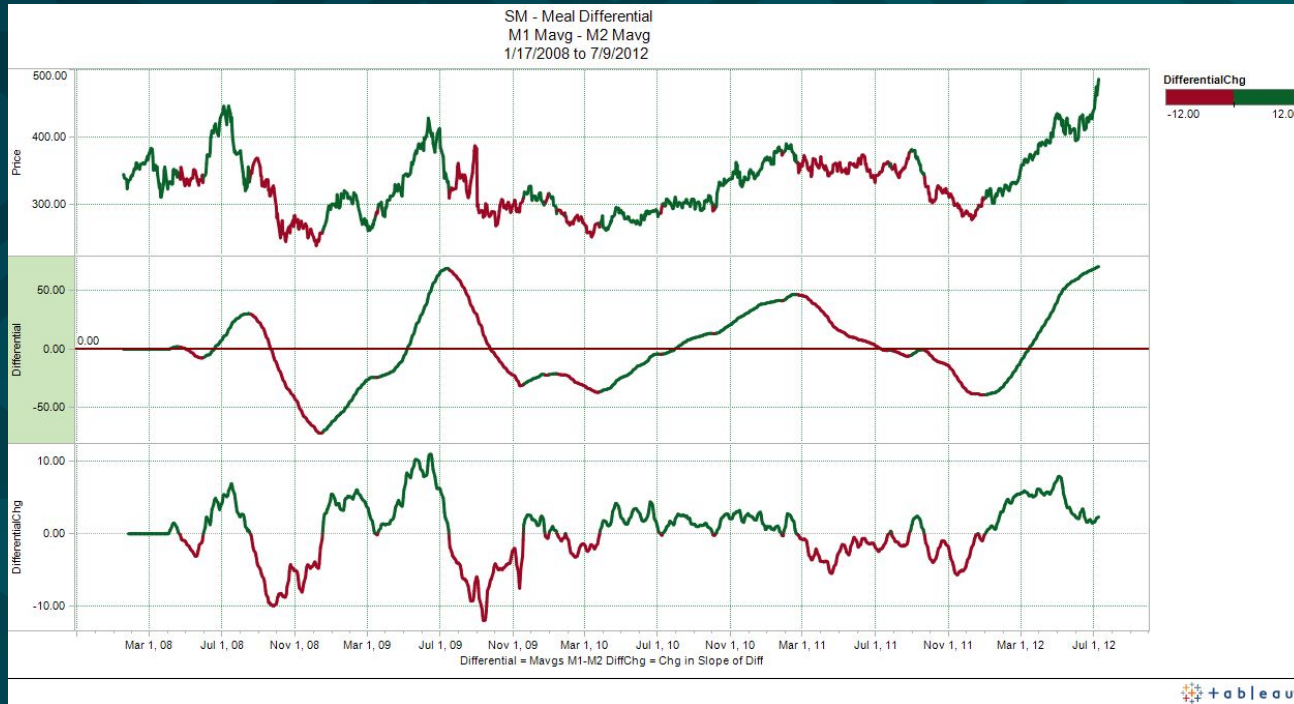
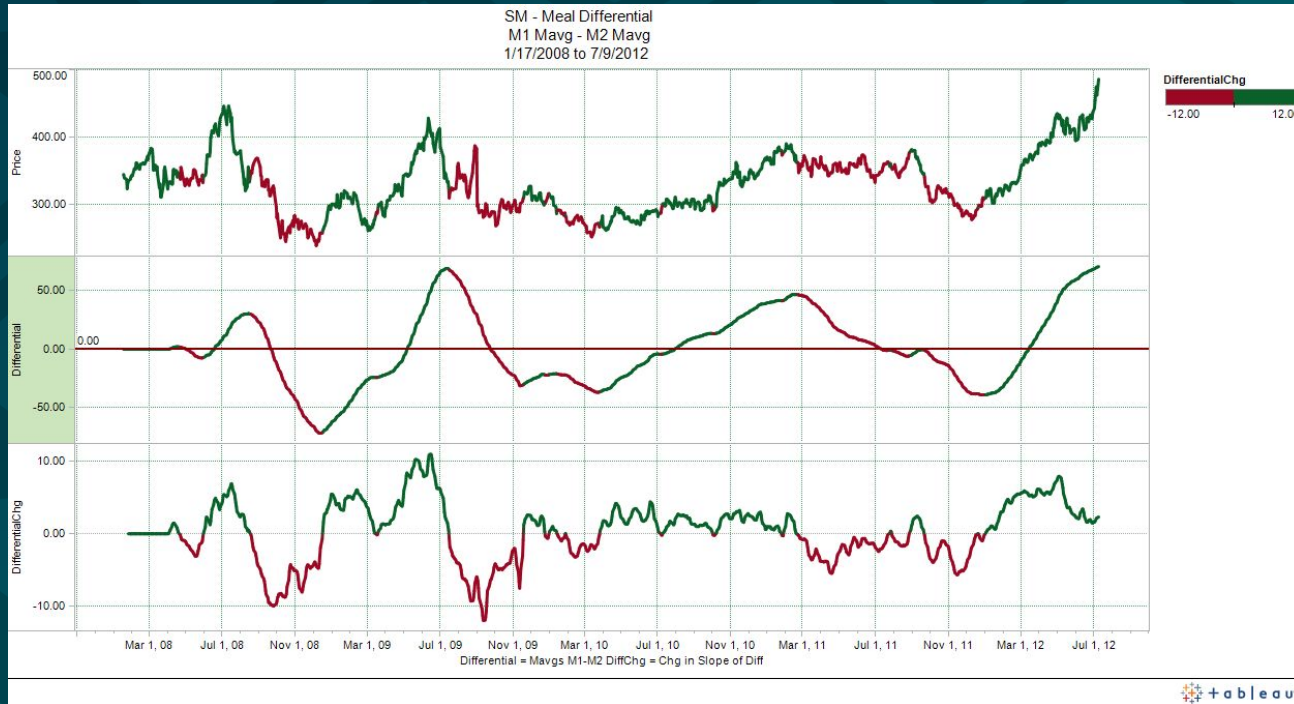


Welcome back!



- What benefits might come from using the middle graph as opposed to the other two?
- Now consider the labels. Let's imagine the price of the top graph represents the price of a meal. What does the green and red coloring represent on each of the graphs? Is there a pattern of similarity that connects all of the graphs?

Part 2



C. Which graph represents the change in the price of the meal over time at any given time?

Answers on the next slide!

Warm-up Answers

(a) The middle graph is easier to read. It is cleaner, showing a clear trend of peaks and troughs, allowing for people to easily gather data. However, it's notable that the data is noticeably less accurate, and may not be sufficient for drawing conclusions.

(b) The green represents when the rate of change of the price of a meal grows, and the red represents when the rate of change of the price of a meal declines.

(c) The first graph is actual data. The middle one represents the (simplified) derivative, and the last one represents the double derivative. So, the middle graph is the one we're looking for as the answer to this question.

The Line Plot

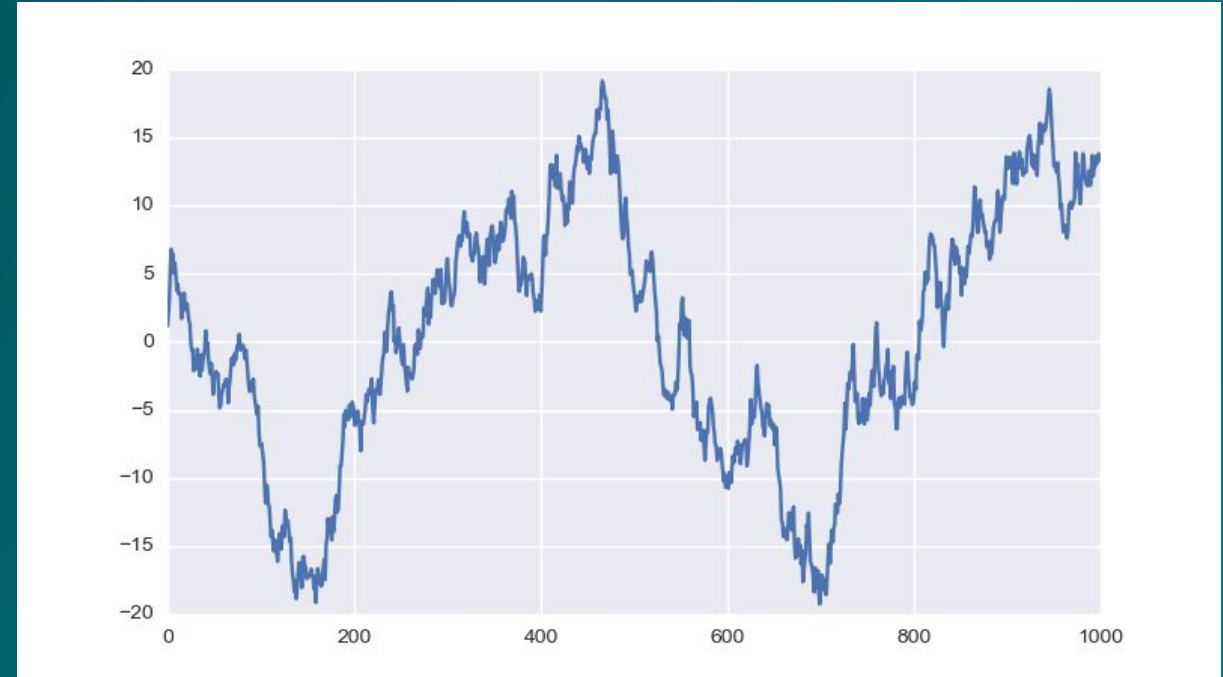
Today, as you could probably tell by the warm-up, we're only going to be focusing on the line plot, an instrumental graph that provides scientists with ways to analyze trends in data.

First, let's go through how line plots can represent temporal trends, trends that focus on the relationship between a variable's change over time.

You already saw an example of a time-based line plot in our warm-up. Let's work a little more with another graph to better figure out what we can learn:

Let's assume that the graph depicts the position of a particle moving back and forth on a single $20\text{ }\mu\text{m}$ line segment (the particle's motion is restricted to 1 plane). The x-axis of the line plot represents time in milliseconds, and the y-axis of the line plot represents μm (micrometers, or one-millionth of a meter) distance from the center of the line segment.

Does this sound familiar? This is a common scenario in physics/mathematics courses. The particle moves erratically and incredibly quickly, as shown by the rapid changes in its position over an extremely short period of time.



The lineplot function

- The lineplot function is part of the seaborn package. Like you've done a million times before now, we call it like this:

```
sns.lineplot(x, y, color)
```

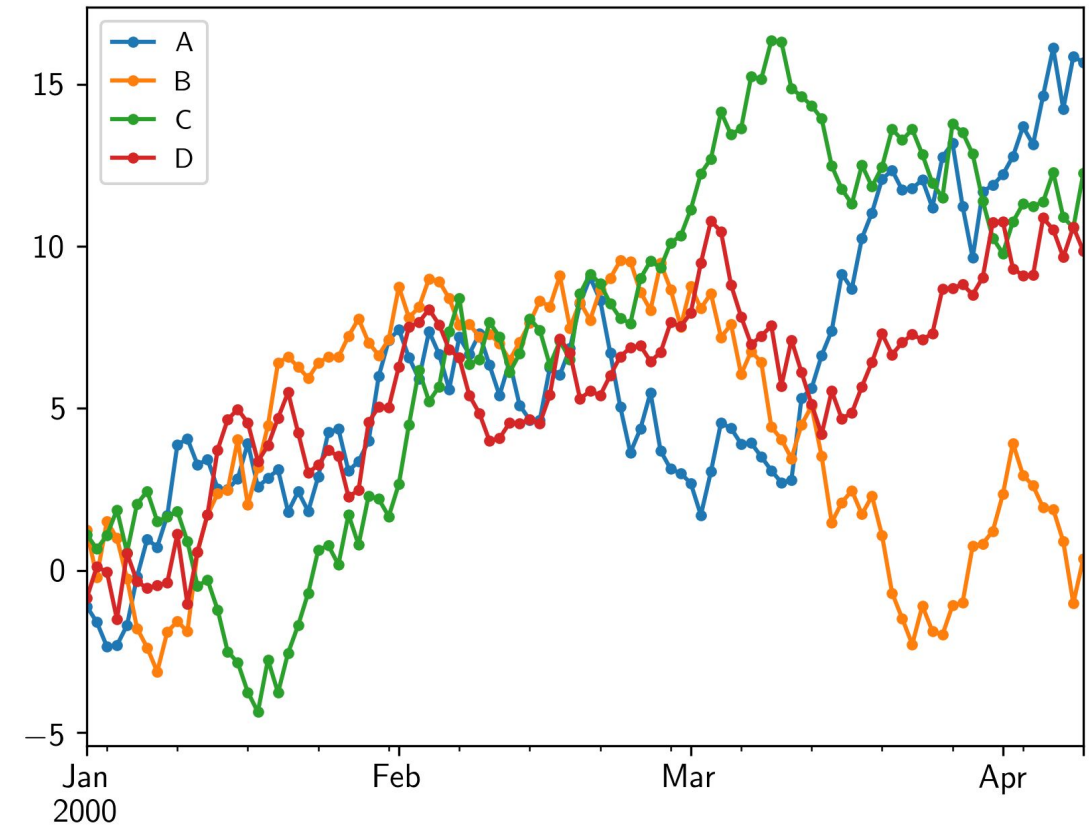
- x in this case must be a column or row of data from the dataset that you wish to represent as the x
- y in this case must be a column or row of data from the dataset that you wish to represent as the y
- color must be set equal to a string denoting the preferred color

If you remember from last week how we created our violin plots with seaborn, the line plot's important parameters are practically identical.

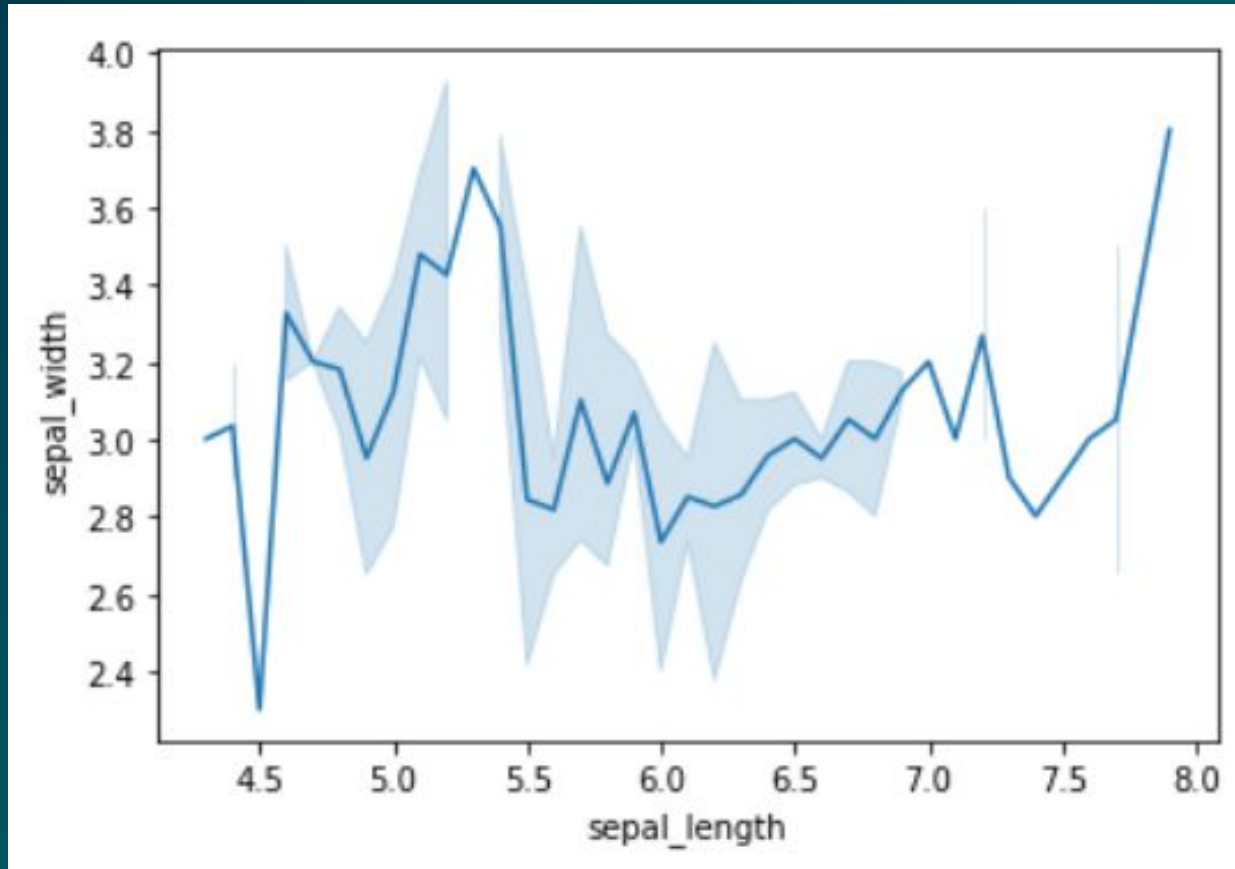
As a quick refresher, what types of variables can be plotted on a line plot? Categorical, or quantitative?

Even if the months might look categorical by the x-axis, both variables can only be quantitative to provide a semi-continuous sequence of data and demonstrate a trend.

With that foundational knowledge out of the way, it's about time we looked at sequence-based line plots instead. Time is a useful axis to plot against, but there's more we can do by plotting two variables against each other.



What if, instead of time, we had another quantitative variable in its place? Take a look at this graph to the left



Sepals are the green leaflike structures at the base of a flower in angiosperm (flowering) plants, intended to protect and support the petals of the flower as it blooms.

The line plot depicts the length and the width of some sample sepals being measured and plotted against each other. What conclusions might you draw from this data? Note: shaded regions of the graph represent the margins of error of the data points that have them.

It's a bit erratic, no? At a glance, the large margins of error and massive spikes across a small range of data suggest that there is no trend. Scientists can test for a number of measured quantitative variables hoping to find new trends, and may come up short just like the example above.

Let's put this into practice.

Give the lineplot function a shot by constructing a line plot titled Annual Income vs Age and Spending Score.

Notice how we're asking for two different strip plots with different categorical variables against the same quantified variable; how do you think we'd go about adding those?

It's easy. Simply calling the function again with the second categorical variable to be on the horizontal axis.

This is how we did it.

We started by defining each column/row of data we wanted to use as a separate variable, making it easier to put into the multiple calls of the function later.

```
# Annual Income vs Age and Spending Score
x = data['Annual Income (k$)']
y = data['Age']
z = data['Spending Score (1-100)']
```

Then, we called the function twice, as we're trying to plot two lines: Age and Spending Score, plotted against Annual Income.

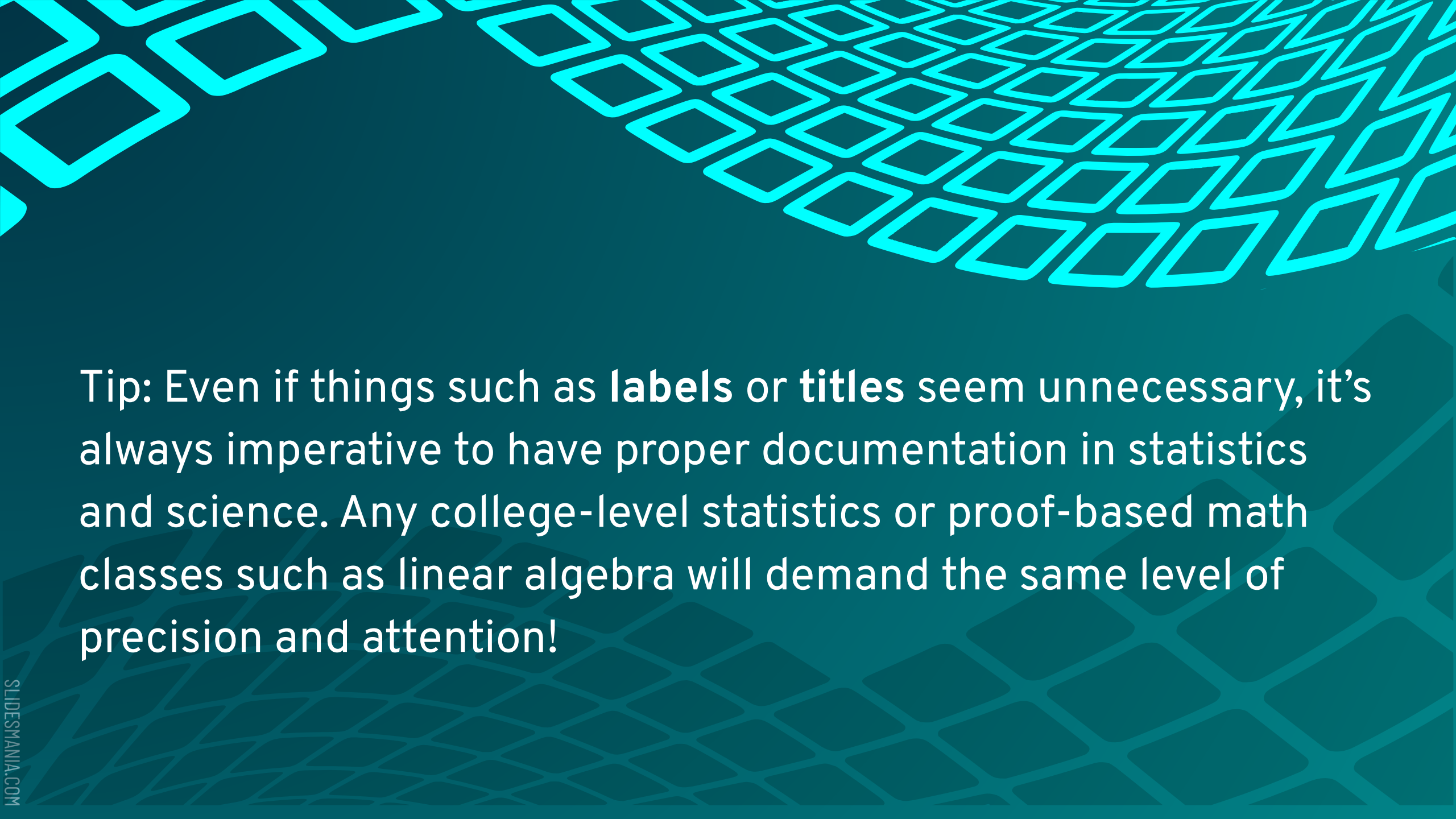
```
sns.lineplot(x, y, color = 'blue')
sns.lineplot(x, z, color = 'pink')
```

Finally, finish it off as usual with proper labeling and show off your work.

```
plt.title('Annual Income vs Age and Spending Score', fontsize = 20)
plt.show()
```

This is how it would look like.





Tip: Even if things such as **labels** or **titles** seem unnecessary, it's always imperative to have proper documentation in statistics and science. Any college-level statistics or proof-based math classes such as linear algebra will demand the same level of precision and attention!

Homework

Craft a line plot showcasing the relationship between Spending Score and Age. Draw conclusions on the nature of this relationship based on the visual data.

You got this!

Helpful Resources

KhanAcademy always has good stuff. If you're more interested in the math and reasoning behind all of the visualizations and code, take a look!

<https://www.khanacademy.org/math/ap-statistics/quantitative-data-ap>

Here's more information for Seaborn line plots in particular.

<https://datagy.io/seaborn-line-plot/>

I've included last week's resources here as well. Help yourself to them if you want a refresher.

The Seaborn website has API documentation that explains all of its features in full detail. Check it out!

<https://seaborn.pydata.org/>

<https://seaborn.pydata.org/tutorial.html>

In addition, GeeksForGeeks has an amazing guide on how to utilize almost every feature of Seaborn.

<https://www.geeksforgeeks.org/python-seaborn-tutorial/>

If you'd prefer video format, this guide (about an hour long) runs through how to use Seaborn. Use the slated video chapters to search for any content you'd like to review or learn more about!

<https://www.youtube.com/watch?v=6GUZXDef2U0>

Thank You!
