```
import pandas as pd
man=pd.read_csv("/content/weatherHistory.csv (3) (1) (1).zip")
man
```

	Formatted Date	Summary	Precip Type	Temperature (C)	Apparent Temperature (C)	Humidity	Wind Speed (km/h)	Wind Bearing (degrees)	Visibility (km)	Loud Cover	(
0	2006-04-01 00:00:00.000 +0200	Partly Cloudy	rain	9.472222	7.388889	0.89	14.1197	251.0	15.8263	0.0	
1	2006-04-01 01:00:00.000 +0200	Partly Cloudy	rain	9.355556	7.227778	0.86	14.2646	259.0	15.8263	0.0	
2	2006-04-01 02:00:00.000 +0200	Mostly Cloudy	rain	9.377778	9.377778	0.89	3.9284	204.0	14.9569	0.0	
3	2006-04-01 03:00:00.000 +0200	Partly Cloudy	rain	8.288889	5.944444	0.83	14.1036	269.0	15.8263	0.0	
4	2006-04-01 04:00:00.000 +0200	Mostly Cloudy	rain	8.755556	6.977778	0.83	11.0446	259.0	15.8263	0.0	
...	...	...	...	...	...	...	...	...	...	...	...
96448	2016-09-09 19:00:00.000 +0200	Partly Cloudy	rain	26.016667	26.016667	0.43	10.9963	31.0	16.1000	0.0	
96449	2016-09-09 20:00:00.000 +0200	Partly Cloudy	rain	24.583333	24.583333	0.48	10.0947	20.0	15.5526	0.0	
96450	2016-09-09 21:00:00.000 +0200	Partly Cloudy	rain	22.038889	22.038889	0.56	8.9838	30.0	16.1000	0.0	
96451	2016-09-09 22:00:00.000 +0200	Partly Cloudy	rain	21.522222	21.522222	0.60	10.5294	20.0	16.1000	0.0	
96452	2016-09-09 23:00:00.000 +0200	Partly Cloudy	rain	20.438889	20.438889	0.61	5.8765	39.0	15.5204	0.0	

96453 rows × 12 columns

Next steps: Generate code with man | New interactive sheet

```
print(man.isnull().sum())
```

```
Formatted Date               0
Summary                      0
Precip Type                517
Temperature (C)              0
Apparent Temperature (C)     0
Humidity                     0
Wind Speed (km/h)            0
Wind Bearing (degrees)       0
Visibility (km)              0
Loud Cover                   0
Pressure (millibars)         0
Daily Summary                0
dtype: int64
```

```
# Display the first few rows of the dataframe
display(man.head())

# Get information about the dataframe (data types, non-null va
display(man.info())

# Get descriptive statistics of the numerical columns
display(man.describe())

# Get the shape of the dataframe (number of rows and columns)
print(f"Shape of the dataframe: {man.shape}")
```

	Formatted Date	Summary	Precip Type	Temperature (C)	Apparent Temperature (C)	Humidity	Wind Speed (km/h)	Wind Bearing (degrees)	Visibility (km)	Loud Cover	Pr (mill
0	2006-04-01 00:00:00.000 +0200	Partly Cloudy	rain	9.472222	7.388889	0.89	14.1197	251.0	15.8263	0.0	
1	2006-04-01 01:00:00.000 +0200	Partly Cloudy	rain	9.355556	7.227778	0.86	14.2646	259.0	15.8263	0.0	
2	2006-04-01 02:00:00.000 +0200	Mostly Cloudy	rain	9.377778	9.377778	0.89	3.9284	204.0	14.9569	0.0	
3	2006-04-01 03:00:00.000 +0200	Partly Cloudy	rain	8.288889	5.944444	0.83	14.1036	269.0	15.8263	0.0	
4	2006-04-01 04:00:00.000 +0200	Mostly Cloudy	rain	8.755556	6.977778	0.83	11.0446	259.0	15.8263	0.0	

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 96453 entries, 0 to 96452
Data columns (total 12 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Formatted Date            96453 non-null  object
 1   Summary                   96453 non-null  object
 2   Precip Type               95936 non-null  object
 3   Temperature (C)           96453 non-null  float64
 4   Apparent Temperature (C)  96453 non-null  float64
 5   Humidity                  96453 non-null  float64
 6   Wind Speed (km/h)         96453 non-null  float64
 7   Wind Bearing (degrees)    96453 non-null  float64
 8   Visibility (km)           96453 non-null  float64
 9   Loud Cover                96453 non-null  float64
 10  Pressure (millibars)      96453 non-null  float64
 11  Daily Summary             96453 non-null  object
dtypes: float64(8), object(4)
memory usage: 8.8+ MB
None
```

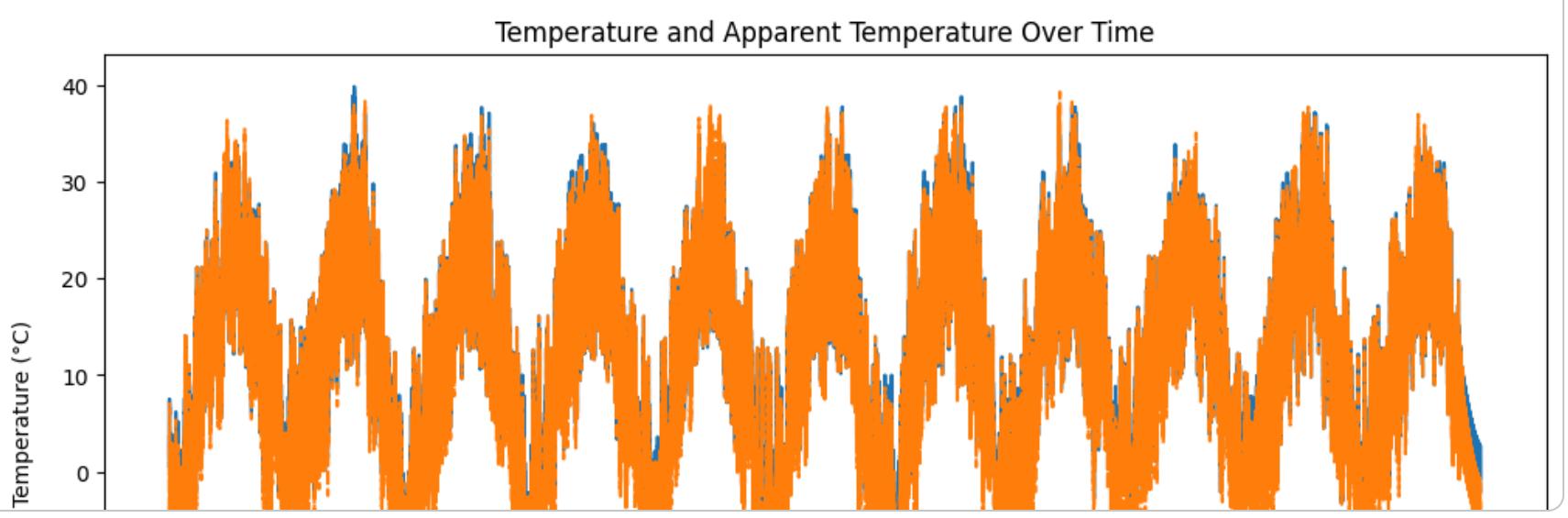
	Temperature (C)	Apparent Temperature (C)	Humidity	Wind Speed (km/h)	Wind Bearing (degrees)	Visibility (km)	Loud Cover	Pressure (millibars)
count	96453.000000	96453.000000	96453.000000	96453.000000	96453.000000	96453.000000	96453.0	96453.000000
mean	11.932678	10.855029	0.734899	10.810640	187.509232	10.347325	0.0	1003.235956
std	9.551546	10.696847	0.195473	6.913571	107.383428	4.192123	0.0	116.969906
min	-21.822222	-27.716667	0.000000	0.000000	0.000000	0.000000	0.0	0.000000
25%	4.688889	2.311111	0.600000	5.828200	116.000000	8.339800	0.0	1011.900000
50%	12.000000	12.000000	0.780000	9.965900	180.000000	10.046400	0.0	1016.450000
75%	18.838889	18.838889	0.890000	14.135800	290.000000	14.812000	0.0	1021.090000
max	39.905556	39.344444	1.000000	63.852600	359.000000	16.100000	0.0	1046.380000

```
Shape of the dataframe: (96453, 12)
```

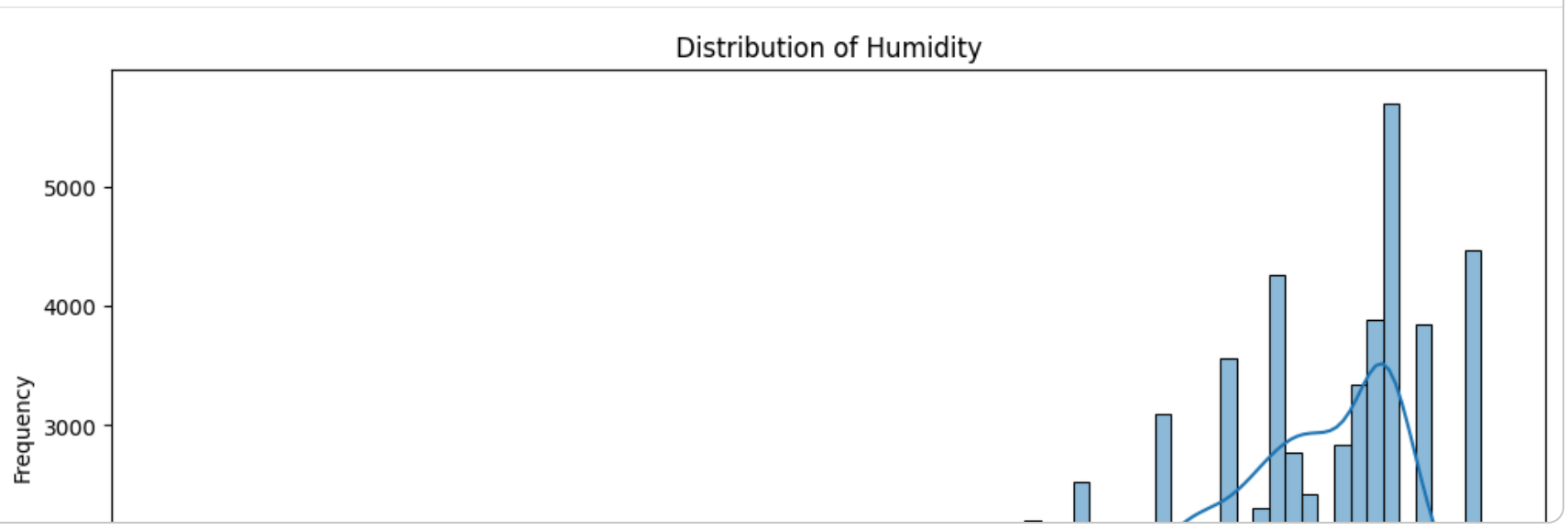
```
import matplotlib.pyplot as plt
import seaborn as sns
```

```
man['Formatted Date'] = pd.to_datetime(man['Formatted Date'], utc=True)
man = man.set_index('Formatted Date')
```

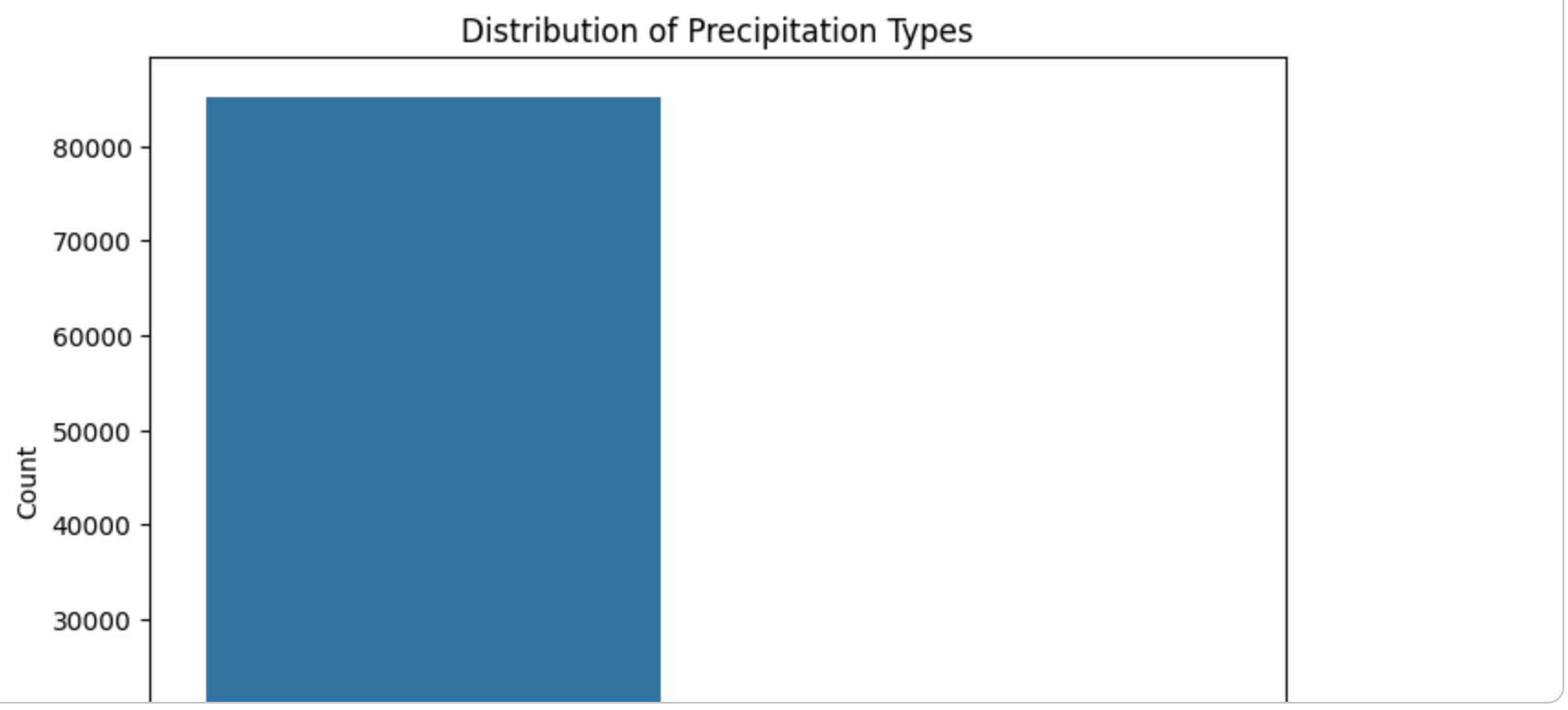
```
plt.figure(figsize=(12, 6))
sns.lineplot(data=man[['Temperature (C)', 'Apparent Temperature (C)']])
plt.title('Temperature and Apparent Temperature Over Time')
plt.xlabel('Date')
plt.ylabel('Temperature (°C)')
plt.show()
```




```
plt.figure(figsize=(12, 6))
sns.histplot(data=man, x='Humidity', kde=True)
plt.title('Distribution of Humidity')
plt.xlabel('Humidity')
plt.ylabel('Frequency')
plt.show()
```

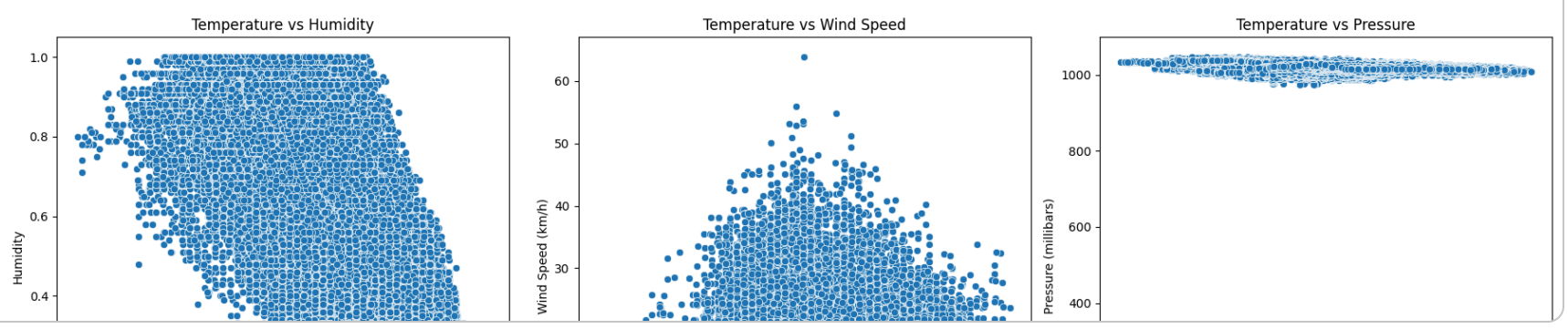



```
man_reset = man.reset_index()
plt.figure(figsize=(8, 6))
sns.countplot(data=man_reset, x='Precip Type')
plt.title('Distribution of Precipitation Types')
plt.xlabel('Precipitation Type')
plt.ylabel('Count')
plt.show()
```

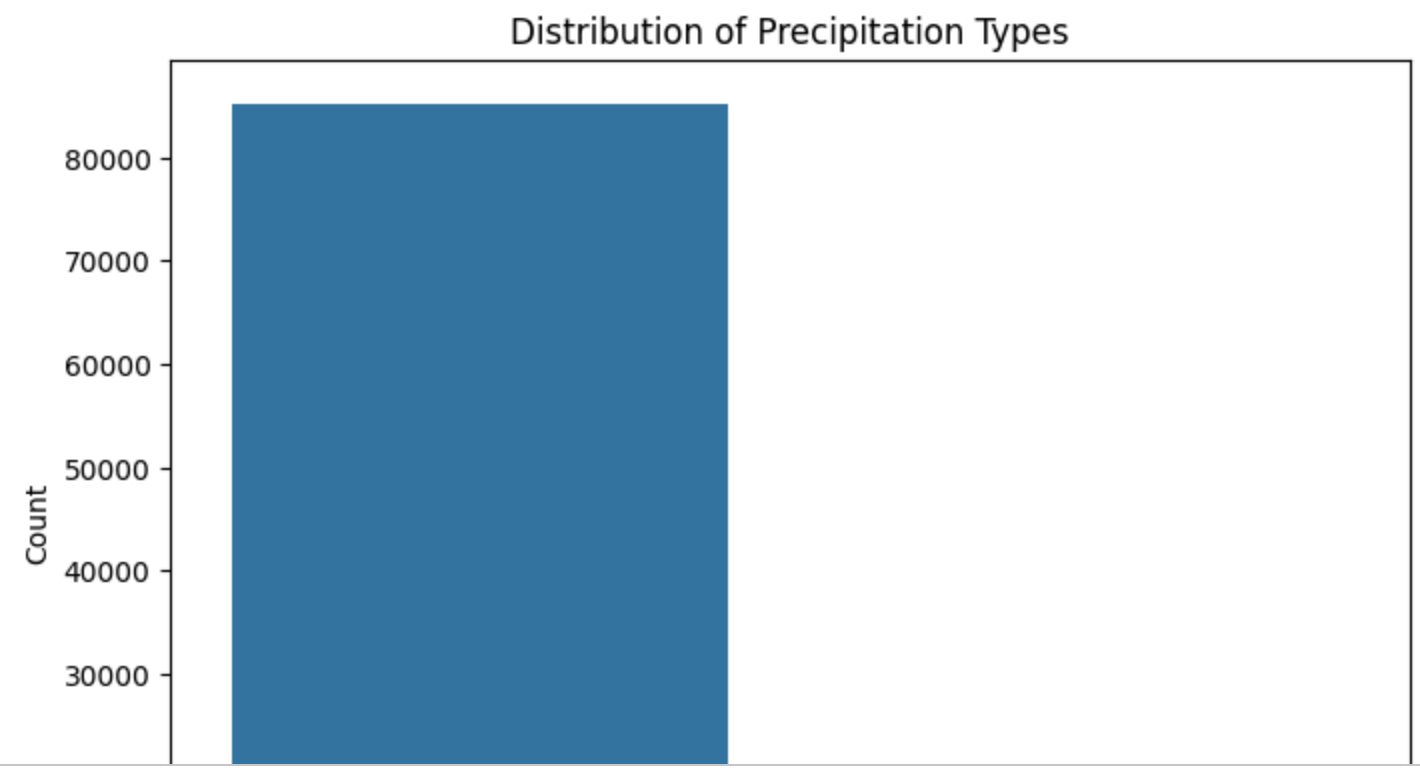



```
plt.figure(figsize=(18, 6))

plt.subplot(1, 3, 1)
sns.scatterplot(data=man, x='Temperature (C)', y='Humidity')
plt.title('Temperature vs Humidity')
plt.xlabel('Temperature (C)')
plt.ylabel('Humidity')
```

```
plt.subplot(1, 3, 2)
sns.scatterplot(data=man, x='Temperature (C)', y='Wind Speed (km/h)')
plt.title('Temperature vs Wind Speed')
plt.xlabel('Temperature (C)')
plt.ylabel('Wind Speed (km/h)')

plt.subplot(1, 3, 3)
sns.scatterplot(data=man, x='Temperature (C)', y='Pressure (millibars)')
plt.title('Temperature vs Pressure')
plt.xlabel('Temperature (C)')
plt.ylabel('Pressure (millibars)')

plt.tight_layout()
plt.show()
```




```
man_reset = man.reset_index()
plt.figure(figsize=(8, 6))
sns.countplot(data=man_reset, x='Precip Type')
plt.title('Distribution of Precipitation Types')
plt.xlabel('Precipitation Type')
plt.ylabel('Count')
plt.show()
```



Visualize relationships between variables

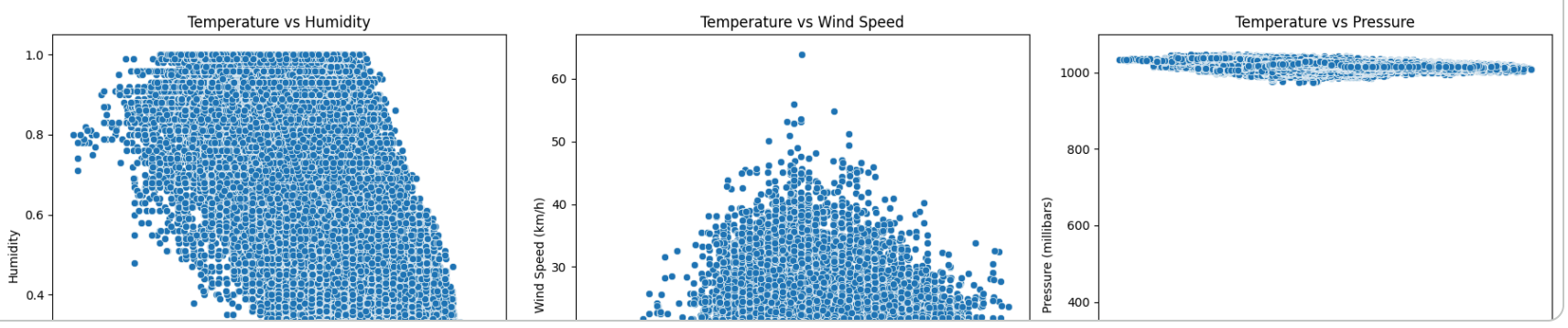
Subtask:

Create scatter plots to explore the relationships between numerical variables like temperature, humidity, wind speed, and pressure.

Reasoning: Create scatter plots to visualize the relationships between numerical variables as requested in the instructions.

```
plt.figure(figsize=(18, 6))

plt.subplot(1, 3, 1)
sns.scatterplot(data=man, x='Temperature (C)', y='Humidity')
plt.title('Temperature vs Humidity')
plt.xlabel('Temperature (C)')
plt.ylabel('Humidity')

plt.subplot(1, 3, 2)
sns.scatterplot(data=man, x='Temperature (C)', y='Wind Speed (km/h)')
plt.title('Temperature vs Wind Speed')
plt.xlabel('Temperature (C)')
```

```
    plt.ylabel('Wind Speed (km/h)')

    plt.subplot(1, 3, 3)
    sns.scatterplot(data=man, x='Temperature (C)', y='Pressure (millibars)')
    plt.title('Temperature vs Pressure')
    plt.xlabel('Temperature (C)')
    plt.ylabel('Pressure (millibars)')

    plt.tight_layout()
    plt.show()
```



Summary:

Data Analysis Key Findings

- The analysis successfully imported the necessary libraries for visualization, `matplotlib.pyplot` and `seaborn`.
- A line plot was generated showing the trends of 'Temperature (C)' and 'Apparent Temperature (C)' over time.
- A histogram with a Kernel Density Estimate was created to visualize the distribution of 'Humidity'.
- A bar plot showing the distribution of 'Precip Type' was successfully generated after resetting the DataFrame index to handle duplicate labels.
- Three scatter plots were created to visualize the relationships between 'Temperature (C)' and 'Humidity', 'Temperature (C)' and 'Wind Speed (km/h)', and 'Temperature (C)' and 'Pressure (millibars)'.

Insights or Next Steps

- Further investigation could explore correlations between 'Temperature (C)' and other numerical features based on the scatter plot visualizations.
- Analyze the temporal trends of 'Temperature (C)' and 'Apparent Temperature (C)' to identify seasonal patterns or significant deviations.