

# Weighted Average Calculation

```
basic_needs = ['Housing', 'Healthcare', 'Safety',
               'Environmental Quality']
economic_opportunity = ['Economy', 'Startups', 'Venture
                        Capital', 'Business Freedom', 'Taxation']
mobility_infrastructure = ['Travel Connectivity', 'Commute',
                           'Internet Access']
quality_of_life = ['Leisure & Culture', 'Education', 'Tolerance',
                  'Outdoors', 'Cost of Living']

dataset_minmax['Basic_Needs_Index'] =
dataset_minmax[basic_needs].mean(axis=1)
dataset_minmax['Economic_Opportunity_Index'] =
dataset_minmax[economic_opportunity].mean(axis=1)
dataset_minmax['Mobility_Infrastructure_Index'] =
dataset_minmax[mobility_infrastructure].mean(axis=1)
dataset_minmax['Quality_of_Life_Index'] =
dataset_minmax[quality_of_life].mean(axis=1)

sub_indices = ['Basic_Needs_Index',
               'Economic_Opportunity_Index',
               'Mobility_Infrastructure_Index',
               'Quality_of_Life_Index']
dataset_minmax['Composite_Index'] =
dataset_minmax[sub_indices].mean(axis=1)

print("Summary of Sub-indices and Composite Index:")

# Adding custom weights to each sub-index
weights_basic = {'Housing': 0.2, 'Healthcare': 0.4, 'Safety':
0.2, 'Environmental Quality': 0.2}
weights_econ = {'Economy': 0.3, 'Startups': 0.2, 'Venture
Capital': 0.1, 'Business Freedom': 0.25, 'Taxation': 0.15}
weights_mob = {'Travel Connectivity': 0.4, 'Commute': 0.3,
'Internet Access': 0.3}
weights_qol = {'Leisure & Culture': 0.25, 'Education': 0.25,
'Tolerance': 0.2, 'Outdoors': 0.15, 'Cost of Living': 0.15}

can you help with a helper function to calculate weighted
average for a row with each of the weights
```

Sure! You can create a helper function to calculate the weighted average for each row based on the weights provided for each sub-index. Here's a function that calculates the weighted average for a given set of columns and their respective weights:

python

```
import pandas as pd

def calculate_weighted_average(row, weights):
    """
    Calculate the weighted average for a given row using the provided weights.

    :param row: The row in the DataFrame (Series object).
    :param weights: A dictionary where the keys are column names, and values are the
    weights for those columns.
    :return: Weighted average of the specified columns.
    """
    weighted_sum = 0
    total_weight = sum(weights.values())

    for column, weight in weights.items():
        weighted_sum += row[column] * weight

    return weighted_sum / total_weight

# Example of how to use it with your dataset
dataset_minmax['Weighted_Basic_Needs'] =
dataset_minmax.apply(calculate_weighted_average,
                      axis=1,
                      weights=weights_basic)

dataset_minmax['Weighted_Economic_Opportunity'] =
dataset_minmax.apply(calculate_weighted_average,
                      axis=1,
                      weights=weights_econ)

dataset_minmax['Weighted_Mobility_Infrastructure'] =
dataset_minmax.apply(calculate_weighted_average,
                      axis=1,
                      weights=weights_mob)

dataset_minmax['Weighted_Quality_of_Life'] =
dataset_minmax.apply(calculate_weighted_average,
```

axis=1,

```
weights=weights_qol)

# You can calculate the overall weighted composite index based on the new weighted
sub-indices
dataset_minmax['Weighted_Composite_Index'] = dataset_minmax[['Weighted_Basic_Needs',

'Weighted_Economic_Opportunity',

'Weighted_Mobility_Infrastructure',

'Weighted_Quality_of_Life']].mean(axis=1)

print("Summary of Weighted Sub-indices and Composite Index:")
print(dataset_minmax[['Weighted_Basic_Needs',
                        'Weighted_Economic_Opportunity',
                        'Weighted_Mobility_Infrastructure',
                        'Weighted_Quality_of_Life',
                        'Weighted_Composite_Index']].describe())
```

## Explanation:

- The `calculate_weighted_average` function takes a row from the DataFrame and the corresponding weight dictionary.
- For each column, it multiplies the value by the weight and sums the results.
- The sum is then divided by the total weight to give the weighted average for that row.

You can call `apply` on the DataFrame to apply this function to each row, calculating the weighted average for each sub-index and then for the composite index.

Let me know if you need further adjustments!

```
live_df = pd.read_csv('movingtoGlobalLiveabilityIndex.csv')

live_df['City'] = live_df['City'].str.strip().str.lower()
live_df['Country'] = live_df['Country'].str.strip().str.lower()

ua_df = dataset[['UA_Name', 'UA_Country']].copy()
ua_df.columns = ['City', 'Country']
ua_df['City'] = ua_df['City'].str.strip().str.lower()
ua_df['Country'] = ua_df['Country'].str.strip().str.lower()
```

```

ua_df = pd.concat([ua_df,
dataset_minmax[['Basic_Needs_Index',
'Economic_Opportunity_Index',
'Mobility_Infrastructure_Index',
'Quality_of_Life_Index',
'Composite_Index']]], axis=1)

merged = ua_df.merge(live_df,
on=['City','Country'],
how='inner',
suffixes=('_UA','_Live'))

print(f"Rows matched: {len(merged)} / {len(ua_df)} UA
cities")

how can i calculate the correlation between the 2 data?

```

To calculate the correlation between the two datasets ( `ua_df` and `live_df` ), specifically the columns from each dataset that you've merged, you can use the `corr()` function provided by pandas. This function computes the Pearson correlation coefficient for numerical columns.

Here's how you can do it for the columns that are present in both datasets:

1. After merging the data, you can drop any non-numeric columns (if needed) or just focus on the ones you're interested in.
2. Then, calculate the correlation between the relevant columns using `merged.corr()`.

Here's an example of how you can do it:

python

```

# Calculate correlation between all numeric columns in the merged DataFrame
correlation_matrix = merged.corr()

```

```

# Display the correlation matrix
print(correlation_matrix)

```

```

# If you want to focus on specific columns (e.g., the indices)
columns_of_interest = ['Basic_Needs_Index', 'Economic_Opportunity_Index',
'Mobility_Infrastructure_Index', 'Quality_of_Life_Index',
'Composite_Index']

```

```

# Calculate the correlation between the 'UA' indices and the 'Live' indices
live_columns = [col for col in merged.columns if col.endswith('_Live')] # Columns
from the 'Live' dataset
relevant_columns = columns_of_interest + live_columns

# Subset the merged DataFrame to the relevant columns
relevant_data = merged[relevant_columns]

# Calculate correlation for the relevant columns
relevant_correlation = relevant_data.corr()

# Display the correlation of interest
print(relevant_correlation)

```

## Explanation:

1. `merged.corr()` computes the correlation between all numerical columns in the merged DataFrame.
2. `columns_of_interest` are the indices from `dataset_minmax` you're interested in, while `live_columns` are the corresponding columns from the `live_df`.
3. `relevant_data` is the subset of the merged DataFrame with the relevant columns, and `relevant_data.corr()` calculates the correlation between them.

This will give you a matrix showing the correlation between each pair of variables. Correlation values closer to 1 or -1 indicate strong positive or negative relationships, respectively, while values near 0 indicate weak or no linear relationship.

Rows matched: 82 / 266 UA cities

Pearson r between Composite\_Index and Global Overall Score: 0.761

Correlation matrix (UA sub-indices vs Global sub-scores):

	Stability	Healthcare	Culture & Environment \
Basic_Needs_Index	0.671	0.304	0.139
Economic_Opportunity_Index	0.571	0.522	0.369
Mobility_Infrastructure_Index	0.528	0.488	0.384
Quality_of_Life_Index	0.392	0.412	0.431

	Education	Infrastructure
Basic_Needs_Index	0.305	0.320
Economic_Opportunity_Index	0.472	0.224
Mobility_Infrastructure_Index	0.144	0.564
Quality_of_Life_Index	0.268	0.076

can you help with the documentation of this? if you can put the findings in a table format for markdown

Sure! Below is the documentation of your correlation findings, structured in a table format for markdown:

## Documentation of Correlation Results

### Pearson Correlation Between Composite Index and Global Overall Score:

- **Value:** 0.761
- **Interpretation:** The Pearson correlation coefficient between the Composite Index and the Global Overall Score is quite strong (positive correlation), indicating a moderately high positive relationship between the two variables.

### Correlation Matrix (UA Sub-indices vs. Global Sub-scores):

Sub-index	Stability	Healthcare	Culture & Environment	Education	Infrastru
Basic_Needs_Index	0.671	0.304	0.139	0.305	0.320
Economic_Opportunity_Index	0.571	0.522	0.369	0.472	0.224
Mobility_Infrastructure_Index	0.528	0.488	0.384	0.144	0.564
Quality_of_Life_Index	0.392	0.412	0.431	0.268	0.076

## Key Insights:

- **Basic Needs** shows a relatively strong correlation with **Stability** (0.671), indicating that cities with better basic needs tend to have higher stability scores.
- **Economic Opportunity** has moderate correlations with both **Healthcare** (0.522) and **Education** (0.472), suggesting that cities with better economic opportunities tend to have better healthcare and education systems.

- **Mobility Infrastructure** is most strongly correlated with **Infrastructure** (0.564), indicating that cities with better mobility infrastructure also tend to have better overall infrastructure.
- **Quality of Life** has the strongest correlation with **Culture & Environment** (0.431) and **Healthcare** (0.412), reflecting a relationship between quality of life and cultural/environmental factors, as well as healthcare quality.

This table and summary can be used for further analysis and to guide decision-making based on the correlations observed between the sub-indices and global scores.

Let me know if you'd like to make any adjustments!