

第二次编程作业

第一题:

第一、三问:

#第1问

```
Sig_Eqs=Eqs()
```

```
Sig_Eqs.Func_1()
```

```
('CHINA', 2075045)
('TURKEY', 1092048)
('IRAN', 995406)
('ITALY', 498478)
('SYRIA', 369224)
('HAITI', 323478)
('AZERBAIJAN', 317219)
('JAPAN', 278085)
('ARMENIA', 191890)
('ISRAEL', 160120)
```

In [49]: #第3问

```
Sig_Eqs.CountEq_LargestEq('CHINA')
```

```
Sig_Eqs.Func_3()
```

```
EL SALVADOR 33 1916 9 7 1 20 48.0
SWITZERLAND 31 1601 9 18 None None None
SYRIA 29 1202 5 20 None None None
PORTUGAL 27 -60 None None None None None
AZORES 27 1968 2 28 None None 0.0
COSTA RICA 27 1822 5 7 None None None
MYANMAR (BURMA) 26 1912 5 23 2 24 6.0
NEW CALEDONIA 25 1875 3 28 None None None
ISRAEL 24 -31 9 2 None None 0.0
TAJIKISTAN 24 1907 10 21 4 23 36.0
AUSTRALIA 23 2004 12 23 14 59 4.4
KERMADec ISLANDS 23 1986 10 20 6 46 9.9
IRAQ 22 1864 12 2 None None None
MOROCCO 21 2023 9 8 22 11 1.0
ARGENTINA 21 1944 1 15 23 49 0.0
HAITI 20 1842 5 7 21 None None
SOUTH KOREA 19 1700 9 12 None None None
JAMAICA 19 1899 6 14 11 9 None
DOMINICAN REPUBLIC 18 1946 8 4 17 51 5.0
TONGA ISLANDS 18 1919 4 30 7 17 17.3
```

第二问:

`earthquake_data = data[['Year', 'Mag']]`: 这一行创建一个新的 DataFrame, 名为 `earthquake_data`, 其中只包含原始数据中的 'Year' 和 'Mag' (地震震级) 两列。它筛选出只与这次分析相关的列。
`large_quakes = earthquake_data[earthquake_data['Mag'] > 6.0]`: 这一行筛选 `earthquake_data` DataFrame, 只包括震级大于 6.0 的地震。它创建一个名为 `large_quakes` 的新 DataFrame, 其中包含这些筛选后的地震记录。

`earthquake_counts = large_quakes.groupby('Year')['Year'].count()`: 这里, `large_quakes` DataFrame 根据 'Year' 列进行分组。然后, 使用 `count()` 函数计算每年地震的数量。结果是一个 Series, 其中索引代表年份, 值代表每年的地震数量。

`plt.figure(figsize=(12, 6))`: 这一行初始化一个 Matplotlib 图, 指定了绘图的大小。

`plt.plot(earthquake_counts.index, earthquake_counts.values, marker='o', linestyle='-')`: 这里绘制时间序列图。`earthquake_counts.index` 代表年份, `earthquake_counts.values` 代表相应年份的地震数量。`marker='o'` 指定数据点用圆圈标记, `linestyle='-'` 指定用实线连接数据点。

`plt.title("Total number of earthquakes with magnitude larger than 6.0")`: 设置图表的标题。

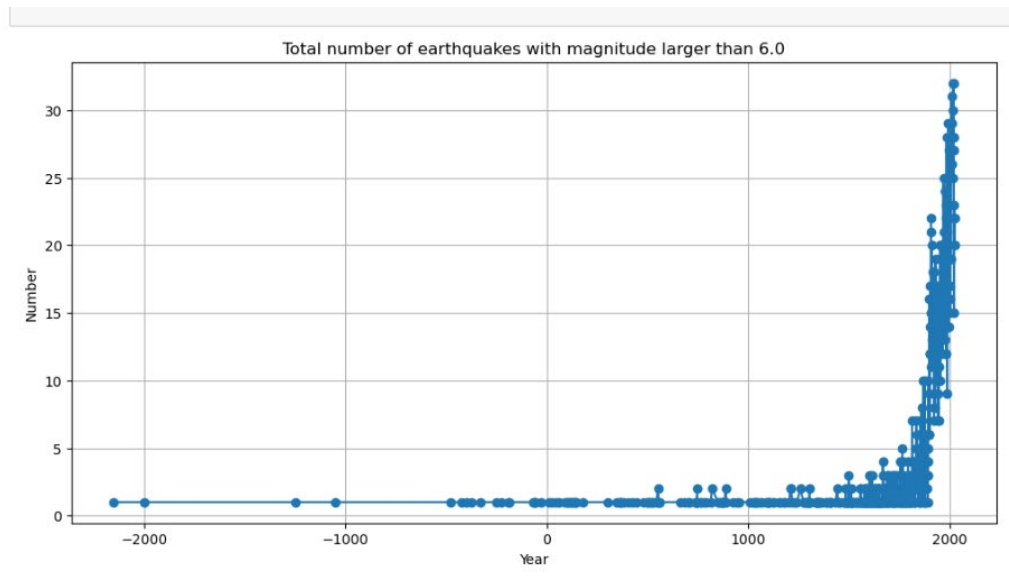
`plt.xlabel("Year")`: 设置 x 轴的标签。

`plt.ylabel("Number")`: 设置 y 轴的标签。

`plt.grid(True)`: 启用图表上的网格线。

plt.show(): 最后, 这一行在你的 Jupyter Notebook 中显示绘制的图表。

输出结果如下:



第二题:

代码解读:

`data['DATE'] = pd.to_datetime(data['DATE'])`: 这一行将名为'DATE'的列中的日期字符串转换为 Pandas 的日期时间格式, 使日期数据可操作。

`data['Year'] = data['DATE'].dt.year`: 在 DataFrame 中创建一个新列'Year', 其中包含从'DATE'列中提取的年份信息。

`data['Month'] = data['DATE'].dt.month`: 类似地, 这一行创建一个新列'Month', 其中包含从'DATE'列中提取的月份信息。

`data['WND'] = data['WND'].str.split(',').str[3].astype(float)`: 这一行将名为'WND'的列中的字符串数据分割为逗号分隔的部分, 并选择第四个部分(索引 3)。然后, 将这些部分转换为浮点数, 以得到风速的数值。

`current_year = pd.Timestamp.now().year`: 获取当前年份。

`data_last_10_years = data[data['Year'] >= current_year - 10]`: 这一行筛选出包括最近 10 年的数据, 使用 `current_year - 10` 来确定截止年份。

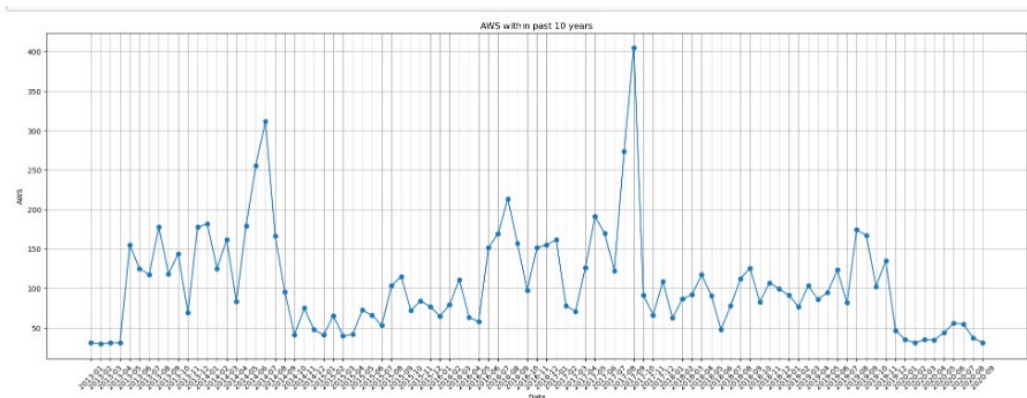
`result = data_last_10_years.groupby(['Year', 'Month'])['WND'].mean().reset_index()`: 这一行将数据按年和月进行分组, 并计算每个组(每年每月)中'WND'列的平均值。然后, 重置索引以创建新的 DataFrame。

`result['Year-Month'] = result['Year'].astype(str) + '-' + result['Month'].astype(str).str.zfill(2)`: 创建一个新列'Year-Month', 其中包含年和月的组合, 格式为'YYYY-MM', 并使用 `str.zfill(2)`来确保月份始终占两位数。

`result = result.sort_values('Year-Month')`: 对结果 DataFrame 按'Year-Month'列进行排序, 以确保时间顺序正确。

`result = result[['Year-Month', 'WND']].reset_index(drop=True)`: 最后, 选择'Year-Month'和'WND'两列, 并重置索引, 以获得最终的结果 DataFrame, 包含时间序列和相应的风速数据。

输出结果:



第三题:

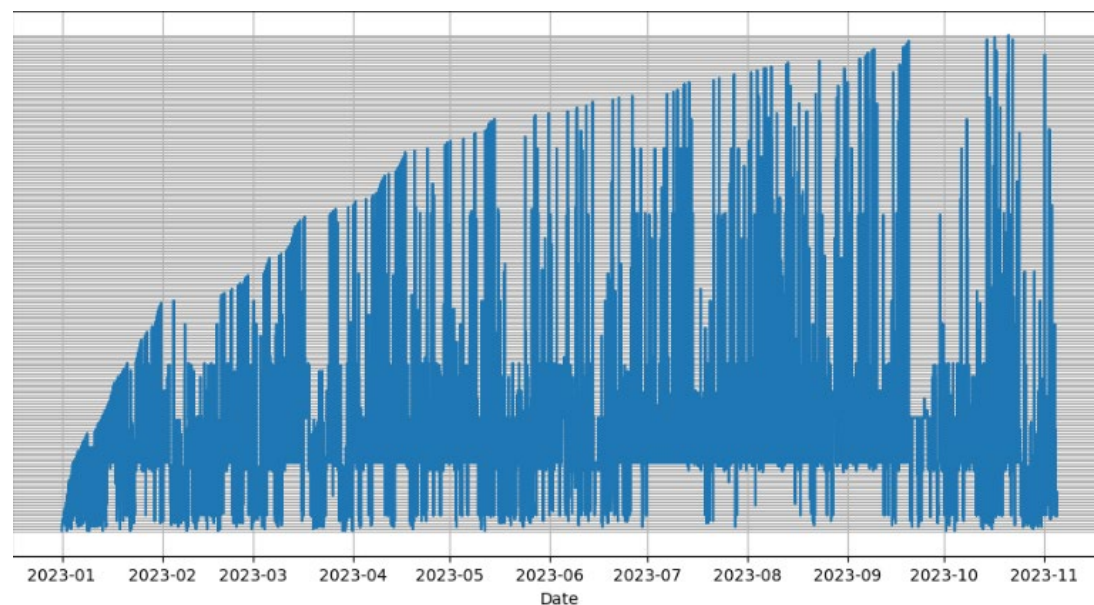
第一问: 直接使用 `cleaned_data = data.dropna(subset=columns_to_check, how='any')` 进行数据检查清除即可, 并打印。

原始数据行数: 9598

清理后的数据行数: 9598

第二问:

直接计算变量 VIS 的时间序列即可, 输出结果如下:



第三问: 首先使用 `.str.replace()` 方法和正则表达式 `[^\d.]` 来删除非数字字符, 然后使用 `.astype(float)` 将字符串转换为浮点数。这样, 'TMP' 和 'DEW' 变量都以相同的数据类型进行相关性计算和其他统计操作。其中一个输出数据如下:

```
'TMP' 和 'DEW' 的相关性: 0.34624616741152503
'TMP' 的均值: 2043.3723692435924
'DEW' 的均值: 1512.897374453011
'TMP' 的标准差: 1348.2911987888717
'DEW' 的标准差: 1292.5655653557192
```

其余四组数据

MW1 均值: 5.1 标准差 0, 分别与上述两组数据比较:

与 TMP 相关性 0.02323515858, 与 DEM 相关性 0.015654285

QD1 均值: 3998.25429654268, 标准差: 2563.256989684 分别比较:

与 TMP 相关性 0.1543654685356, 与 DEM 相关性 0.2568454866