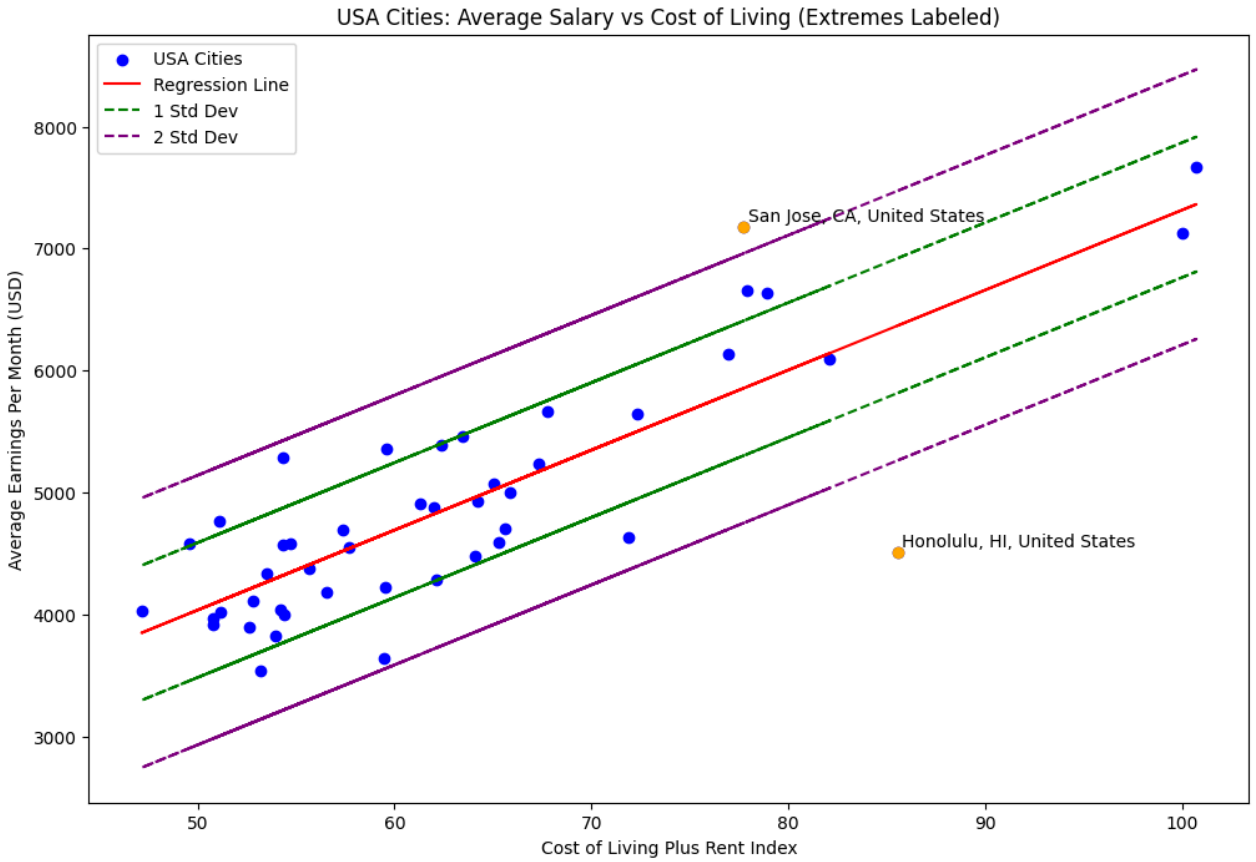
In [147…

```
# Create a DataFrame for USA cities and explicitly make a copy
usa_df = merged_df[merged_df['Country'] == 'United States'].copy()

# Perform linear regression
X_usa = usa_df['Cost of Living Plus Rent Index']
Y_usa = usa_df['Average Earnings Per Month USD']
mean_X_usa = X_usa.mean()
mean_Y_usa = Y_usa.mean()

# Use .loc[] for assignments
usa_df.loc[:, 'xycov_usa'] = (X_usa - mean_X_usa) * (Y_usa - mean_Y_usa)
usa_df.loc[:, 'xvar_usa'] = (X_usa - mean_X_usa)**2

# Calculate beta (slope) and alpha (intercept) for USA DataFrame
beta_usa = usa_df['xycov_usa'].sum() / usa_df['xvar_usa'].sum()
alpha_usa = mean_Y_usa - (beta_usa * mean_X_usa)

# More .loc[] for assignments
usa_df.loc[:, 'predictions_usa'] = alpha_usa + beta_usa * X_usa
usa_df.loc[:, 'residuals_usa'] = Y_usa - usa_df['predictions_usa']
std_residuals_usa = usa_df['residuals_usa'].std()

usa_reg_line = 'Y = ' + str(round(alpha_usa, 2)) + ' + ' + str(round(beta_usa, 2)) + 'X'

print(f'USA Data Regression Line: {eu_reg_line}')
print(f'USA Data Standard Deviation of the Residuals: {std_residuals_usa}')
print('\n')
```

```
USA Data Regression Line: Y = -871.31 + 68.62X
USA Data Standard Deviation of the Residuals: 552.4541127577369
```

USA Plotting

In [99]:
```
# First, identify USA cities that are outside of 2 standard deviations
extreme_usa = usa_df[usa_df['residuals_usa'].abs() > 2 * std_residuals_usa]

# Create the scatter plot with regression and std deviation lines for USA cities
plt.figure(figsize=(12, 8))
plt.scatter(usa_df['Cost of Living Plus Rent Index'], usa_df['Average Earnings Per Month USD'], col
plt.plot(usa_df['Cost of Living Plus Rent Index'], usa_df['predictions_usa'], color='red', label='F

# Plot lines one and two standard deviations away
plt.plot(usa_df['Cost of Living Plus Rent Index'], usa_df['predictions_usa'] + std_residuals_usa, c
plt.plot(usa_df['Cost of Living Plus Rent Index'], usa_df['predictions_usa'] - std_residuals_usa, c
plt.plot(usa_df['Cost of Living Plus Rent Index'], usa_df['predictions_usa'] + 2 * std_residuals_us
plt.plot(usa_df['Cost of Living Plus Rent Index'], usa_df['predictions_usa'] - 2 * std_residuals_us

# Plot and store the text objects for adjustment
texts = []
for index, row in extreme_usa.iterrows():
    plt.scatter(row['Cost of Living Plus Rent Index'], row['Average Earnings Per Month USD'], color
    texts.append(plt.text(row['Cost of Living Plus Rent Index'], row['Average Earnings Per Month US

# Adjust the text so they don't overlap
adjust_text(texts)

# Adding title, labels, and legend
plt.title('USA Cities: Average Salary vs Cost of Living (Extremes Labeled)')
plt.xlabel('Cost of Living Plus Rent Index')
plt.ylabel('Average Earnings Per Month (USD)')
plt.legend()

# Show the plot
plt.show()
```



Lowest 50 Cost of Living Countries Data Analysis

In [148…
```
# Create a DataFrame for the 50 lowest-cost countries
lowest_cost_df = merged_df.nsmallest(50, 'Cost of Living Plus Rent Index')
```

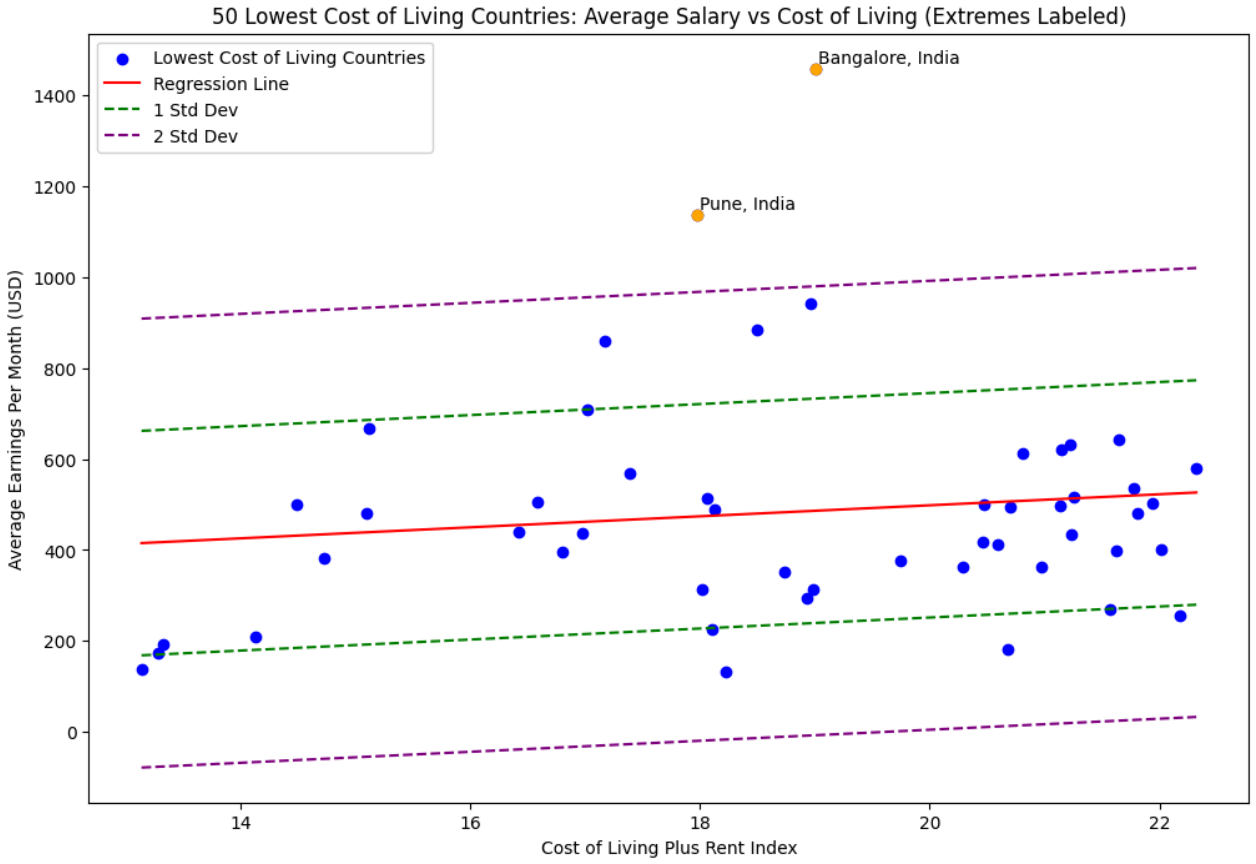
```
# Perform linear regression
# Assuming 'Cost of Living Plus Rent Index' is the independent variable and 'Average Earnings Per M
X_lowest_cost = lowest_cost_df['Cost of Living Plus Rent Index']
Y_lowest_cost = lowest_cost_df['Average Earnings Per Month USD']
mean_X_lowest_cost = X_lowest_cost.mean()
mean_Y_lowest_cost = Y_lowest_cost.mean()

lowest_cost_df['xycov_lowest_cost'] = (X_lowest_cost - mean_X_lowest_cost) * (Y_lowest_cost - mean_
lowest_cost_df['xvar_lowest_cost'] = (X_lowest_cost - mean_X_lowest_cost)**2

# Calculate beta (slope) and alpha (intercept) for the lowest-cost DataFrame
beta_lowest_cost = lowest_cost_df['xycov_lowest_cost'].sum() / lowest_cost_df['xvar_lowest_cost'].s
alpha_lowest_cost = mean_Y_lowest_cost - (beta_lowest_cost * mean_X_lowest_cost)

# Calculate the standard deviation of residuals
lowest_cost_df['predictions_lowest_cost'] = alpha_lowest_cost + beta_lowest_cost * X_lowest_cost
lowest_cost_df['residuals_lowest_cost'] = Y_lowest_cost - lowest_cost_df['predictions_lowest_cost']
std_residuals_lowest_cost = lowest_cost_df['residuals_lowest_cost'].std()

lowest_cost_reg_line = 'Y = ' + str(round(alpha_lowest_cost, 2)) + ' + ' + str(round(beta_lowest_co

print(f'Lowest Cost of Living Countries Data Regression Line: {lowest_cost_reg_line}')
print(f'Lowest Cost of Living Countries Data Standard Deviation of the Residuals: {std_residuals_lo
print('\n')
```

```
Lowest Cost of Living Countries Data Regression Line: Y = 255.85 + 12.12X
Lowest Cost of Living Countries Data Standard Deviation of the Residuals: 246.80234920002366
```

Lowest 50 Cost of Living Countries Plotting

In [101…

```
# First, identify countries in the lowest_cost_df that are outside of 2 standard deviations
extreme_lowest_cost = lowest_cost_df[lowest_cost_df['residuals_lowest_cost'].abs() > 2 * std_residu

# Create the scatter plot with regression and std deviation lines
plt.figure(figsize=(12, 8))
plt.scatter(lowest_cost_df['Cost of Living Plus Rent Index'], lowest_cost_df['Average Earnings Per
plt.plot(lowest_cost_df['Cost of Living Plus Rent Index'], lowest_cost_df['predictions_lowest_cost'

# Plot lines one and two standard deviations away
plt.plot(lowest_cost_df['Cost of Living Plus Rent Index'], lowest_cost_df['predictions_lowest_cost'
plt.plot(lowest_cost_df['Cost of Living Plus Rent Index'], lowest_cost_df['predictions_lowest_cost'
plt.plot(lowest_cost_df['Cost of Living Plus Rent Index'], lowest_cost_df['predictions_lowest_cost'
plt.plot(lowest_cost_df['Cost of Living Plus Rent Index'], lowest_cost_df['predictions_lowest_cost'

# Plot and store the text objects for adjustment
texts = []
for index, row in extreme_lowest_cost.iterrows():
    plt.scatter(row['Cost of Living Plus Rent Index'], row['Average Earnings Per Month USD'], color
    texts.append(plt.text(row['Cost of Living Plus Rent Index'], row['Average Earnings Per Month US

# Adjust the text so they don't overlap
adjust_text(texts)

# Adding title, labels, and legend
plt.title('50 Lowest Cost of Living Countries: Average Salary vs Cost of Living (Extremes Labeled)'
plt.xlabel('Cost of Living Plus Rent Index')
plt.ylabel('Average Earnings Per Month (USD)')
plt.legend(loc='upper left')

# Show the plot
plt.show()
```



Countries of Interest/Developing Nations Data Analysis

In [149…

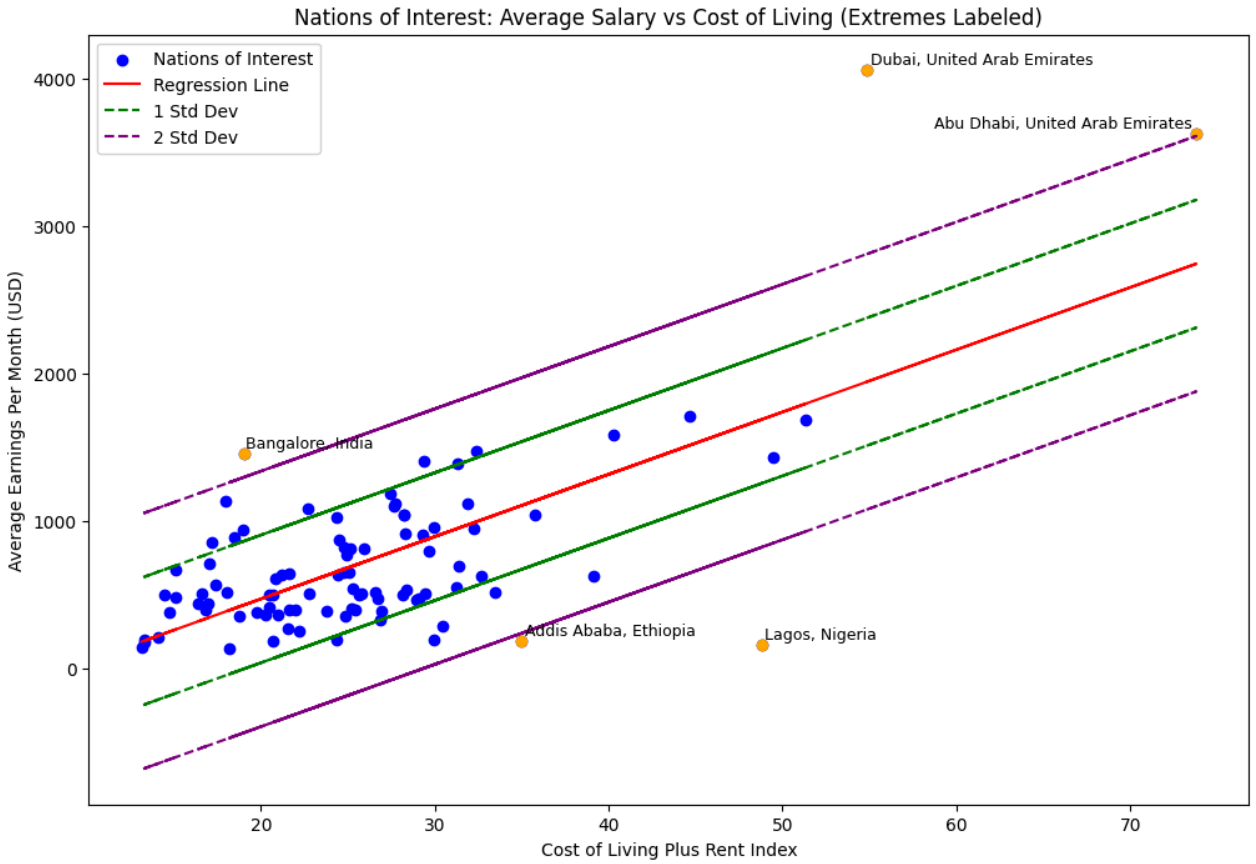
```
# List of countries for the nations of interest
countries = ['China', 'India', 'Indonesia', 'Brazil', 'Mexico', 'Vietnam', 'Turkey', 'South Africa'

# Create a copy of the sliced DataFrame to avoid SettingWithCopyWarning
nations_of_interest_df = merged_df[merged_df['Country'].isin(countries)].copy()

# Perform linear regression calculations using .loc
X_nations_of_interest = nations_of_interest_df['Cost of Living Plus Rent Index']
Y_nations_of_interest = nations_of_interest_df['Average Earnings Per Month USD']
mean_X_nations_of_interest = X_nations_of_interest.mean()
mean_Y_nations_of_interest = Y_nations_of_interest.mean()

nations_of_interest_df.loc[:, 'xycov_nations_of_interest'] = (X_nations_of_interest - mean_X_natior
nations_of_interest_df.loc[:, 'xvar_nations_of_interest'] = (X_nations_of_interest - mean_X_nations

# Calculate beta (slope) and alpha (intercept)
beta_nations_of_interest = nations_of_interest_df['xycov_nations_of_interest'].sum() / nations_of_i
alpha_nations_of_interest = mean_Y_nations_of_interest - (beta_nations_of_interest * mean_X_nations

# More .loc for assignments
nations_of_interest_df.loc[:, 'predictions_nations_of_interest'] = alpha_nations_of_interest + beta
nations_of_interest_df.loc[:, 'residuals_nations_of_interest'] = Y_nations_of_interest - nations_of
std_residuals_nations_of_interest = nations_of_interest_df['residuals_nations_of_interest'].std()

nations_of_interest_reg_line = 'Y = ' + str(round(alpha_nations_of_interest, 2)) + ' + ' + str(rour

print(f'Nations of Interest Data Regression Line: {nations_of_interest_reg_line}')
print(f'Nations of Interest Data Standard Deviation of the Residuals: {std_residuals_nations_of_int
print('\n')
```

```
Nations of Interest Data Regression Line: Y = -374.74 + 42.27X
Nations of Interest Data Standard Deviation of the Residuals: 433.3106039037332
```

Countries of Interest/Developing Nations plotting

In [120…

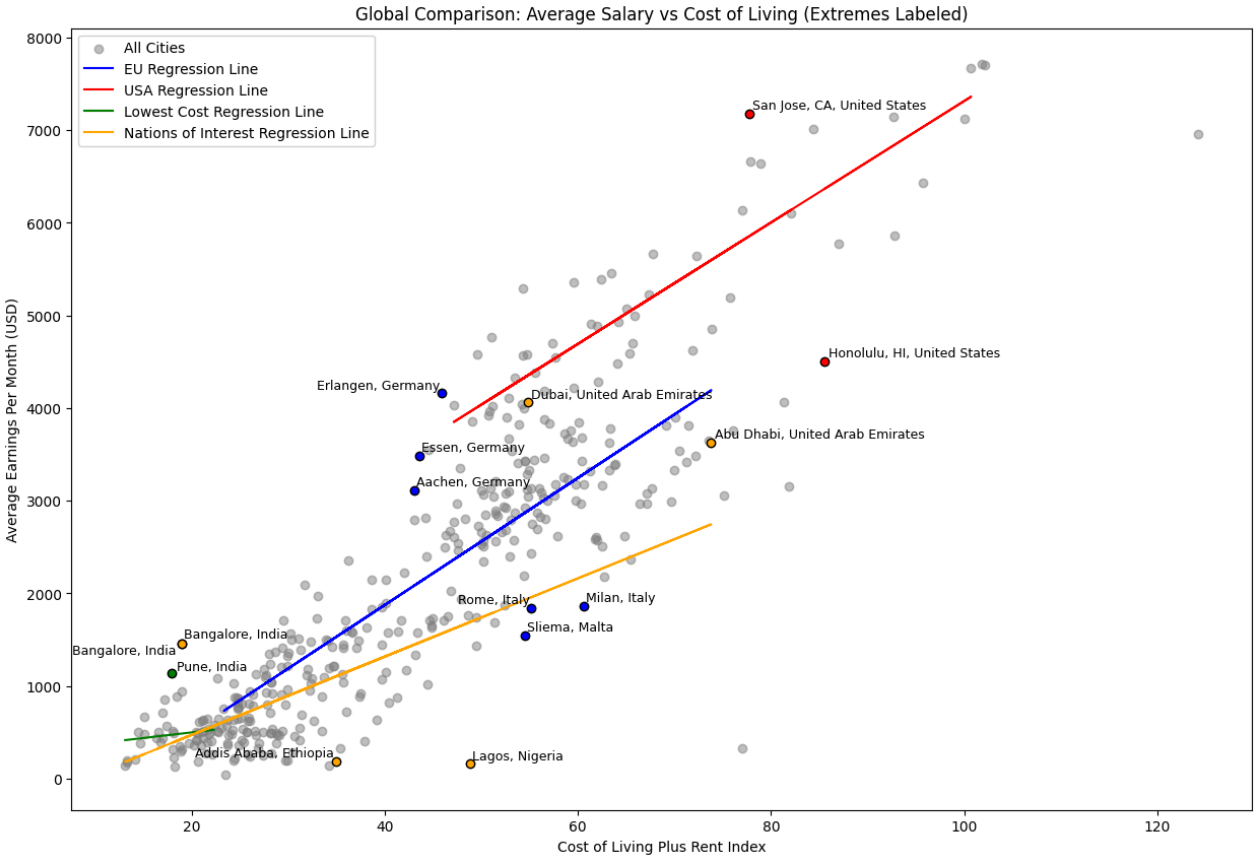
```
# Creating a scatter plot for nations of interest
plt.figure(figsize=(12, 8))
plt.scatter(nations_of_interest_df['Cost of Living Plus Rent Index'], nations_of_interest_df['Avera

# Overlaying the regression line
plt.plot(nations_of_interest_df['Cost of Living Plus Rent Index'], nations_of_interest_df['predicti

# Plot lines one and two standard deviations away
plt.plot(nations_of_interest_df['Cost of Living Plus Rent Index'], nations_of_interest_df['predicti
plt.plot(nations_of_interest_df['Cost of Living Plus Rent Index'], nations_of_interest_df['predicti
plt.plot(nations_of_interest_df['Cost of Living Plus Rent Index'], nations_of_interest_df['predicti
plt.plot(nations_of_interest_df['Cost of Living Plus Rent Index'], nations_of_interest_df['predicti

# Identifying and labeling extreme cities
extreme_nations_of_interest = nations_of_interest_df[nations_of_interest_df['residuals_nations_of_i
texts = []
for index, row in extreme_nations_of_interest.iterrows():
    plt.scatter(row['Cost of Living Plus Rent Index'], row['Average Earnings Per Month USD'], color
    texts.append(plt.text(row['Cost of Living Plus Rent Index'], row['Average Earnings Per Month US

# Adjusting the text to prevent overlap
adjust_text(texts)

# Adding title, labels, and legend
plt.title('Nations of Interest: Average Salary vs Cost of Living (Extremes Labeled)')
plt.xlabel('Cost of Living Plus Rent Index')
plt.ylabel('Average Earnings Per Month (USD)')
plt.legend(loc='upper left')

# Showing the plot
plt.show()
```



Results

In [122…

```
plt.figure(figsize=(15, 10))

# Scatter plot of all cities using 'Cost of Living Plus Rent Index'
plt.scatter(merged_df['Cost of Living Plus Rent Index'], merged_df['Average Earnings Per Month USD'

# Overlaying regression lines for EU, USA, Lowest Cost, and Nations of Interest
plt.plot(eu_df['Cost of Living Plus Rent Index'], eu_df['predictions_eu'], color='blue', label='EU
plt.plot(usa_df['Cost of Living Plus Rent Index'], usa_df['predictions_usa'], color='red', label='U
plt.plot(lowest_cost_df['Cost of Living Plus Rent Index'], lowest_cost_df['predictions_lowest_cost'
plt.plot(nations_of_interest_df['Cost of Living Plus Rent Index'], nations_of_interest_df['predicti

# Highlighting and labeling extreme cities for each category
texts = []
for df, color, label_column in [(extreme_eu, 'blue', 'City'), (extreme_usa, 'red', 'City'), (extrem
    for index, row in df.iterrows():
        plt.scatter(row['Cost of Living Plus Rent Index'], row['Average Earnings Per Month USD'], c
        texts.append(plt.text(row['Cost of Living Plus Rent Index'], row['Average Earnings Per Mont

# Adjusting the text to prevent overlap
adjust_text(texts)

# Adding title, labels, and legend
plt.title('Global Comparison: Average Salary vs Cost of Living (Extremes Labeled)')
plt.xlabel('Cost of Living Plus Rent Index')
plt.ylabel('Average Earnings Per Month (USD)')
plt.legend(loc='upper left')

# Showing the plot
plt.show()
```



From this graph of all the cities, with the regression lines for different groups of interest, we can observe several interesting trends. First, the countries with the lowest cost of living have the regression line with the shallowest slope. This suggests that in these low-cost countries, the cost of living can increase without a significant difference in annual income. Exceptions to this include Bangalore, India, and Pune, India. These cities exhibit higher economic prosperity due to advancements in education and the tech industry. Consequently, the local economy has

experienced a rise in average salary, but the rest of the market has not yet adjusted to this new average. This might be because workers in these areas are spending most of their increased incomes on families that supported them during their education for tech jobs.

The nations of interest represent a collection of fast-growing and promising nations from around the globe. They serve as a metric for a developing economy. As such, we can see that the slope of their regression lies between that of the lowest-cost countries and more developed economies like the United States and the EU. This trend suggests that a steeper slope indicates a more developed economy.

Cities in the United States occupy a unique section of the graph because they are entirely at the top of the cost of living and earnings index. It appears that the United States prioritizes monetary compensation compared to other developed economies, like the EU. Interestingly, outliers from the EU, such as Erlangen, Essen, and Aachen in Germany, show greater compensation per cost of living than many other EU nations. However, this is average or below average for the United States. Many EU countries have more involved governments that cover more costs than the United States government, which is not reflected in the cost of living index. For instance, medical expenses are heavily subsidized in most EU countries but are not considered in the index. This discrepancy widens the gap between cities in the United States and the EU.

An interesting detail lies in the slope of the EU's regression line. The slightly steeper slope compared to that of the United States suggests a small but noticeable difference in compensation for high-expenditure areas. This implies that for EU citizens, working in an expensive area requires a larger increase in compensation relative to the region compared to the United States. This could indicate greater discretionary income, but it's important to note that average compensation is still much higher in America.

Another quirk of the cost of living index is that it doesn't account for the lifestyles of people in these cities. For example, a Hawaiian living in Honolulu might have a high cost of living to income ratio for an American, but they may also lead a lifestyle different from the average American. If they spend their free time engaging in low-cost activities like relaxing on the beach, hiking, or surfing, the cost of living index does not reflect this inexpensive discretionary spending. Conversely, the average tech worker in San Francisco might have a low cost of living to income ratio, but the index doesn't account for their lifestyle. Such a person may frequently attend events, spending more on rideshares and leisure than the average American or Hawaiian. This illustrates that in a competitive, overcast, and cooler place like San Francisco, one might need to spend more to entertain oneself compared to the relaxed, warm environment of Hawaii.

Ethics & Privacy

Biases/Privacy/Terms of Use Issues with the Proposed Data: The data sourced may have inherent biases based on the methodology used to gather compensation figures. For instance, self-reported salaries could be inflated or underreported. It's crucial to ensure that the data does not contain personally identifiable information (PII) about employees. Even if specific names aren't mentioned, combining several pieces of information can lead to de-anonymization. The terms of use for many databases or surveys may restrict certain types of analyses or prohibit the blending of their data with other sources.

Potential Biases in Dataset Composition and Collection: Economic, cultural, and societal differences across countries can influence the reporting of salaries. For instance, in some cultures, discussing one's salary is taboo, which could lead to underrepresentation in voluntary salary surveys. The data may exclude or underrepresent certain job roles, industries, or demographics. Some professions might be unique to certain countries and not others, potentially skewing a direct comparison. The collection method itself could introduce biases, especially if salaries are reported in different currencies and not adequately adjusted for exchange rates or cost of living.

Detecting and Addressing Biases: It's essential to analyze the methodology behind the datasets to understand potential biases and shortcomings. Data should be scrutinized for representativeness, considering various job sectors, demographics, and levels of experience. When presenting findings, acknowledge these limitations and ensure that any comparisons made between countries account for cultural, economic, and societal differences.

Other Data Privacy and Equitable Impact Issues: The comparison might inadvertently perpetuate stereotypes about certain countries or regions, especially if results are not contextualized appropriately. Salary disparities can be influenced by numerous factors, including the cost of living, taxation, benefits, work culture, and more. A mere numerical comparison without this context can be misleading.

Handling Identified Issues: Ensure datasets are anonymized and stripped of PII. If using multiple datasets, consider potential risks of de-anonymization when combining them. Be transparent about the data's sources, its limitations, and the methodology used. This transparency can help address concerns and provide clarity to the audience. Contextualize findings to avoid perpetuating stereotypes or misconceptions. For instance, if there is a noticeable wage disparity between two countries, discuss potential reasons (like cost of living or cultural differences) rather than merely highlighting the gap.

Conclusion & Discussion

Our research question was to find out how jobs in different countries compared to one another in terms of compensation and cost of living. This question is relevant in a globalized economy where job markets and living costs vary widely across different geographic locations.

We started our data analysis by cross checking the cost of living index with the average salaries per month (calculated in USD for consistency) of a given country. This gave us a baseline, which we aimed to further normalize to obtain specific results. We made three additional linear models of the data with slight modifications from the baseline, seeing if adding these hypertuning factors would yield a more linear trend. Our data analysis now consists of the following models:

1. Cost of Living Index vs. Average Earnings
2. Cost of Living Plus Rent vs. Average Earnings
3. Cost of Living vs. Average Earnings (Adjusting based on Purchasing Power)
4. Cost of Living Plus Rent vs. Average Earnings (Adjusted like #3)

Comparing these graphs showed us Model #2 was the most linear, and we therefore decided to proceed with this model for future analyses. We then conducted a regression analysis on the data, showing the line of best fit (also showing the 1&2 STD Lines and Outliers).

Our next step was to analyze the disparity in Cost of Living in specific regions (USA Cities, EU Nations). After breaking down the data by region, we conducted a similar regression analysis for each new dataset. We additionally explored the relationship between the lowest 50 cost of living countries and countries of interest/developing nations.

Our results were interesting and plentiful for each stage of the analysis. They are described in detail in the ‘Results’ section. One of the major discrepancies of our analysis is also described in that section as the following: “Another quirk of the cost of living index is that it doesn't account for the lifestyles of people in these cities. For example, a Hawaiian living in Honolulu might have a high cost of living to income ratio for an American, but they may also lead a lifestyle different from the average American. If they spend their free time engaging in low-cost activities like relaxing on the beach, hiking, or surfing, the cost of living index does not reflect this inexpensive discretionary spending. Conversely, the average tech worker in San Francisco might have a low cost of living to income ratio, but the index doesn't account for their lifestyle. Such a person may frequently attend events, spending more on rideshares and leisure than the average American or Hawaiian. This illustrates that in a competitive, overcast, and cooler place like San Francisco, one might need to spend more to entertain oneself compared to the relaxed, warm environment of Hawaii.” Though not fully comprehensive and clear-cut for every country’s cost of living and job market, the overall results confirmed our hypothesis of more developed regions generally having a higher earnings to cost of living ratio than less developed countries.

Team Expectations

- *Meet at least once per week (virtually or in-person) for at least 30 minutes to touch base on our tasks.*
- *Make sure to inform the team if you are unable to finish your designated task.*
- *Always be respectful and communicative with the team.*

Project Timeline Proposal

Meeting Date	Meeting Time	Completed Before Meeting	Discuss at Meeting
10/29	1 PM	Read & Think about COGS 108 expectations; brainstorm topics/questions	Determine best form of communication; Discuss and decide on final project topic; discuss hypothesis; begin background research
10/30	10 AM	Do background research on topic	Discuss ideal dataset(s) and ethics; draft project proposal
11/1	10 AM	Edit, finalize, and submit proposal; Search for datasets	Discuss Wrangling and possible analytical approaches; Assign group members to lead each specific part
11/14	6 PM	Import & Wrangle Data; EDA	Review/Edit wrangling/EDA; Discuss Analysis Plan
11/29	12 PM	Finalize wrangling/EDA; Begin Analysis	Discuss/edit Analysis; Complete project check-in
12/10	12 PM	Complete analysis; Draft results/conclusion/discussion	Discuss/edit full project
12/12	Before 11:59 PM	NA	Turn in Final Project & Group Project Surveys