**Notes for Q3.**

The relationship between mood, location, and weather conditions.

**Key Steps in the Code**

 **Data Loading and Cleaning:**

* Data is read from a CSV file and cleaned to ensure valid date formats and numerical columns.
* Rows with missing data are dropped.

 **Mapping Mood to Numerical Values:**

* Moods are converted to numerical values for easier analysis (Tired=1, Neutral=2, Happy=3).

 **Analysis Logic:**

* Data is grouped by weather\_conditions and location.
* For each combination of weather and location, the proportion of moods is calculated and visualized using stacked bar charts.

 **Visualization:**

* Subplots are used to display mood distributions for each location under different weather conditions.
* An overall mood distribution by location is also created.

Improved Version of the Code (Snippet)

|  |
| --- |
| # Mapping mood to numerical values with error handling  mood\_mapping = {'Tired': 1, 'Neutral': 2, 'Happy': 3}  df['mood\_numerical'] = df['mood'].map(mood\_mapping)  if df['mood\_numerical'].isna().any():  print("Warning: Some rows have unmapped mood values. Dropping these rows.")  df.dropna(subset=['mood\_numerical'], inplace=True)  # Dynamic mood labels  mood\_labels = df['mood'].unique()  # Create cross-tabulations and visualize with stacked bar charts  unique\_weather = df['weather\_conditions'].unique()  unique\_locations = df['location'].unique()  # Loop through weather conditions  for weather in unique\_weather:  df\_weather = df[df['weather\_conditions'] == weather]    fig, axes = plt.subplots(1, len(unique\_locations), figsize=(6\*len(unique\_locations), 5), squeeze=False)  fig.suptitle(f"Mood Distribution by Location (Weather: {weather})", fontsize=16)    for i, location in enumerate(unique\_locations):  df\_location\_weather = df\_weather[df\_weather['location'] == location]  if not df\_location\_weather.empty:  mood\_counts = df\_location\_weather['mood'].value\_counts(normalize=True) # Normalize to proportions    # Color coding for moods  colors = ['red', 'orange', 'green'][:len(mood\_counts)]  axes[0, i].bar(mood\_counts.index, mood\_counts.values, color=colors)    axes[0, i].set\_title(f"Location: {location}")  axes[0, i].set\_ylabel("Proportion of Mood")  axes[0, i].set\_xlabel("Mood")  else:  axes[0, i].axis("off")    plt.tight\_layout()  plt.subplots\_adjust(top=0.85)  plt.show()  # Overall mood distribution by location  fig, axes = plt.subplots(1, len(unique\_locations), figsize=(6\*len(unique\_locations), 5), squeeze=False)  fig.suptitle(f"Overall Mood Distribution by Location", fontsize=16)  for i, location in enumerate(unique\_locations):  df\_location = df[df['location'] == location]  if not df\_location.empty:  mood\_counts = df\_location['mood'].value\_counts(normalize=True)    colors = ['red', 'orange', 'green'][:len(mood\_counts)]  axes[0, i].bar(mood\_counts.index, mood\_counts.values, color=colors)    axes[0, i].set\_title(f"Location: {location}")  axes[0, i].set\_ylabel("Proportion of Mood")  axes[0, i].set\_xlabel("Mood")  else:  axes[0, i].axis("off")  plt.tight\_layout()  plt.subplots\_adjust(top=0.85)  plt.show() |

 **Objective:**

* Analyze mood distribution across locations and weather conditions.

 **Data Preparation:**

* Mapped moods to numerical values for easier analysis.
* Filtered rows with invalid or missing mood data.

 **Analysis Process:**

* Explored mood proportions for each combination of location and weather.
* Created separate visualizations for detailed insights.

 **Findings:**

* Highlight mood patterns under specific weather conditions and locations.
* Discuss notable trends (e.g., happier moods in certain locations or weather conditions).

 **Improvements and Next Steps:**

* Add more dynamic visualizations, such as grouped bar charts for combined analysis.
* Explore correlations between mood and other factors like sleep or activity levels.