



VISUALIZING TIME SERIES DATA IN PYTHON

Clean your time series data

Thomas Vincent

Senior Data Science Engineer, DigitalOcean



The CO2 level time series

A snippet of the weekly measurements of CO2 levels at the Mauna Loa Observatory, Hawaii.

```
datestamp    datestamp    co2
1958-03-29   1958-03-29   316.1
1958-04-05   1958-04-05   317.3
1958-04-12   1958-04-12   317.6
...
...
2001-12-15   2001-12-15   371.2
2001-12-22   2001-12-22   371.3
2001-12-29   2001-12-29   371.5
```



Finding missing values in a DataFrame

```
In [1]: print(df.isnull())  
          co2
```

```
datestamp
```

```
1958-03-29    False
```

```
1958-04-05    False
```

```
1958-04-12    False
```

```
...
```

```
In [2]: print(df.notnull())  
          co2
```

```
datestamp
```

```
1958-03-29     True
```

```
1958-04-05     True
```

```
1958-04-12     True
```

```
...
```



Counting missing values in a DataFrame

```
In [1]: print(df.isnull().sum())  
datestamp      0  
co2             59  
dtype: int64
```



Replacing missing values in a DataFrame

```
In [1]: print(df)
```

```
...
```

```
5    1958-05-03    316.9
```

```
6    1958-05-10      NaN
```

```
7    1958-05-17    317.5
```

```
...
```

```
In [2]: df = df.fillna(method='bfill')
```

```
In [3]: print(df)
```

```
...
```

```
5    1958-05-03    316.9
```

```
6    1958-05-10    317.5
```

```
7    1958-05-17    317.5
```

```
...
```



VISUALIZING TIME SERIES DATA IN PYTHON

Let's practice!



VISUALIZING TIME SERIES DATA IN PYTHON

Plot aggregates of your data

Thomas Vincent

Senior Data Science Engineer, DigitalOcean



Moving averages

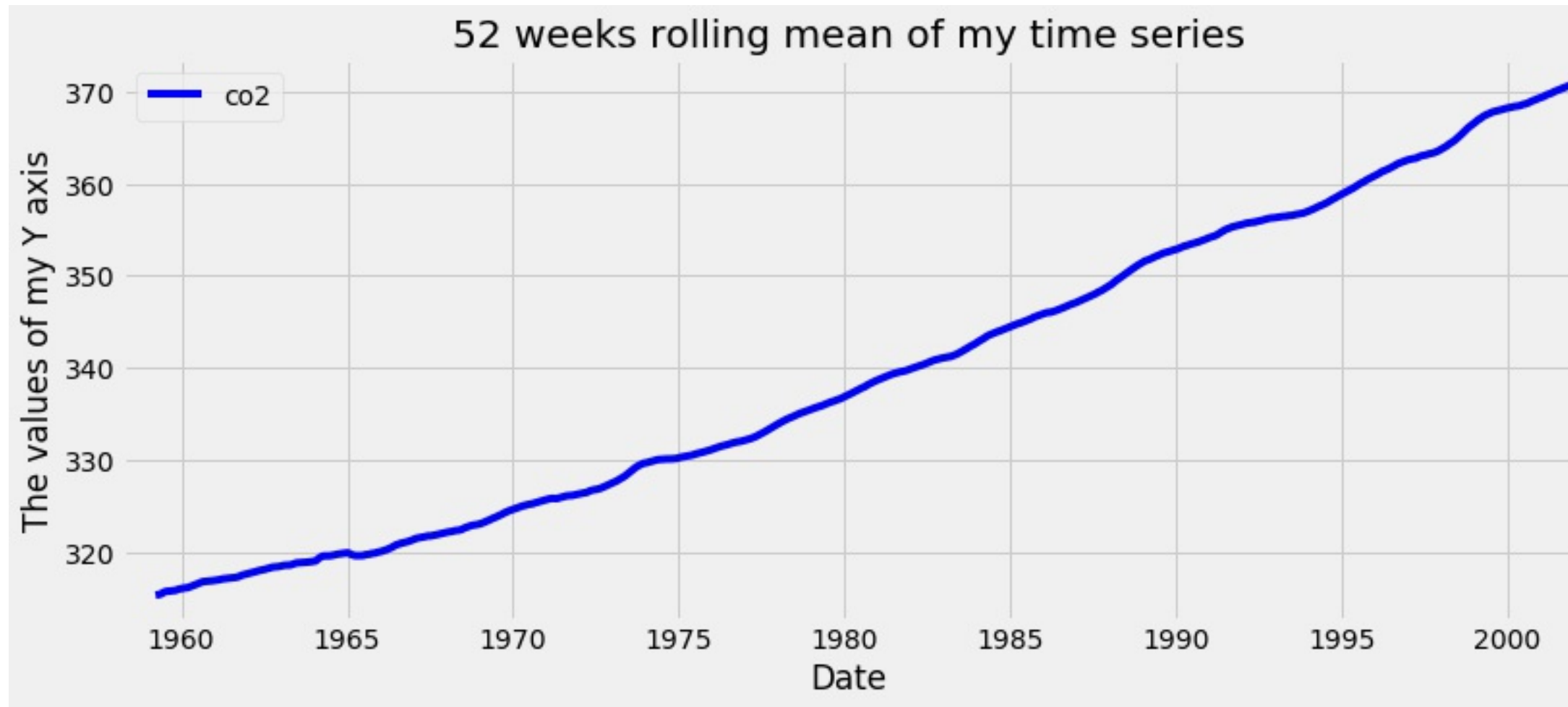
- In the field of time series analysis, a moving average can be used for many different purposes:
 - smoothing out short-term fluctuations
 - removing outliers
 - highlighting long-term trends or cycles.

The moving average model

```
In [1]: co2_levels_mean = co2_levels.rolling(window=52).mean()  
  
In [2]: ax = co2_levels_mean.plot()  
  
In [3]: ax.set_xlabel("Date")  
  
In [4]: ax.set_ylabel("The values of my Y axis")  
  
In [5]: ax.set_title("52 weeks rolling mean of my time series")  
  
In [6]: plt.show()
```



A plot of the moving average for the CO2 data





Computing aggregate values of your time series

```
In [1]: co2_levels.index
DatetimeIndex(['1958-03-29', '1958-04-05', ...],
              dtype='datetime64[ns]', name='datestamp',
              length=2284, freq=None)

In [2]: print(co2_levels.index.month)
array([ 3,  4,  4, ..., 12, 12, 12], dtype=int32)

In [3]: print(co2_levels.index.year)
array([1958, 1958, 1958, ..., 2001, 2001, 2001], dtype=int32)
```



Plotting aggregate values of your time series

```
In [1]: index_month = co2_levels.index.month
```

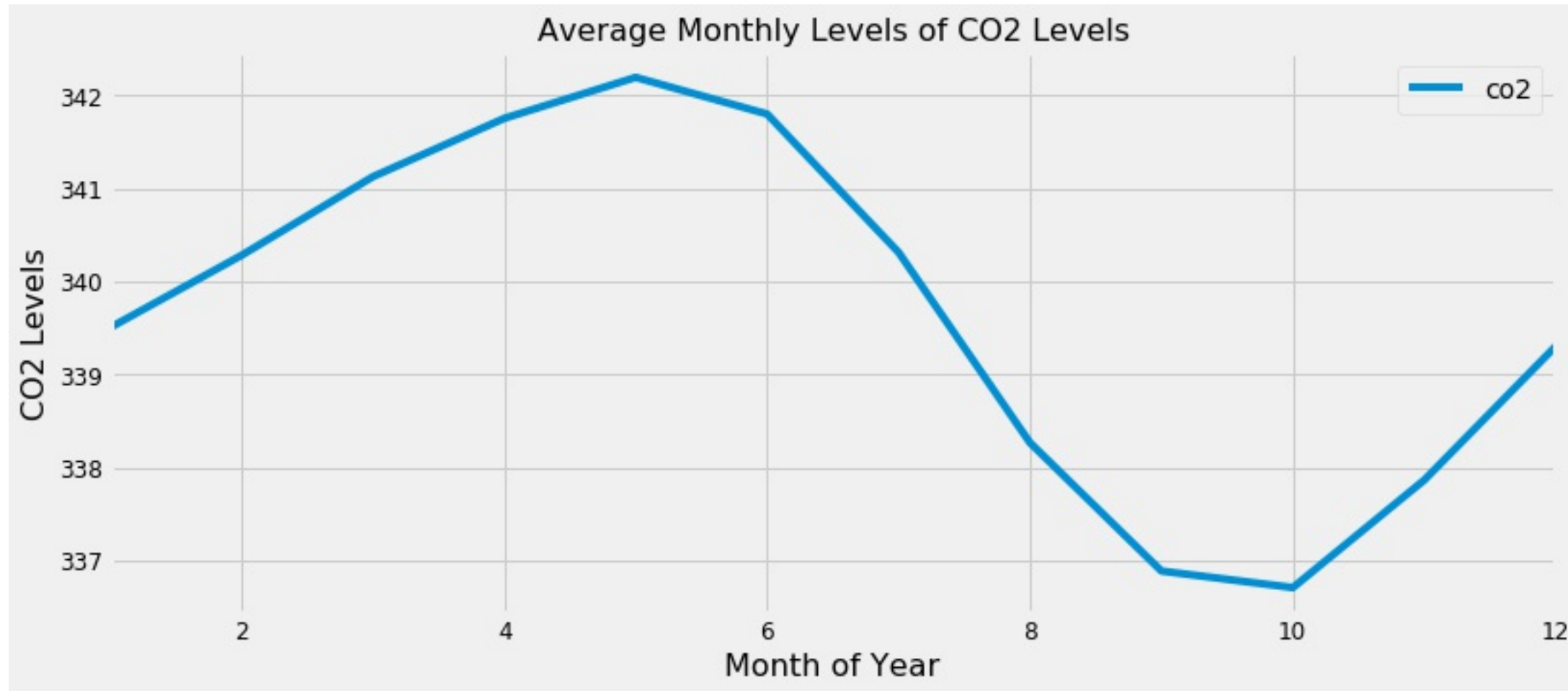
```
In [2]: co2_levels_by_month = co2_levels.groupby(index_month).m
```

```
In [3]: co2_levels_by_month.plot()
```

```
In [4]: plt.show()
```



Plotting aggregate values of your time series





VISUALIZING TIME SERIES DATA IN PYTHON

Let's practice!



VISUALIZING TIME SERIES DATA IN PYTHON

Summarizing the values in your time series data

Thomas Vincent

Senior Data Science Engineer, DigitalOcean



Obtaining numerical summaries of your data

- What is the average value of this data?
- What is the maximum value observed in this time series?



Obtaining numerical summaries of your data

The `.describe()` method automatically computes key statistics of all numeric columns in your DataFrame

```
In [1]: print(df.describe())
```

```
              co2  
count  2284.000000  
mean    339.657750  
std     17.100899  
min     313.000000  
25%     323.975000  
50%     337.700000  
75%     354.500000  
max     373.900000
```



Summarizing your data with boxplots

```
In [1]: ax1 = df.boxplot()
```

```
In [2]: ax1.set_xlabel('Your first boxplot')
```

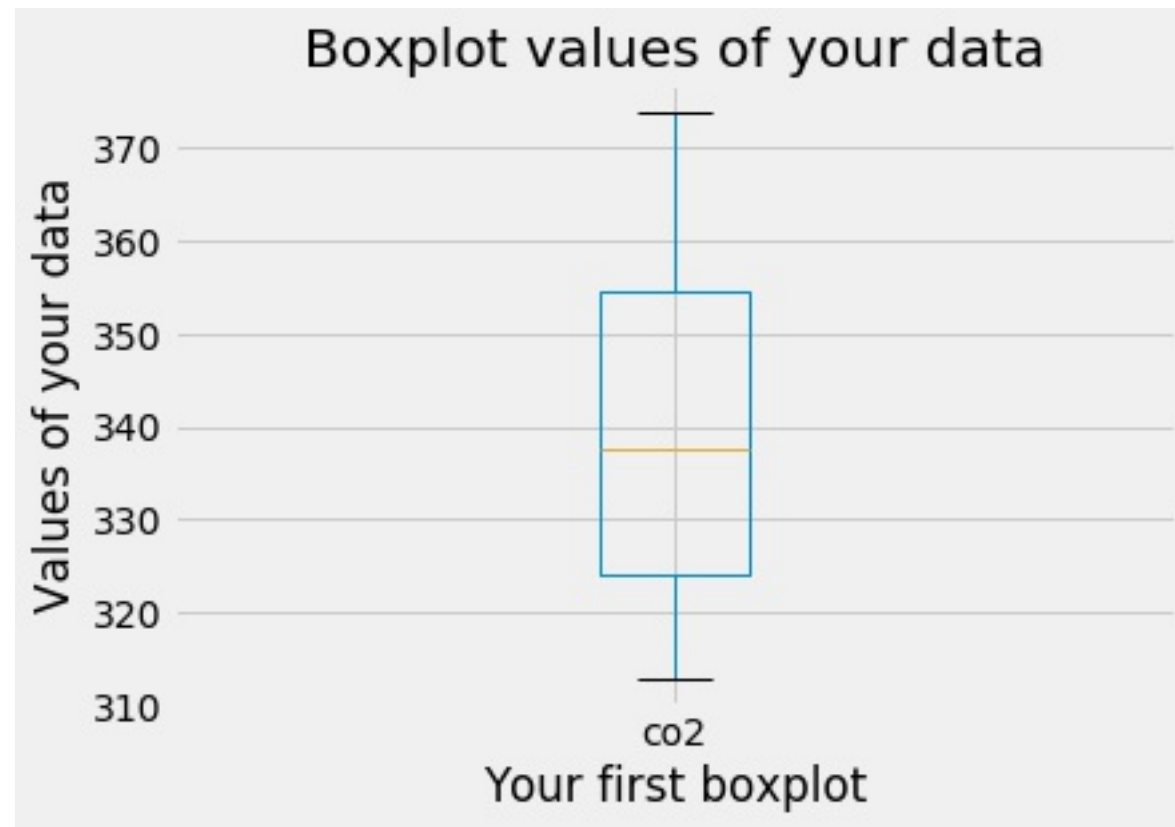
```
In [3]: ax1.set_ylabel('Values of your data')
```

```
In [4]: ax1.set_title('Boxplot values of your data')
```

```
In [5]: plt.show()
```



A boxplot of the values in the CO2 data





Summarizing your data with histograms

```
In [1]: ax2 = df.plot(kind='hist', bins=100)
```

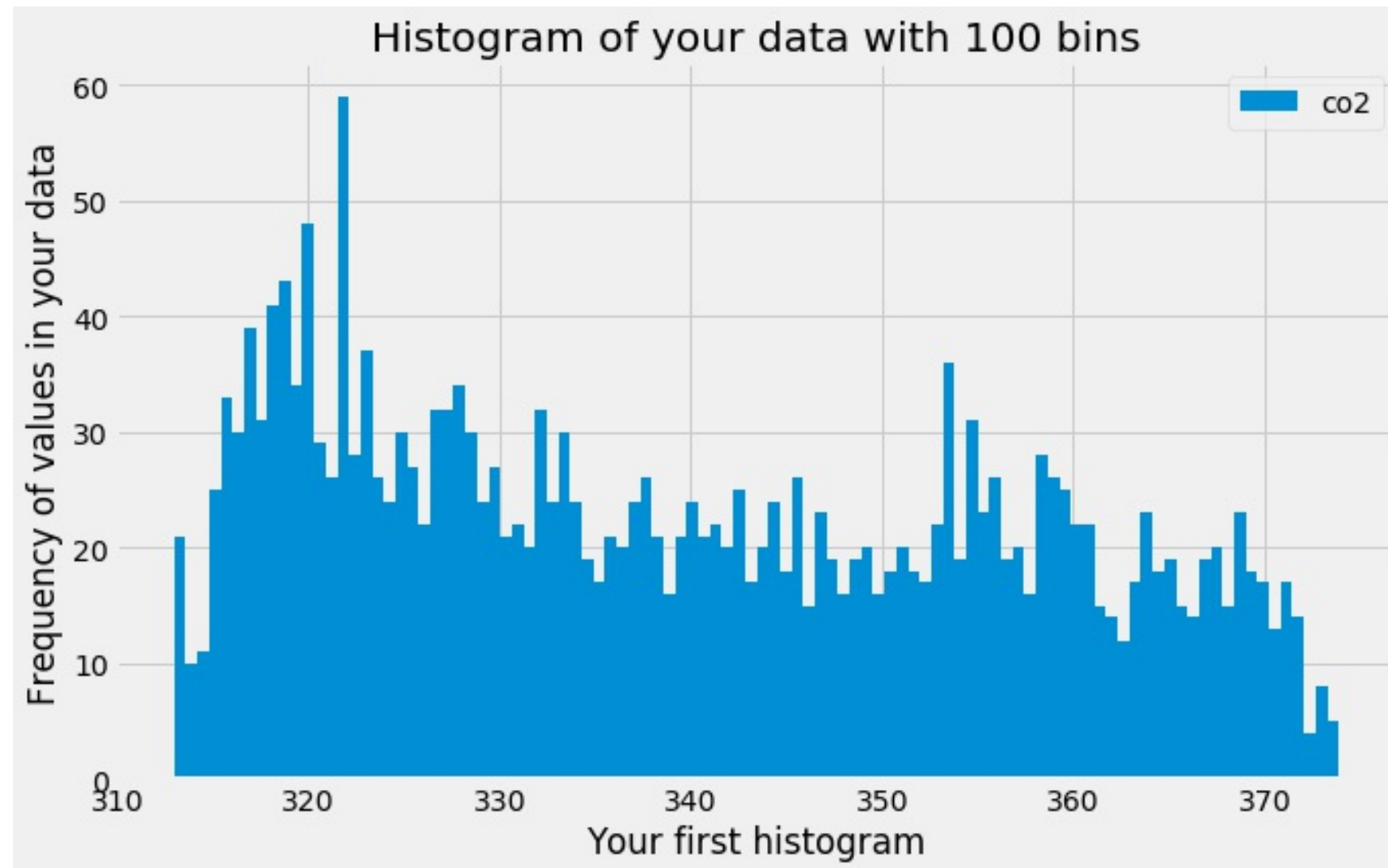
```
In [2]: ax2.set_xlabel('Your first histogram')
```

```
In [3]: ax2.set_ylabel('Frequency of values in your data')
```

```
In [4]: ax2.set_title('Histogram of your data with 100 bins')
```

```
In [5]: plt.show()
```

A histogram plot of the values in the CO2 data





Summarizing your data with density plots

```
In [1]: ax3 = df.plot(kind='density', linewidth=2)

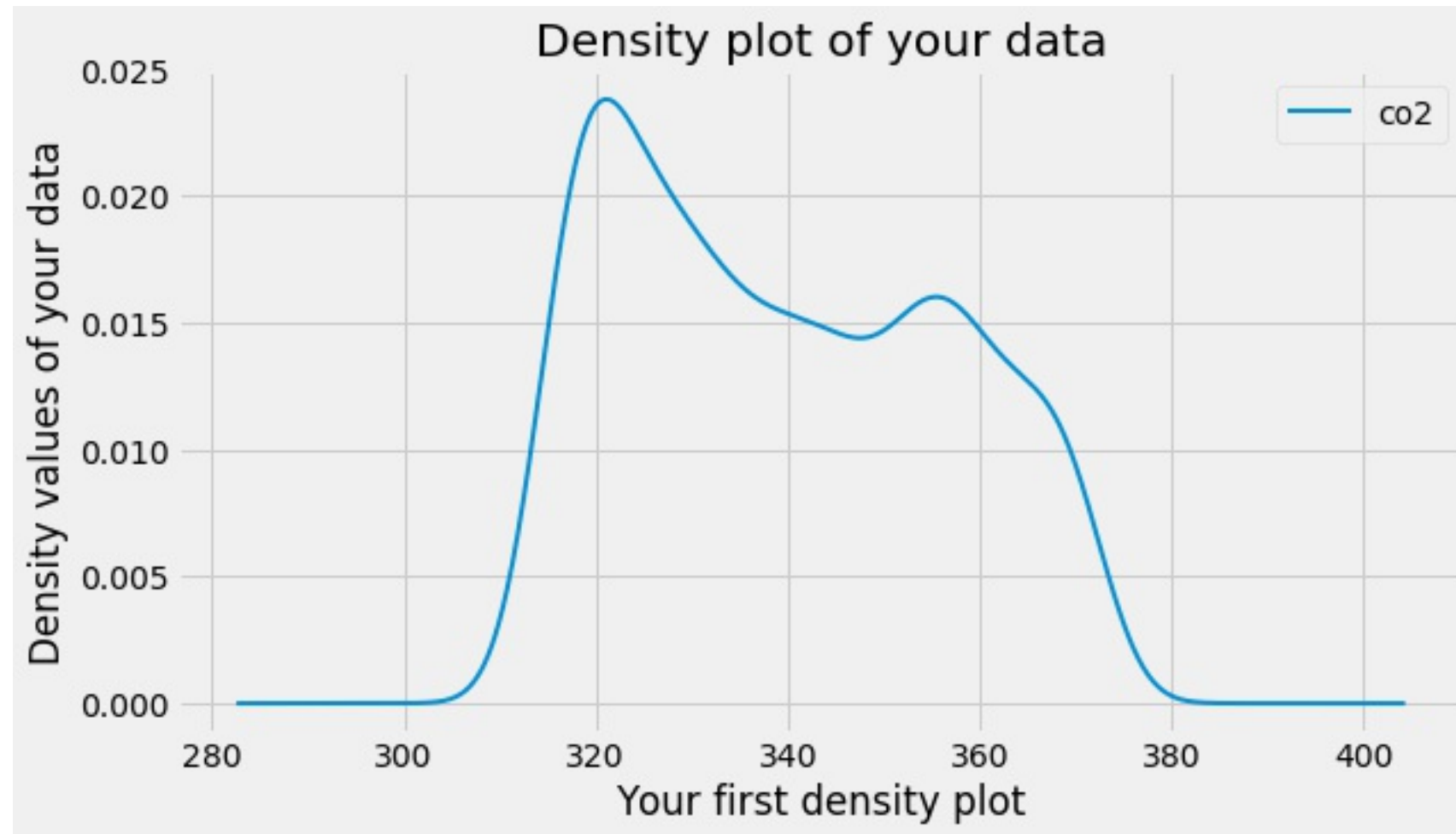
In [2]: ax3.set_xlabel('Your first density plot')

In [3]: ax3.set_ylabel('Density values of your data')

In [4]: ax3.set_title('Density plot of your data')

In [5]: plt.show()
```

A density plot of the values in the CO2 data





VISUALIZING TIME SERIES DATA IN PYTHON

Let's practice!