


```
from google.colab import files
uploaded = files.upload() # This will prompt you to upload the file
```



Choose files

 No file chosen

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.


Saving archive (5).zip to archive (5) (1).zip

```
import zipfile
import pandas as pd

# Unzip the uploaded file
with zipfile.ZipFile('archive (5).zip', 'r') as zip_ref:
    zip_ref.extractall('extracted_data')

# Read the CSV file from the extracted folder
df = pd.read_csv('extracted_data/Instagram data.csv', encoding='ISO-8859-1')

# Display the first few rows
df.head()
```




	Impressions	From Home	From Hashtags	From Explore	From Other	Saves	Comments	Shares	Likes	Profile Visits	Follows	Caption
0	3920	2586	1028	619	56	98	9	5	162	35	2	Here are some of the most important data visua... #finance #money #business #inv
1	5394	2727	1838	1174	78	194	7	14	224	48	10	Here are some of the best data science project... #healthcare #health #covid #da

```
import pandas as pd

# Load the dataset (make sure to upload the file in Colab)
from google.colab import files
uploaded = files.upload()

# Read the CSV file
df = pd.read_csv('extracted_data/Instagram data.csv', encoding='ISO-8859-1')

# Display first few rows
df.head()
```



Choose files

 No file chosen

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

Saving archive (5).zip to archive (5) (2).zip

```
import re

# Function to clean the text
def clean_text(text):
    text = re.sub(r'http\S+', '', text) # Remove URLs
```

```

text = re.sub(r'^A-Za-z\s', '', text) # Remove non-alphabetic characters
text = text.lower() # Convert to lowercase
return text

```

```

# Apply the cleaning function to the Caption column
df['Cleaned_Caption'] = df['Caption'].apply(clean_text)

```

```

# Display cleaned captions
df[['Caption', 'Cleaned_Caption']].head()

```



	Caption	Cleaned_Caption
0	Here are some of the most important data visua...	here are some of the most important data visua...
1	Here are some of the best data science project...	here are some of the best data science project...
2	Learn how to train a machine learning model an...	learn how to train a machine learning model an...
3	Here's how you can write a Python program to d...	heres how you can write a python program to de...
4	Plotting annotations while visualizing your da...	plotting annotations while visualizing your da...

```

from textblob import TextBlob

```

```

# Function to get the sentiment score
def get_sentiment(text):
    analysis = TextBlob(text)
    return analysis.sentiment.polarity

```

```

# Apply the sentiment analysis
df['Sentiment'] = df['Cleaned_Caption'].apply(get_sentiment)

```

```

# Classify sentiments as Positive, Negative, or Neutral
df['Sentiment_Label'] = df['Sentiment'].apply(lambda x: 'Positive' if x > 0 else ('Negative' if x < 0 else 'Neutral'))

```

```

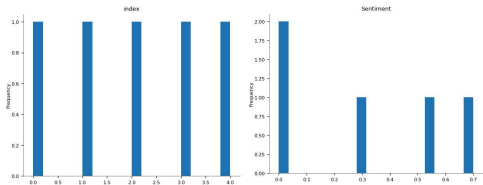
# Show the sentiment analysis results
df[['Cleaned_Caption', 'Sentiment', 'Sentiment_Label']].head()

```

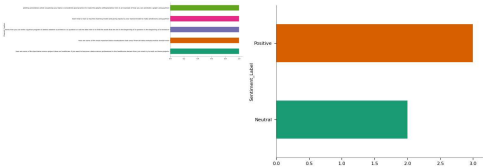


	Cleaned_Caption	Sentiment	Sentiment_Label
0	here are some of the most important data visua...	0.30	Positive
1	here are some of the best data science project...	0.55	Positive
2	learn how to train a machine learning model an...	0.00	Neutral
3	heres how you can write a python program to de...	0.00	Neutral
4	plotting annotations while visualizing your da...	0.70	Positive

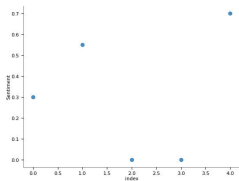
Distributions



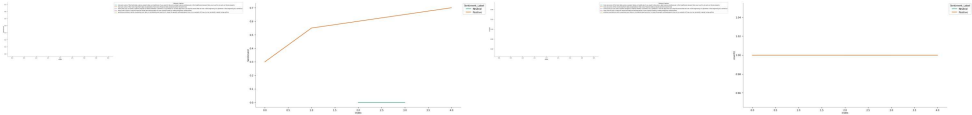
Categorical distributions



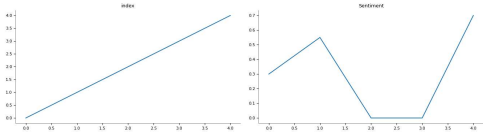
2-d distributions



Time series



Values



2-d categorical distributions



Faceted distributions

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `leg

<string>:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `leg

<string>:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `leg

<string>:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `leg

<string>:5: FutureWarning:

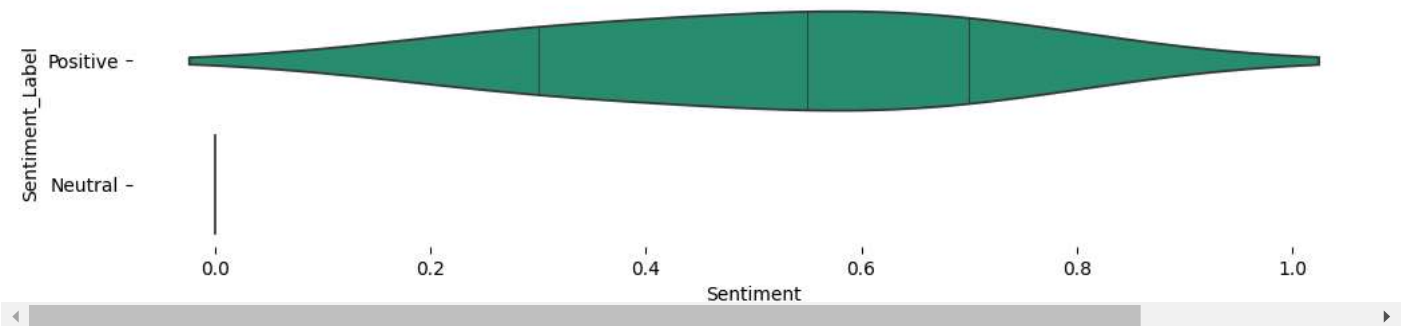
Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `leg

```
from matplotlib import pyplot as plt
import seaborn as sns
figsize = (12, 1.2 * len(_df_15['Sentiment_Label'].unique()))
plt.figure(figsize=figsize)
sns.violinplot(_df_15, x='Sentiment', y='Sentiment_Label', inner='stick', palette='Dark2')
sns.despine(top=True, right=True, bottom=True, left=True)
```

<ipython-input-29-f271a61a91de>:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `legenc

sns.violinplot(\_df\_15, x='Sentiment', y='Sentiment\_Label', inner='stick', palette='Dark2')

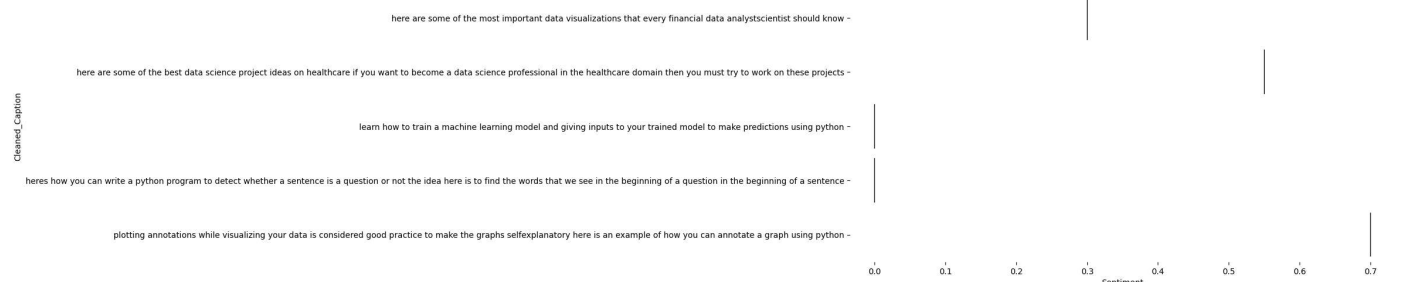


```
from matplotlib import pyplot as plt
import seaborn as sns
figsize = (12, 1.2 * len(_df_14['Cleaned_Caption'].unique()))
plt.figure(figsize=figsize)
sns.violinplot(_df_14, x='Sentiment', y='Cleaned_Caption', inner='stick', palette='Dark2')
sns.despine(top=True, right=True, bottom=True, left=True)
```


<ipython-input-28-831931312578>:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `legenc

sns.violinplot(\_df\_14, x='Sentiment', y='Cleaned\_Caption', inner='stick', palette='Dark2')

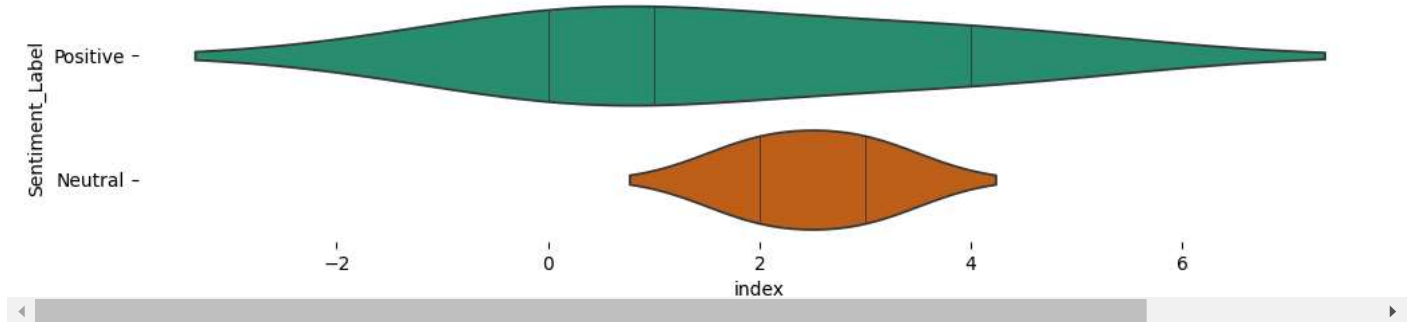


```
from matplotlib import pyplot as plt
import seaborn as sns
figsize = (12, 1.2 * len(_df_13['Sentiment_Label'].unique()))
plt.figure(figsize=figsize)
sns.violinplot(_df_13, x='index', y='Sentiment_Label', inner='stick', palette='Dark2')
sns.despine(top=True, right=True, bottom=True, left=True)
```


 <ipython-input-27-97b0735f6a51>:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `legend`

```
sns.violinplot(_df_13, x='index', y='Sentiment_Label', inner='stick', palette='Dark2')
```

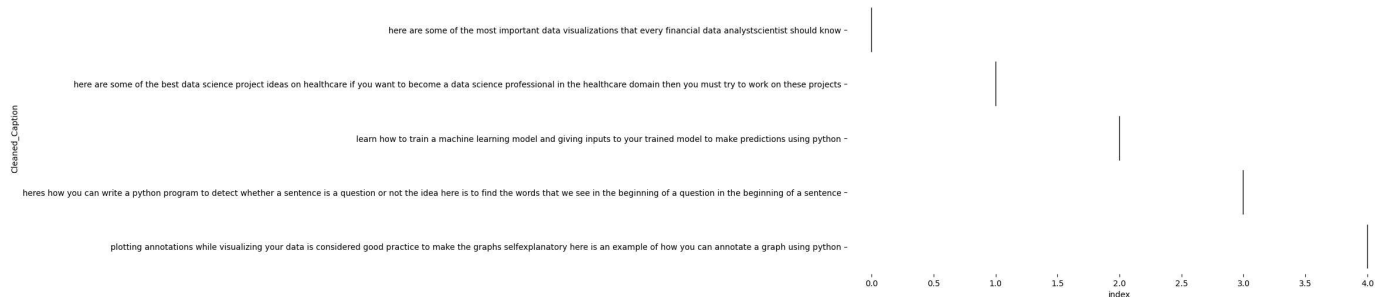


```
from matplotlib import pyplot as plt
import seaborn as sns
figsize = (12, 1.2 * len(_df_12['Cleaned_Caption'].unique()))
plt.figure(figsize=figsize)
sns.violinplot(_df_12, x='index', y='Cleaned_Caption', inner='stick', palette='Dark2')
sns.despine(top=True, right=True, bottom=True, left=True)
```

 <ipython-input-26-654c514fe4ae>:5: FutureWarning:

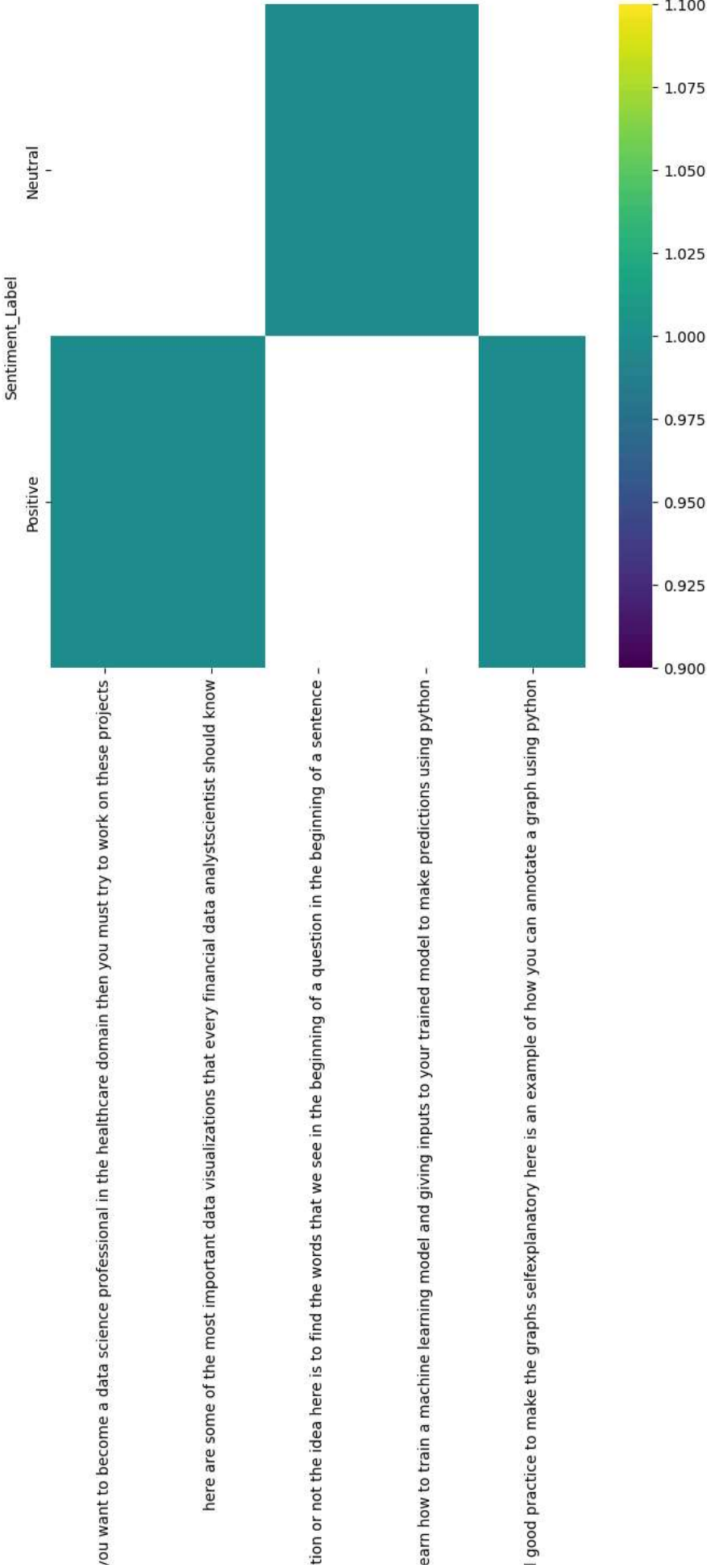
Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `legend`

```
sns.violinplot(_df_12, x='index', y='Cleaned_Caption', inner='stick', palette='Dark2')
```



```
from matplotlib import pyplot as plt
import seaborn as sns
import pandas as pd
plt.subplots(figsize=(8, 8))
df_2dhist = pd.DataFrame({
    x_label: grp['Sentiment_Label'].value_counts()
    for x_label, grp in _df_11.groupby('Cleaned_Caption')
})
sns.heatmap(df_2dhist, cmap='viridis')
plt.xlabel('Cleaned_Caption')
_ = plt.ylabel('Sentiment_Label')
```

↕



here are some of the best data science project ideas on healthcare if y

heres how you can write a python program to detect whether a sentence is a ques

li

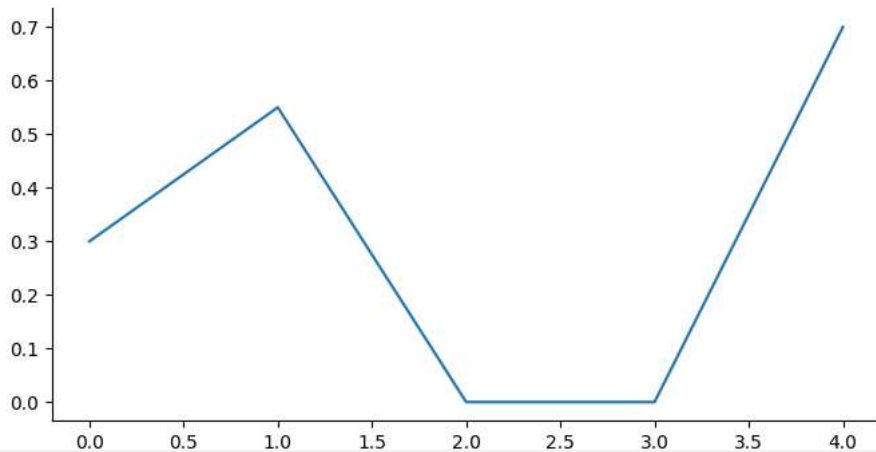
plotting annotations while visualizing your data is considered

Cleaned Caption

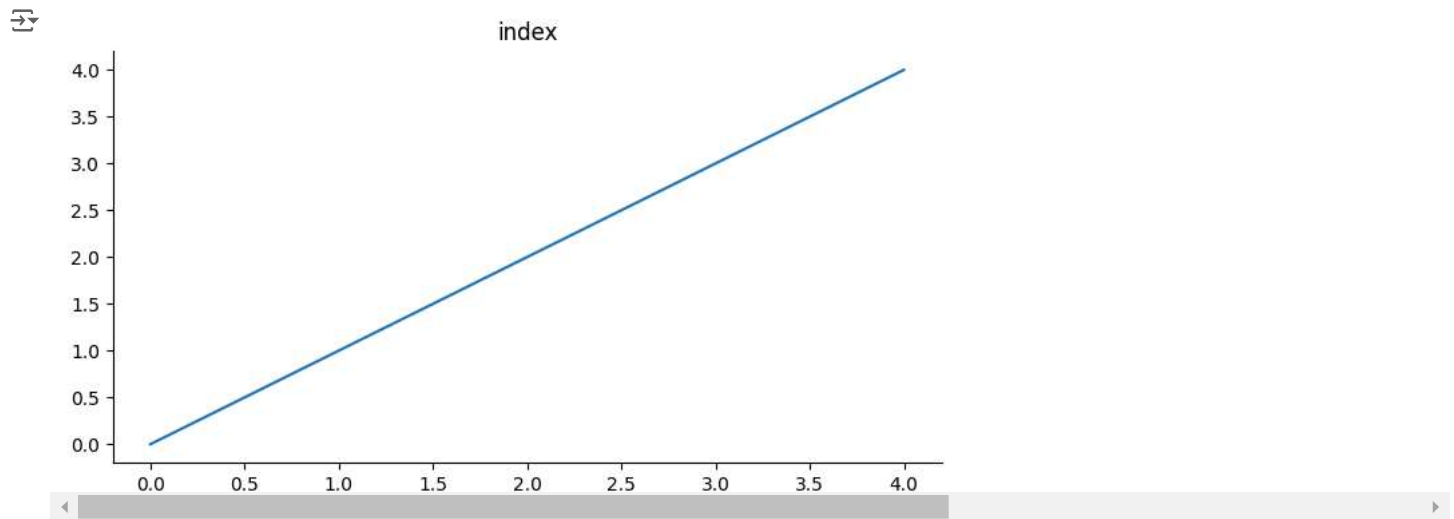
```
from matplotlib import pyplot as plt
_df_10['Sentiment'].plot(kind='line', figsize=(8, 4), title='Sentiment')
plt.gca().spines[['top', 'right']].set_visible(False)
```



Sentiment

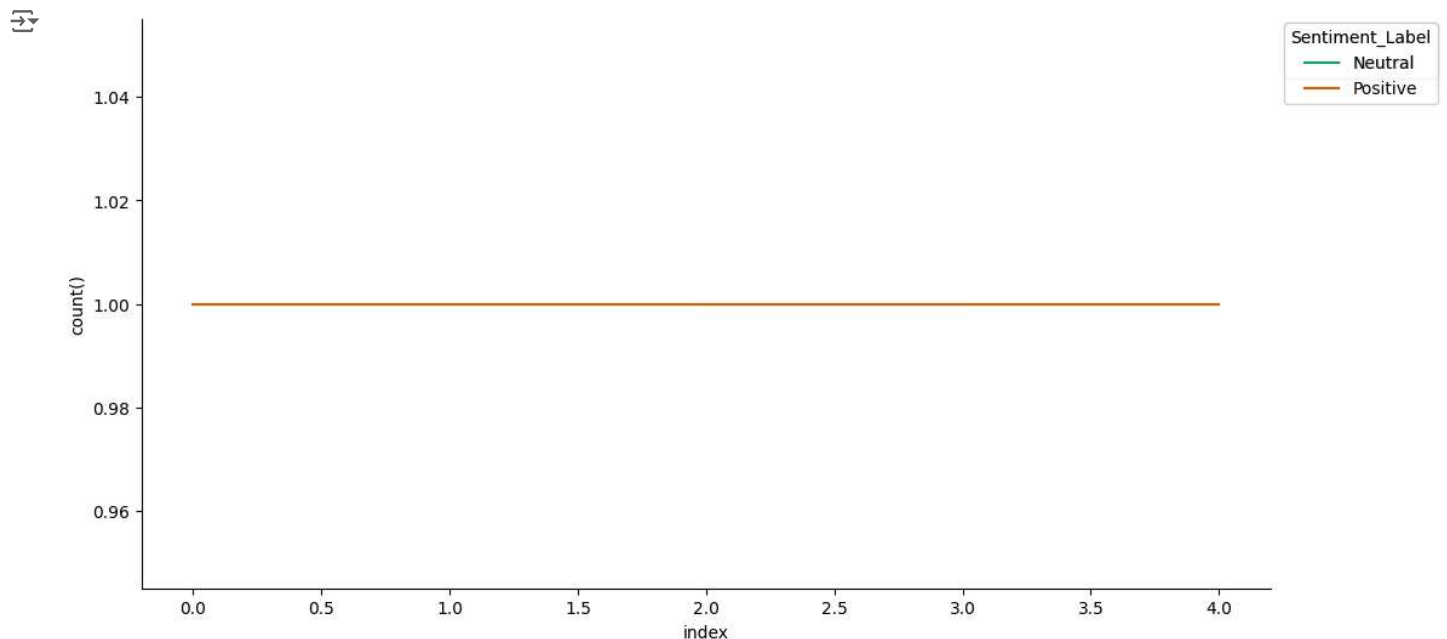


```
from matplotlib import pyplot as plt
_df_9['index'].plot(kind='line', figsize=(8, 4), title='index')
plt.gca().spines[['top', 'right']].set_visible(False)
```



```
from matplotlib import pyplot as plt
import seaborn as sns
def _plot_series(series, series_name, series_index=0):
    palette = list(sns.palettes.mpl_palette('Dark2'))
    counted = (series['index']
               .value_counts()
               .reset_index(name='counts')
               .rename({'index': 'index'}, axis=1)
               .sort_values('index', ascending=True))
    xs = counted['index']
    ys = counted['counts']
    plt.plot(xs, ys, label=series_name, color=palette[series_index % len(palette)])

fig, ax = plt.subplots(figsize=(10, 5.2), layout='constrained')
df_sorted = _df_8.sort_values('index', ascending=True)
for i, (series_name, series) in enumerate(df_sorted.groupby('Sentiment_Label')):
    _plot_series(series, series_name, i)
    fig.legend(title='Sentiment_Label', bbox_to_anchor=(1, 1), loc='upper left')
sns.despine(fig=fig, ax=ax)
plt.xlabel('index')
_ = plt.ylabel('count()')
```



```
from matplotlib import pyplot as plt
import seaborn as sns
def _plot_series(series, series_name, series_index=0):
    palette = list(sns.palettes.mpl_palette('Dark2'))
```



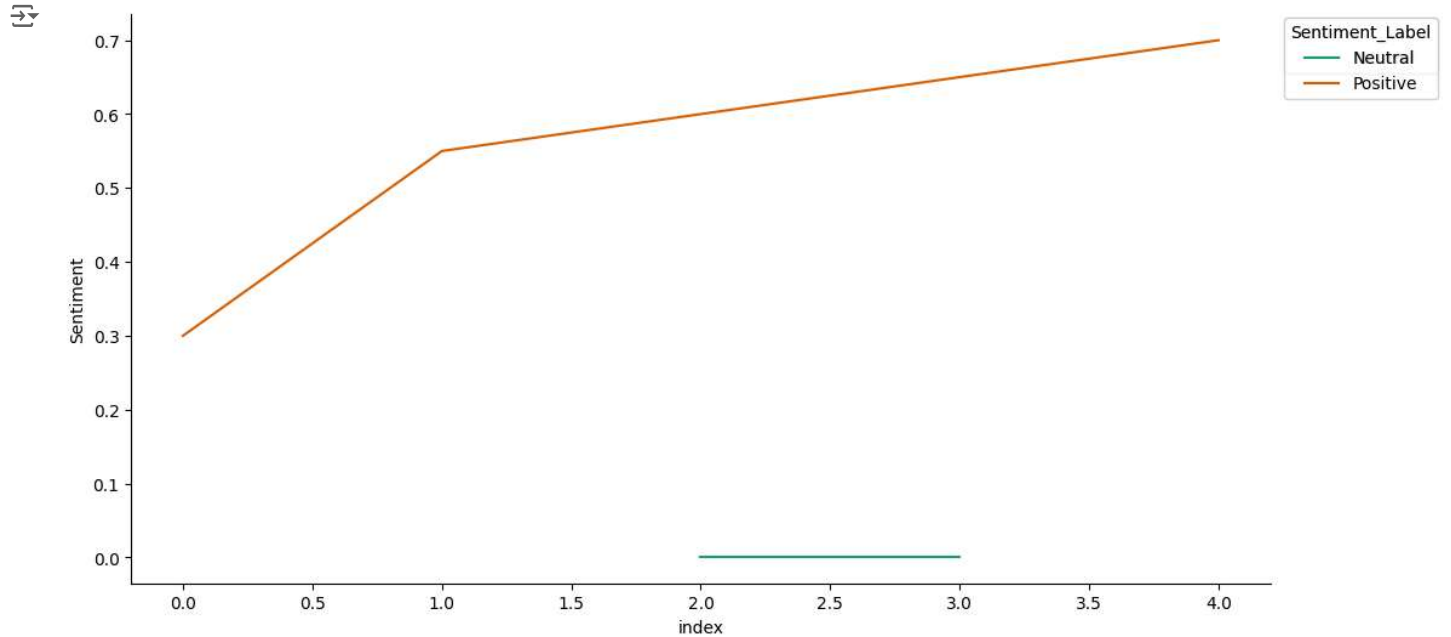
```

xs = series['index']
ys = series['Sentiment']

plt.plot(xs, ys, label=series_name, color=palette[series_index % len(palette)])

fig, ax = plt.subplots(figsize=(10, 5.2), layout='constrained')
df_sorted = _df_6.sort_values('index', ascending=True)
for i, (series_name, series) in enumerate(df_sorted.groupby('Sentiment_Label')):
    _plot_series(series, series_name, i)
    fig.legend(title='Sentiment_Label', bbox_to_anchor=(1, 1), loc='upper left')
sns.despine(fig=fig, ax=ax)
plt.xlabel('index')
_ = plt.ylabel('Sentiment')

```



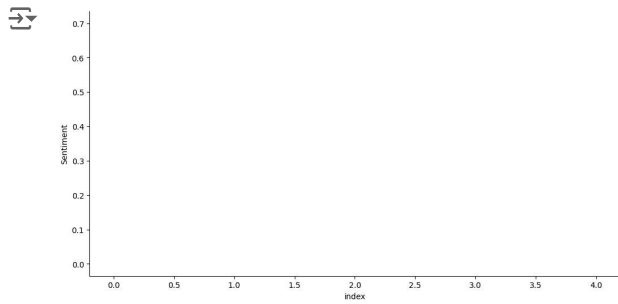
```

from matplotlib import pyplot as plt
import seaborn as sns
def _plot_series(series, series_name, series_index=0):
    palette = list(sns.palettes.mpl_palette('Dark2'))
    xs = series['index']
    ys = series['Sentiment']

    plt.plot(xs, ys, label=series_name, color=palette[series_index % len(palette)])

fig, ax = plt.subplots(figsize=(10, 5.2), layout='constrained')
df_sorted = _df_5.sort_values('index', ascending=True)
for i, (series_name, series) in enumerate(df_sorted.groupby('Cleaned_Caption')):
    _plot_series(series, series_name, i)
    fig.legend(title='Cleaned_Caption', bbox_to_anchor=(1, 1), loc='upper left')
sns.despine(fig=fig, ax=ax)
plt.xlabel('index')
_ = plt.ylabel('Sentiment')

```



Cleaned\_Caption

here are some of the best data science project ideas on healthcare if you want to become a data science professional in the healthcare domain then you must try to work on these projects

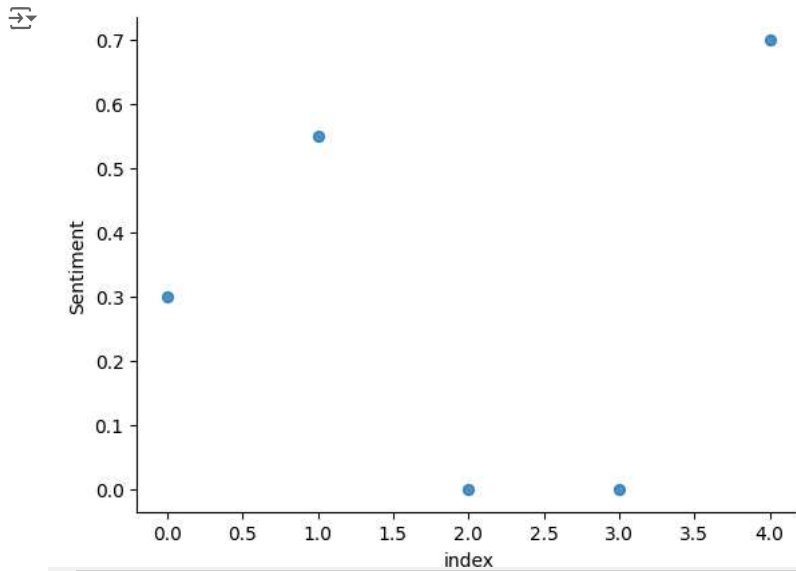
here are some of the most important data visualizations that every financial data analyst/scientist should know

heres how you can write a python program to detect whether a sentence is a question or not the idea here is to find the words that we see in the beginning of a question in the beginning of a sentence

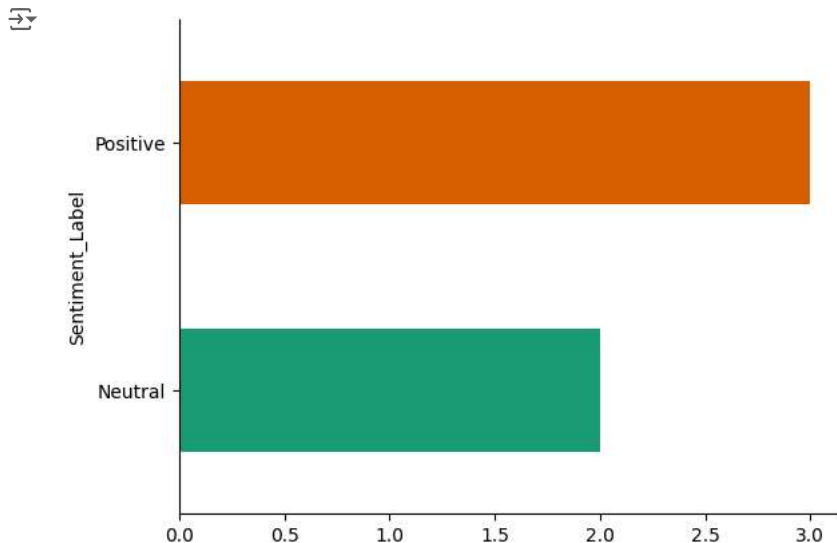
learn how to train a machine learning model and giving inputs to your trained model to make predictions using python

plotting annotations while visualizing your data is considered good practice to make the graphs selfexplanatory here is an example of how you can annotate a graph using python

```
from matplotlib import pyplot as plt
_df4.plot(kind='scatter', x='index', y='Sentiment', s=32, alpha=.8)
plt.gca().spines[['top', 'right']].set_visible(False)
```



```
from matplotlib import pyplot as plt
import seaborn as sns
_df3.groupby('Sentiment_Label').size().plot(kind='barh', color=sns.palettes.mpl_palette('Dark2'))
plt.gca().spines[['top', 'right']].set_visible(False)
```



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