First we will read the data, noting the different columns and accordingly, label the columns.

1. Year: Year of observation.
2. Month: Month of observation.
3. Day: Day of observation.
4. Fractional Year: Decimal representation of the date.
5. Sunspot Number: Daily sunspot count (can be missing -1).
6. Standard Deviation: Uncertainty in sunspot count.
7. Number of Observations: How many observations contributed to the sunspot count.
8. Definitive or Provisional: 1 for final data, 0 for provisional.

```
import pandas as pd

data = pd.read_csv('raw_data.csv', sep = ";", header = None)

data.columns = ['Year', 'Month', 'Day', 'Fractional Year', 'Sunspot Number',
                'Sunspot Std Dev', 'Num Observations', 'Data Quality']

print(data)
```

```
       Year  Month  Day  Fractional Year  Sunspot Number  Sunspot Std Dev  \
0      1818      1    1         1818.001              -1             -1.0
1      1818      1    2         1818.004              -1             -1.0
2      1818      1    3         1818.007              -1             -1.0
3      1818      1    4         1818.010              -1             -1.0
4      1818      1    5         1818.012              -1             -1.0
...     ...    ...  ...              ...             ...              ...
75601  2024     12   27         2024.988             258             34.1
75602  2024     12   28         2024.990             252             52.2
75603  2024     12   29         2024.993             234             32.2
75604  2024     12   30         2024.996             218             23.6
75605  2024     12   31         2024.999             179             22.9

       Num Observations  Data Quality
0                     0             1
1                     0             1
2                     0             1
3                     0             1
4                     0             1
...                 ...           ...
75601                22             0
75602                26             0
75603                16             0
75604                17             0
75605                12             0

[75606 rows x 8 columns]
```

The year, month and day columns have been combined as the date column, which was then set as the index for the dataset, so it will be easier to run models on the data.

```
required_columns = ['Year','Month','Day']

data['Date'] = pd.to_datetime(data[required_columns])

data.set_index('Date', inplace=True)

data.drop(['Year', 'Month', 'Day', 'Fractional Year'], axis=1, inplace=True)

print(data)
```

```
            Sunspot Number  Sunspot Std Dev  Num Observations  Data Quality
Date
1818-01-01              -1             -1.0                 0             1
1818-01-02              -1             -1.0                 0             1
1818-01-03              -1             -1.0                 0             1
1818-01-04              -1             -1.0                 0             1
1818-01-05              -1             -1.0                 0             1
...                    ...              ...               ...           ...
2024-12-27             258             34.1                22             0
2024-12-28             252             52.2                26             0
2024-12-29             234             32.2                16             0
2024-12-30             218             23.6                17             0
2024-12-31             179             22.9                12             0

[75606 rows x 4 columns]
```

We remove the sunspot Numbers that are -1, and save the remaining rows in a new csv titled "filtered_sunspot_data.csv".

```
data_filtered = data[data['Sunspot Number'] != -1]

data_filtered = data_filtered.reset_index()

required_columns = ['Date', 'Sunspot Number', 'Sunspot Std Dev', 'Num Observations', 'Data Quality']
data_filtered = data_filtered[required_columns]


data_filtered.to_csv('filtered_sunspot_data.csv', index=False)

print("Filtered data has been saved to 'filtered_sunspot_data.csv'")

data = pd.read_csv('filtered_sunspot_data.csv', index_col='Date', parse_dates=True)
print(data)
```

```
Filtered data has been saved to 'filtered_sunspot_data.csv'
            Sunspot Number  Sunspot Std Dev  Num Observations  Data Quality
Date
1818-01-08              65             10.2                 1             1
1818-01-13              37              7.7                 1             1
1818-01-17              77             11.1                 1             1
1818-01-18              98             12.6                 1             1
1818-01-19             105             13.0                 1             1
...                    ...              ...               ...           ...
2024-12-27             258             34.1                22             0
2024-12-28             252             52.2                26             0
2024-12-29             234             32.2                16             0
2024-12-30             218             23.6                17             0
2024-12-31             179             22.9                12             0

[72359 rows x 4 columns]
```

```
# Select features for anomaly detection
features = data[['Sunspot Number', 'Sunspot Std Dev']]
```

```
from sklearn.ensemble import IsolationForest
# Configure Isolation Forest
iso_forest = IsolationForest(n_estimators=100, contamination=0.05, random_state=42)

# Fit the model
iso_forest.fit(features)
```

```
IsolationForest
IsolationForest(contamination=0.05, random_state=42)
```

```
# Predict anomalies (outliers)
data['Anomaly_Score'] = iso_forest.fit_predict(features)
```

```
# Count anomalies and normal points
anomalies_count = data['Anomaly_Score'].value_counts()
print(anomalies_count)
```

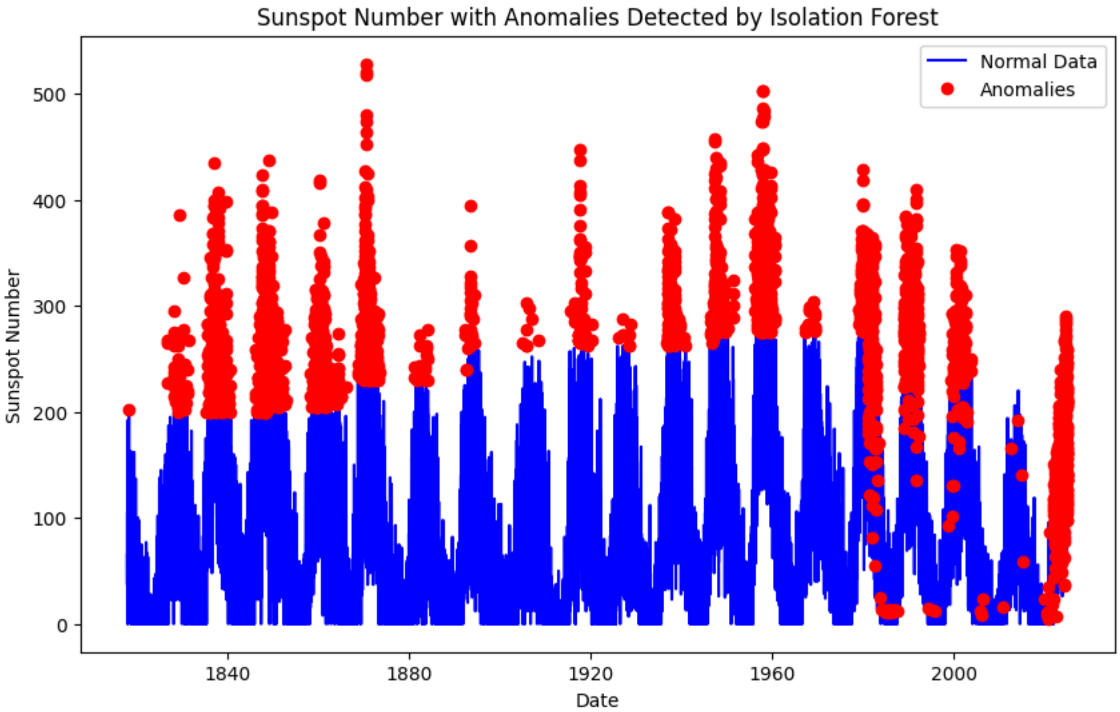
```
Anomaly_Score
 1    68744
-1     3615
Name: count, dtype: int64
```

```
# Filter out anomalies
anomalies = data[data['Anomaly_Score'] == -1]
normal_data = data[data['Anomaly_Score'] == 1]

# Preview anomalies
print(anomalies)
```

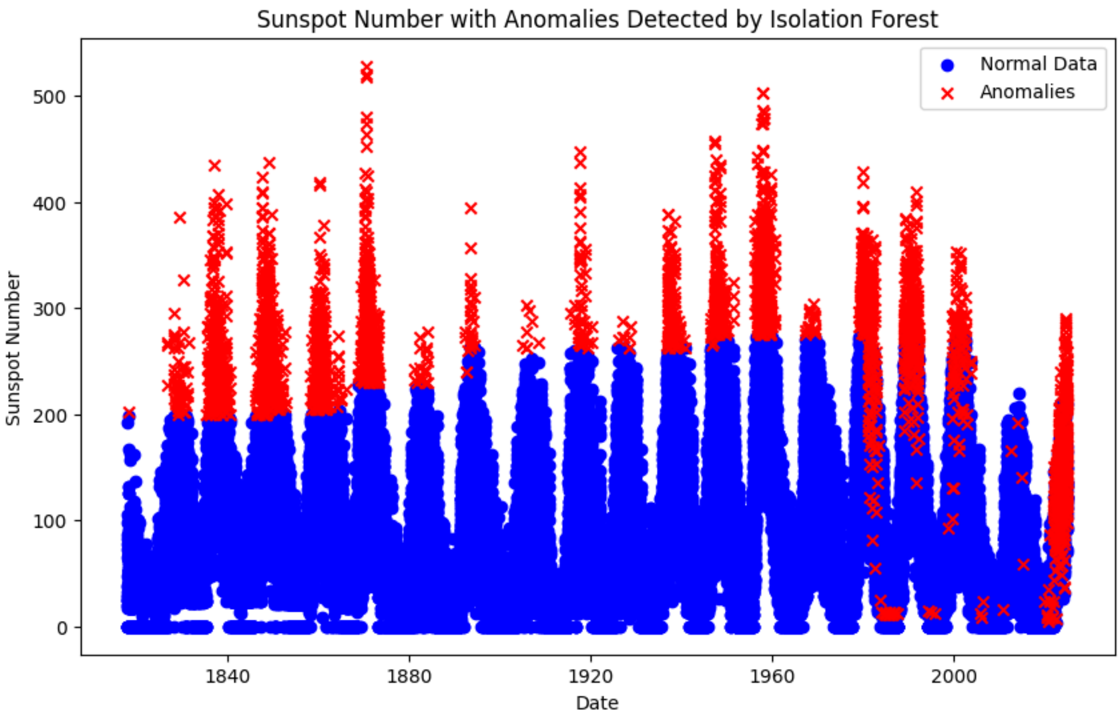
```
            Sunspot Number  Sunspot Std Dev  Num Observations  Data Quality  \
Date
1818-05-29             202             18.0                 1             1
1826-11-18             265             20.6                 1             1
1826-12-07             268             20.8                 1             1
1826-12-09             227             19.1                 1             1
1828-03-14             215             18.6                 1             1
...                    ...              ...               ...           ...
2024-12-27             258             34.1                22             0
2024-12-28             252             52.2                26             0
2024-12-29             234             32.2                16             0
2024-12-30             218             23.6                17             0
```

```
2024-12-31          179          22.9          12          0

            Anomaly_Score
Date
1818-05-29             -1
1826-11-18             -1
1826-12-07             -1
1826-12-09             -1
1828-03-14             -1
...                   ...
2024-12-27             -1
2024-12-28             -1
2024-12-29             -1
2024-12-30             -1
2024-12-31             -1

[3615 rows x 5 columns]
```

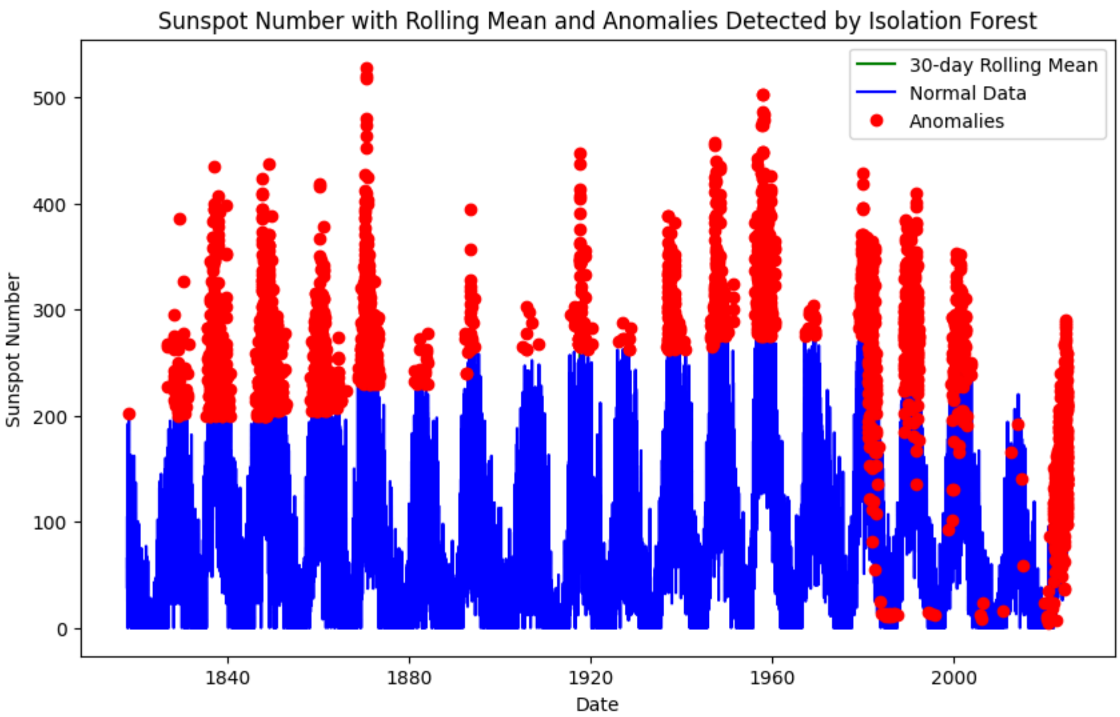
```
import matplotlib.pyplot as plt

# Plot sunspot number with anomalies highlighted
plt.figure(figsize=(10,6))

# Plot the normal points
plt.plot(normal_data.index, normal_data['Sunspot Number'], 'b-', label='Normal Data')

# Plot the anomalies
plt.plot(anomalies.index, anomalies['Sunspot Number'], 'ro', label='Anomalies')

plt.title('Sunspot Number with Anomalies Detected by Isolation Forest')
plt.xlabel('Date')
plt.ylabel('Sunspot Number')
plt.legend()
plt.show()
```



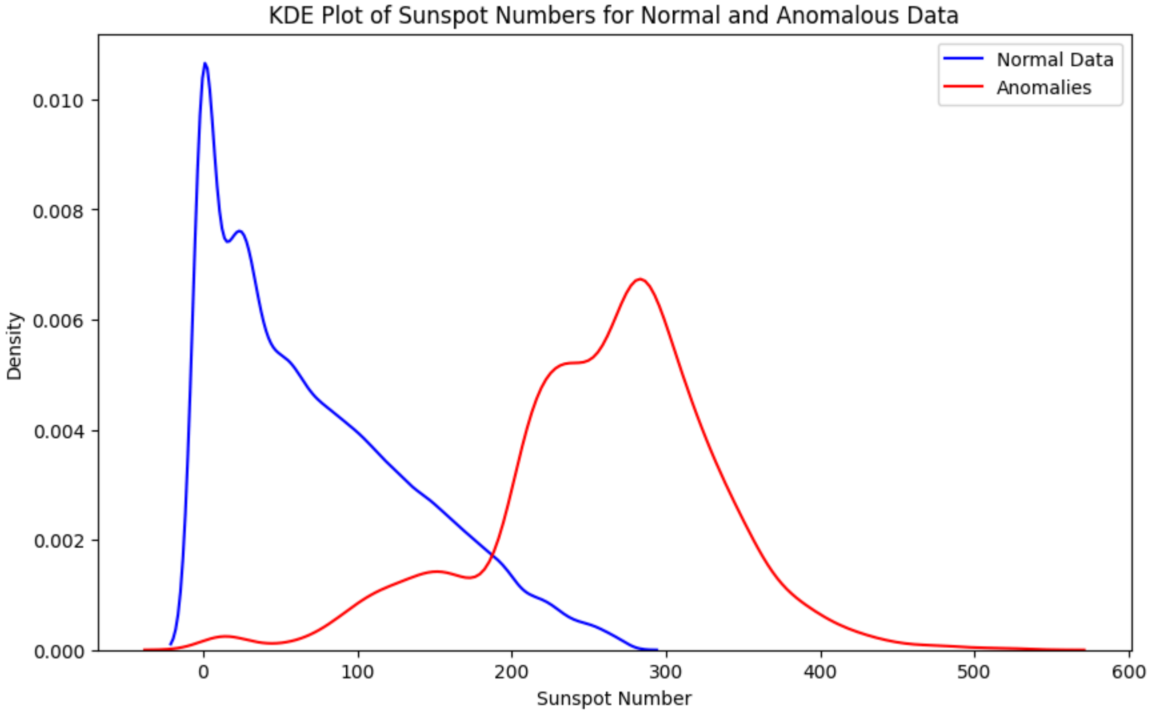

```
import matplotlib.pyplot as plt

# Scatter plot for normal and anomaly points
plt.figure(figsize=(10,6))

# Plot normal data points
plt.scatter(normal_data.index, normal_data['Sunspot Number'], label='Normal Data', color='blue')

# Plot anomalies with different marker
plt.scatter(anomalies.index, anomalies['Sunspot Number'], label='Anomalies', color='red', marker='x')

plt.title('Sunspot Number with Anomalies Detected by Isolation Forest')
plt.xlabel('Date')
plt.ylabel('Sunspot Number')
plt.legend()
plt.show()
```




```
# Calculate rolling mean to smooth out the series
rolling_mean = data['Sunspot Number'].rolling(window=30).mean()

# Plot the rolling mean along with anomalies
plt.figure(figsize=(10,6))

# Plot rolling mean
plt.plot(data.index, rolling_mean, label='30-day Rolling Mean', color='green')

# Plot normal sunspot numbers
plt.plot(normal_data.index, normal_data['Sunspot Number'], 'b-', label='Normal Data')

# Plot anomalies
plt.plot(anomalies.index, anomalies['Sunspot Number'], 'ro', label='Anomalies')

plt.title('Sunspot Number with Rolling Mean and Anomalies Detected by Isolation Forest')
plt.xlabel('Date')
plt.ylabel('Sunspot Number')
plt.legend()
plt.show()
```

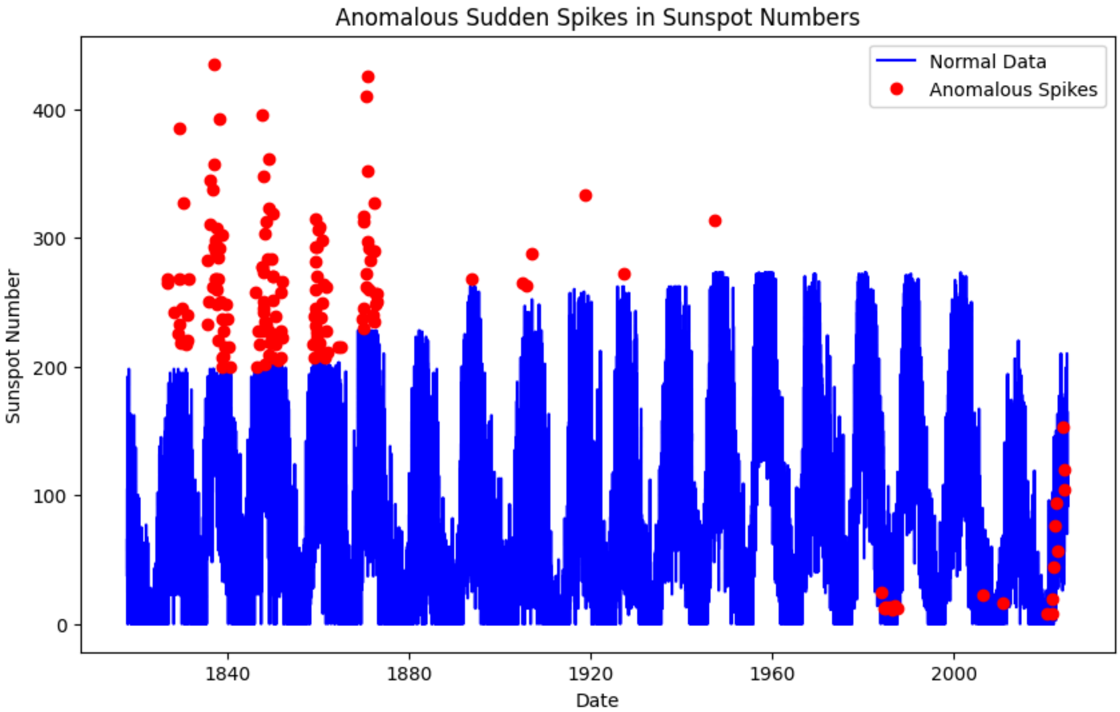


```
import seaborn as sns

# KDE plot for normal data and anomalies
plt.figure(figsize=(10,6))

# Plot KDE for normal sunspot numbers
sns.kdeplot(normal_data['Sunspot Number'], label='Normal Data', color='blue')

# Plot KDE for anomalous sunspot numbers
sns.kdeplot(anomalies['Sunspot Number'], label='Anomalies', color='red')

plt.title('KDE Plot of Sunspot Numbers for Normal and Anomalous Data')
plt.xlabel('Sunspot Number')
plt.legend()
plt.show()
```



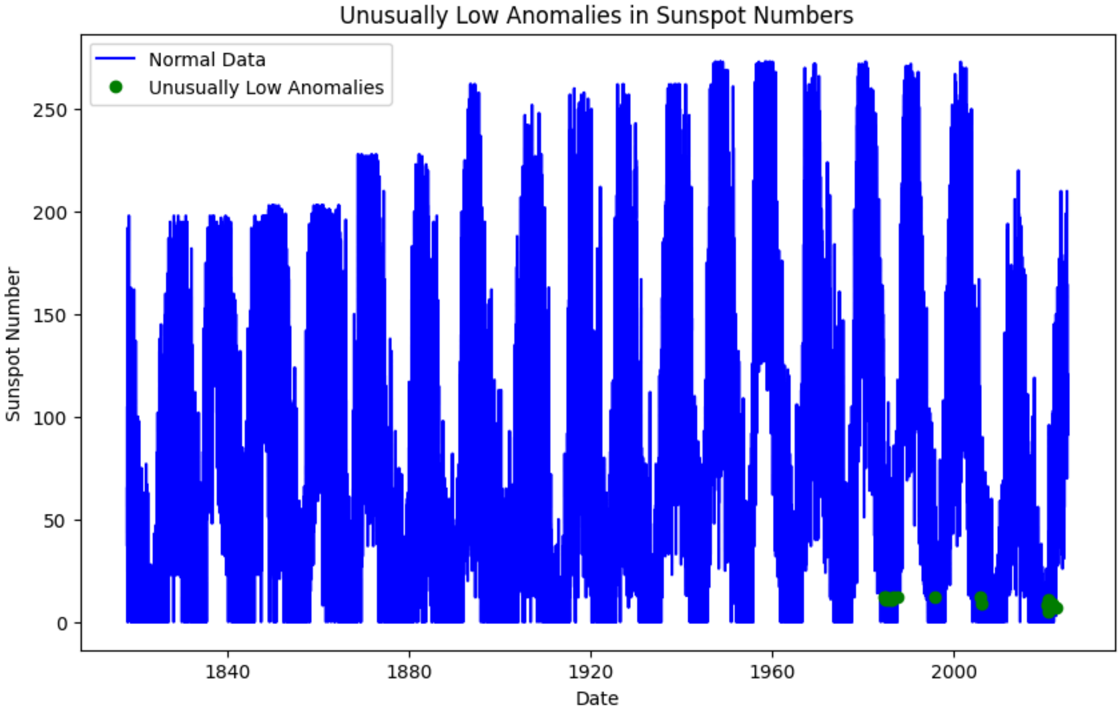

```
# Calculate the percentage change of sunspot numbers
data['Percentage Change'] = data['Sunspot Number'].pct_change() * 100

# Define a threshold for a "sudden spike" (e.g., change > 50%)
spikes = data[data['Percentage Change'] > 50]

# Filter anomalies that are also spikes
anomalous_spikes = anomalies[anomalies.index.isin(spikes.index)]

# Display sudden spikes that are also anomalies
print(anomalous_spikes)

# Plot these spikes
plt.figure(figsize=(10,6))

# Plot normal data
plt.plot(normal_data.index, normal_data['Sunspot Number'], 'b-', label='Normal Data')

# Highlight anomalies that are spikes
plt.plot(anomalous_spikes.index, anomalous_spikes['Sunspot Number'], 'ro', label='Anomalous Spikes')

plt.title('Anomalous Sudden Spikes in Sunspot Numbers')
plt.xlabel('Date')
plt.ylabel('Sunspot Number')
plt.legend()
plt.show()
```

```
            Sunspot Number  Sunspot Std Dev  Num Observations  Data Quality  \
Date
1826-11-18             265             20.6                 1             1
1826-12-07             268             20.8                 1             1
1828-06-19             242             19.7                 1             1
1829-04-21             225             19.0                 1             1
1829-06-29             385             24.9                 1             1
...                    ...              ...               ...           ...
2022-06-13              94             19.9                57             1
2022-10-21              57             18.4                34             1
2023-11-21             153             18.9                35             1
2024-02-23             104             27.4                34             1
2024-03-18             120             17.4                49             1

            Anomaly_Score
Date
1826-11-18             -1
1826-12-07             -1
1828-06-19             -1
1829-04-21             -1
1829-06-29             -1
...                   ...
2022-06-13             -1
2022-10-21             -1
2023-11-21             -1
2024-02-23             -1
2024-03-18             -1

[174 rows x 5 columns]
```




```
# Calculate the 10th percentile to define unusually low values
low_threshold = data['Sunspot Number'].quantile(0.20)

# Filter anomalies that have unusually low values
unusually_low_anomalies = anomalies[anomalies['Sunspot Number'] < low_threshold]

# Display these unusually low anomalies
print(unusually_low_anomalies)

# Plot these anomalies
plt.figure(figsize=(10,6))

# Plot normal data
plt.plot(normal_data.index, normal_data['Sunspot Number'], 'b-', label='Normal Data')

# Highlight unusually low anomalies
plt.plot(unusually_low_anomalies.index, unusually_low_anomalies['Sunspot Number'], 'go', label='Unusually Low Anomalies')

plt.title('Unusually Low Anomalies in Sunspot Numbers')
plt.xlabel('Date')
plt.ylabel('Sunspot Number')
plt.legend()
plt.show()
```

```
            Sunspot Number  Sunspot Std Dev  Num Observations  Data Quality  \
Date
1984-09-15              12             10.9                 8             1
1984-12-31              11             10.6                11             1
1985-01-08              12             11.4                16             1
1985-08-17              11             10.0                19             1
1985-12-08              11             10.3                10             1
1986-04-01              11             10.2                18             1
1986-10-16              12             11.6                15             1
1987-06-11              12             10.0                24             1
1995-11-03              12             10.8                18             1
2005-10-23              12             10.9                19             1
2006-02-11               9              9.4                17             1
2020-07-07               8             10.4                38             1
2020-08-03              11             11.6                45             1
2020-11-01               5              8.7                33             1
2021-01-28               7              9.0                32             1
2021-08-01               8             10.2                33             1
2021-08-13               9             11.1                45             1
2022-06-07               7             11.0                31             1

            Anomaly_Score
Date
1984-09-15             -1
1984-12-31             -1
1985-01-08             -1
1985-08-17             -1
1985-12-08             -1
1986-04-01             -1
1986-10-16             -1
1987-06-11             -1
1995-11-03             -1
2005-10-23             -1
2006-02-11             -1
2020-07-07             -1
2020-08-03             -1
2020-11-01             -1
2021-01-28             -1
2021-08-01             -1
2021-08-13             -1
2022-06-07             -1
```




```
# Get the time differences between consecutive anomalies
anomalies['Time Diff'] = anomalies.index.to_series().diff().dt.days

# Plot histogram of time gaps between anomalies
plt.figure(figsize=(10,6))
plt.hist(anomalies['Time Diff'].dropna(), bins=20, color='purple', alpha=0.7)

plt.title('Distribution of Time Gaps Between Anomalies')
plt.xlabel('Days Between Anomalies')
plt.ylabel('Frequency')
plt.show()
```

```
<ipython-input-15-d2b68ffe0f60>:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
  anomalies['Time Diff'] = anomalies.index.to_series().diff().dt.days
```