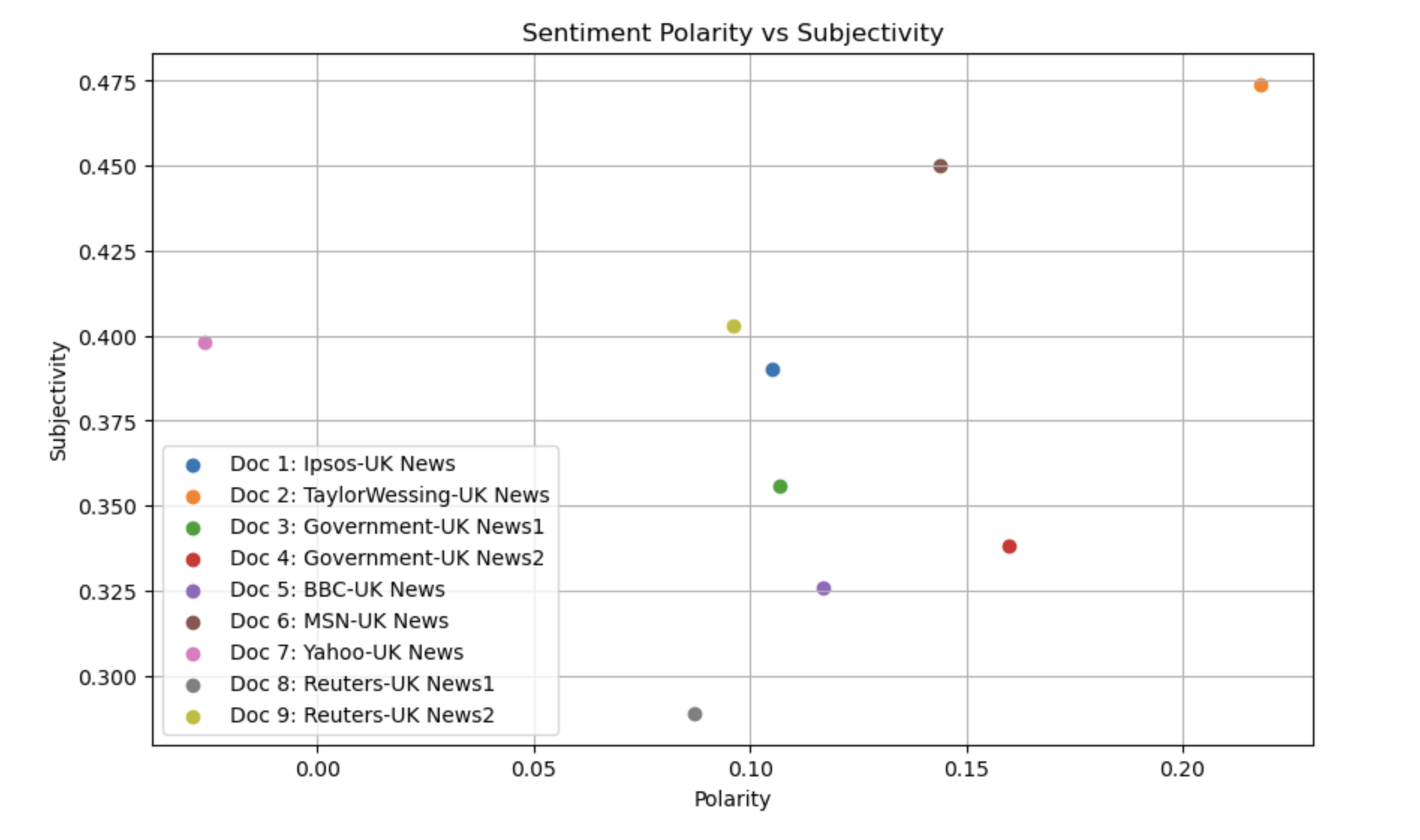
We have successfully extracted sentiment analysis results from all seven notebooks. Here are the polarity and subjectivity values for each:

* Ipsos-UK News: Polarity = 0.105, Subjectivity = 0.390
* Yahoo-UK News: Polarity = 0.218, Subjectivity = 0.474
* TaylorWessing-UK News: Polarity = 0.107, Subjectivity = 0.356
* MSN-UK News: Polarity = 0.160, Subjectivity = 0.338
* Government-UK News1: Polarity = 0.117, Subjectivity = 0.326
* Government-UK News2: Polarity = 0.144, Subjectivity = 0.450
* BBC-UK News: Polarity = -0.026, Subjectivity = 0.398
* Reuters-UK News1:Polarity = 0.087, Subjectivity = 0.289
* Reuters-UK News2:Polarity = 0.096, Subjectivity = 0.403

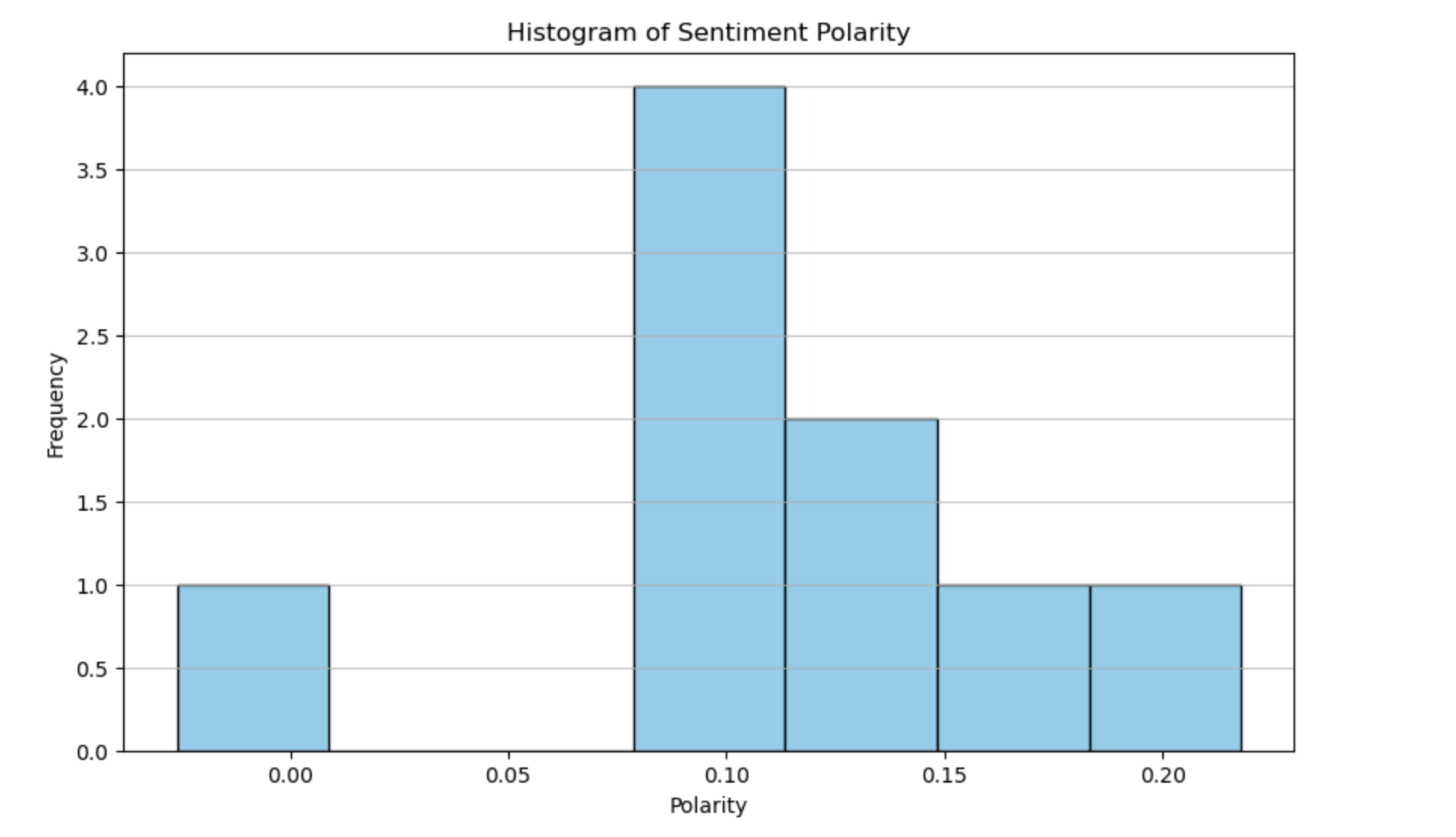
The overall average sentiment across all seven documents related to UK immigration news coverage is as follows:

* Average Polarity: 0.112
* Average Subjectivity: 0.380

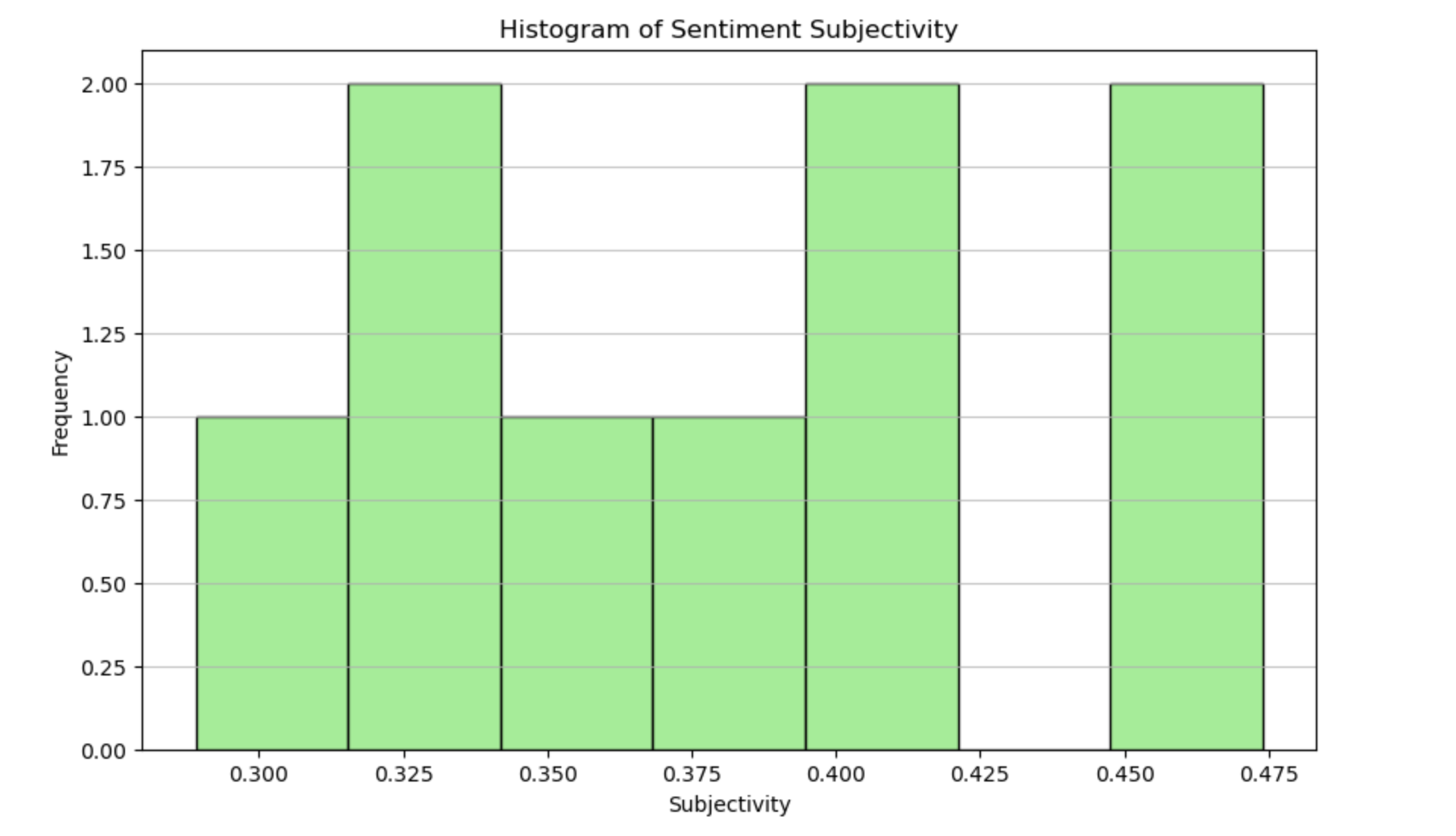
Sentiment Polarity and Subjectivity Scatter Plot



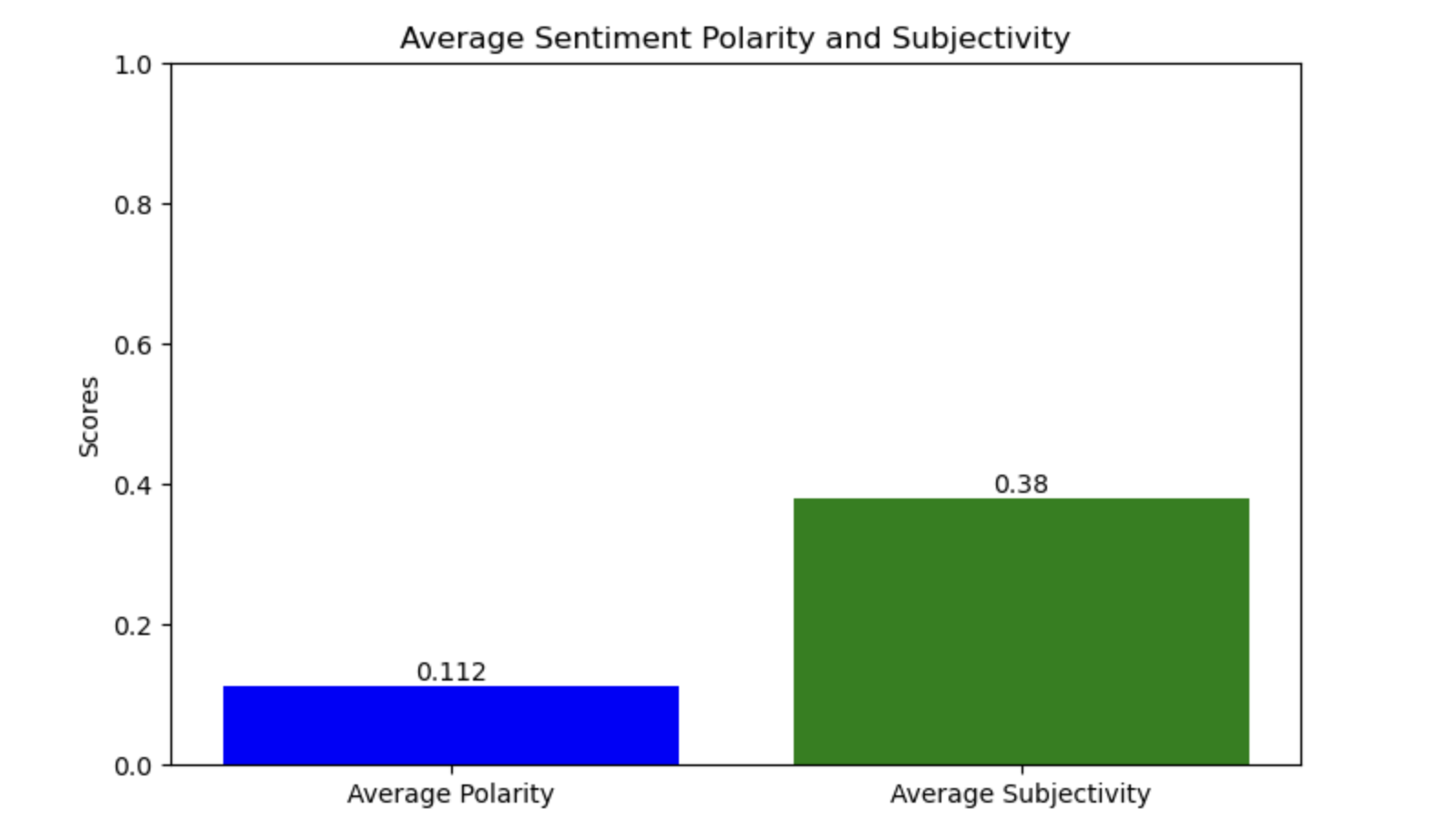
Sentiment Polarity Histogram



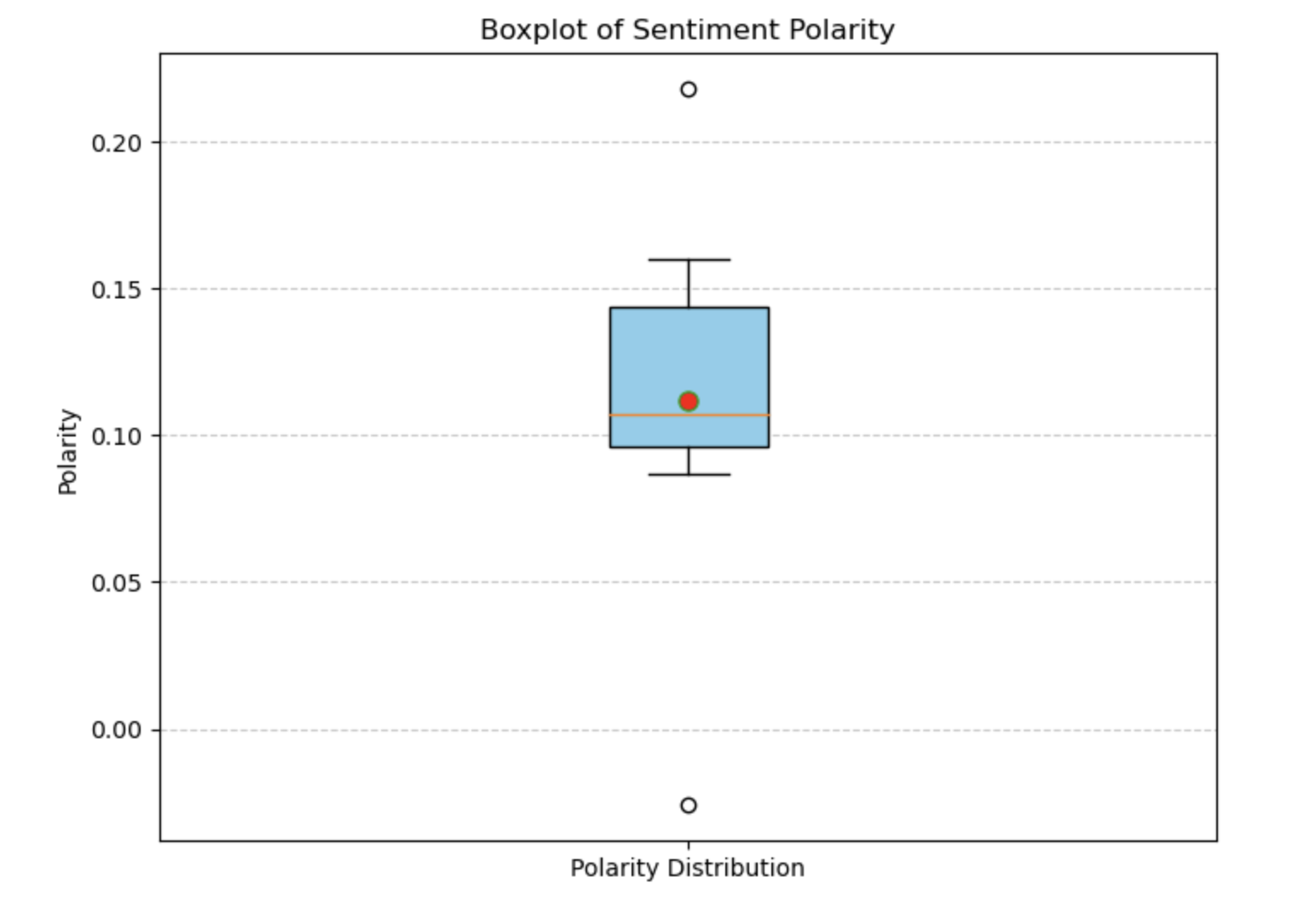
Subjectivity histogram



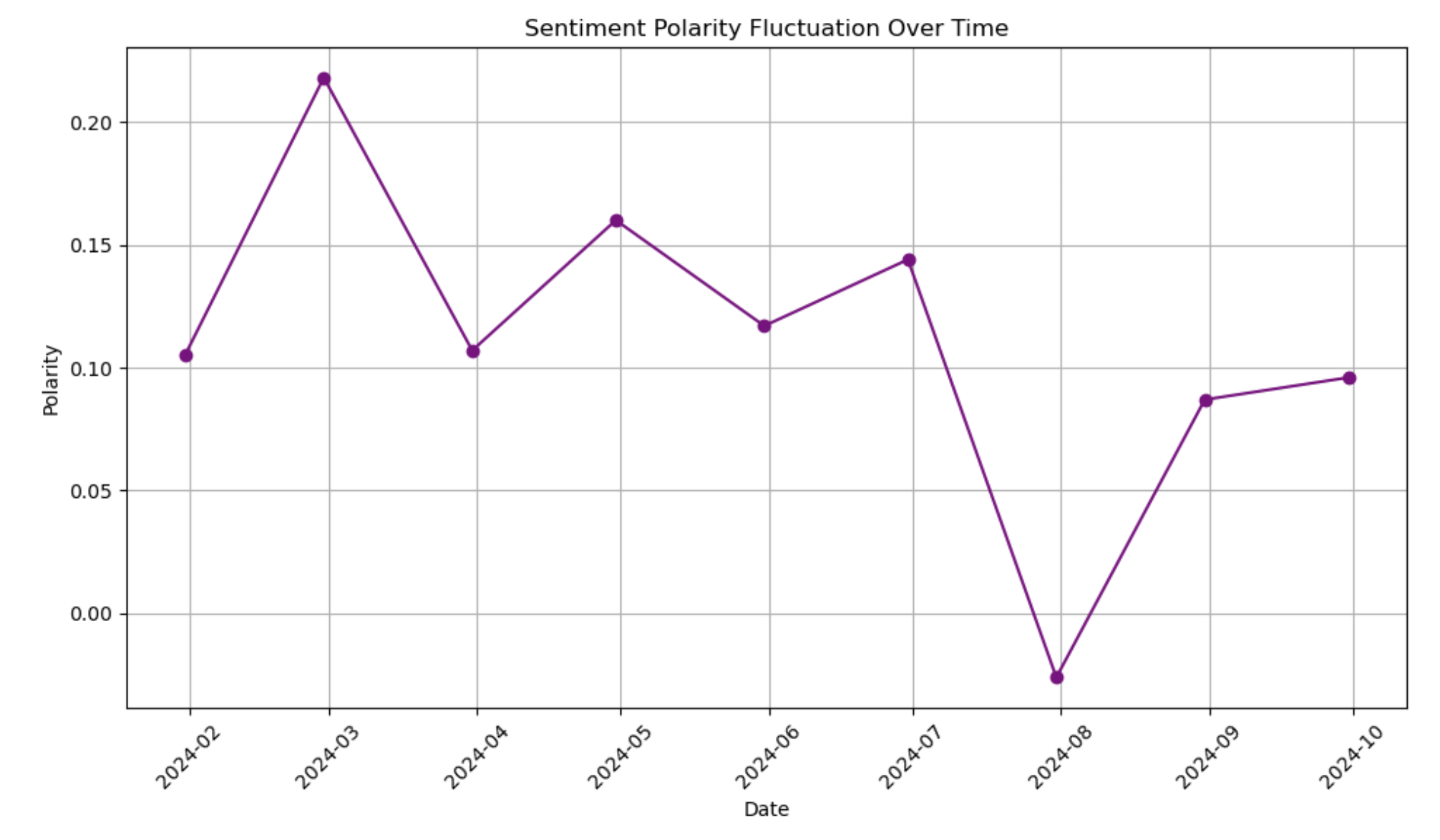
Average sentiment polarity and subjectivity bar chart



Sentiment Polarity Box Plot



Mood swings timeline chart

  
  
  
  
Code

# Load and review the content of the first notebook to understand its structure and data

import nbformat

file\_path = '/Users/zhaolongjiang/Desktop/UK Python/Ipsos-UK News.ipynb'

# Read the notebook

with open(file\_path, 'r', encoding='utf-8') as nb\_file:

nb\_content = nbformat.read(nb\_file, as\_version=4)

# Display the structure of the notebook

nb\_content.cells[:5] # Display the first few cells to understand the structure

# Define a function to extract sentiment analysis results from a notebook

def extract\_sentiment\_from\_nb(nb\_cells):

sentiment\_data = {

"polarity": None,

"subjectivity": None

}

for cell in nb\_cells:

if cell.cell\_type == 'code':

for output in cell.get('outputs', []):

if output.output\_type == 'stream' and 'Polarity' in output.text:

lines = output.text.split('\n')

for line in lines:

if 'Polarity' in line:

parts = line.split(',')

polarity\_part = parts[0].strip()

subjectivity\_part = parts[1].strip()

# Extract numerical values

polarity = float(polarity\_part.split(':')[1].strip())

subjectivity = float(subjectivity\_part.split(':')[1].strip())

sentiment\_data["polarity"] = polarity

sentiment\_data["subjectivity"] = subjectivity

return sentiment\_data

return sentiment\_data

# Extract sentiment analysis result from the first notebook

sentiment\_data\_1 = extract\_sentiment\_from\_nb(nb\_content.cells)

sentiment\_data\_1

# Define a function to read a notebook and extract sentiment analysis results

def read\_and\_extract\_sentiment(file\_path):

with open(file\_path, 'r', encoding='utf-8') as nb\_file:

nb\_content = nbformat.read(nb\_file, as\_version=4)

return extract\_sentiment\_from\_nb(nb\_content.cells)

# File paths for the remaining notebooks

file\_paths = [

'/Users/zhaolongjiang/Desktop/UK Python/TaylorWessing-UK News.ipynb',

'/Users/zhaolongjiang/Desktop/UK Python/Government-UK News1.ipynb',

'/Users/zhaolongjiang/Desktop/UK Python/Government-UK News2.ipynb',

'/Users/zhaolongjiang/Desktop/UK Python/BBC-UK News.ipynb',

'/Users/zhaolongjiang/Desktop/UK Python/MSN-UK News.ipynb',

'/Users/zhaolongjiang/Desktop/UK Python/Yahoo-UK News.ipynb',

'/Users/zhaolongjiang/Desktop/UK Python/Reuters-UK News1.ipynb',

'/Users/zhaolongjiang/Desktop/UK Python/Reuters-UK News2.ipynb'

]

# Extract sentiment data from all notebooks

sentiment\_data\_all = [sentiment\_data\_1] + [read\_and\_extract\_sentiment(fp) for fp in file\_paths]

sentiment\_data\_all

# Calculate the average polarity and subjectivity across all notebooks

average\_polarity = sum([data['polarity'] for data in sentiment\_data\_all]) / len(sentiment\_data\_all)

average\_subjectivity = sum([data['subjectivity'] for data in sentiment\_data\_all]) / len(sentiment\_data\_all)

average\_polarity, average\_subjectivity

import matplotlib.pyplot as plt

# Data preparation

polarities = [0.105, 0.218, 0.107, 0.160, 0.117, 0.144, -0.026, 0.087, 0.096]

subjectivities = [0.390, 0.474, 0.356, 0.338, 0.326, 0.450, 0.398, 0.289, 0.403]

documents = ['Doc 1: Ipsos-UK News', 'Doc 2: TaylorWessing-UK News', 'Doc 3: Government-UK News1',

'Doc 4: Government-UK News2', 'Doc 5: BBC-UK News',

'Doc 6: MSN-UK News', 'Doc 7: Yahoo-UK News',

'Doc 8: Reuters-UK News1', 'Doc 9: Reuters-UK News2']

# Creating the scatter plot for sentiment polarity and subjectivity

plt.figure(figsize=(10, 6))

for i in range(len(polarities)):

plt.scatter(polarities[i], subjectivities[i], label=documents[i])

plt.title('Sentiment Polarity vs Subjectivity')

plt.xlabel('Polarity')

plt.ylabel('Subjectivity')

plt.legend()

plt.grid(True)

plt.show()

import matplotlib.pyplot as plt

# Data preparation

polarities = [0.105, 0.218, 0.107, 0.160, 0.117, 0.144, -0.026, 0.087, 0.096]

# Creating the histogram for sentiment polarity

plt.figure(figsize=(10, 6))

plt.hist(polarities, bins=7, color='skyblue', edgecolor='black')

plt.title('Histogram of Sentiment Polarity')

plt.xlabel('Polarity')

plt.ylabel('Frequency')

plt.grid(axis='y', alpha=0.75)

plt.show()

import matplotlib.pyplot as plt

# Data preparation

subjectivities = [0.390, 0.474, 0.356, 0.338, 0.326, 0.450, 0.398, 0.289, 0.403]

# Creating the histogram for sentiment subjectivity

plt.figure(figsize=(10, 6))

plt.hist(subjectivities, bins=7, color='lightgreen', edgecolor='black')

plt.title('Histogram of Sentiment Subjectivity')

plt.xlabel('Subjectivity')

plt.ylabel('Frequency')

plt.grid(axis='y', alpha=0.75)

plt.show()

import matplotlib.pyplot as plt

# Pre-calculated average polarity and subjectivity

average\_polarity = 0.112

average\_subjectivity = 0.380

# Data for plotting

labels = ['Average Polarity', 'Average Subjectivity']

values = [average\_polarity, average\_subjectivity]

# Creating the bar chart

plt.figure(figsize=(8, 5))

bars = plt.bar(labels, values, color=['blue', 'green'])

# Adding value labels on top of each bar

for bar in bars:

yval = bar.get\_height()

plt.text(bar.get\_x() + bar.get\_width() / 2, yval, round(yval, 3), ha='center', va='bottom')

plt.title('Average Sentiment Polarity and Subjectivity')

plt.ylabel('Scores')

plt.ylim(0, 1) # Assuming subjectivity is the highest value and it can't be more than 1

plt.show()

import matplotlib.pyplot as plt

# Pre-defined sentiment polarities

polarities = [0.105, 0.218, 0.107, 0.160, 0.117, 0.144, -0.026, 0.087, 0.096]

# Creating the boxplot

plt.figure(figsize=(8, 6))

plt.boxplot(polarities, vert=True, patch\_artist=True, showmeans=True,

boxprops=dict(facecolor='skyblue'),

meanprops=dict(marker='o', markerfacecolor='red', markersize=8))

plt.title('Boxplot of Sentiment Polarity')

plt.ylabel('Polarity')

plt.xticks([1], ['Polarity Distribution'])

plt.grid(axis='y', linestyle='--', alpha=0.7)

plt.show()

import matplotlib.pyplot as plt

import pandas as pd

# Simulated dates for each document. In a real scenario, these would be the actual dates associated with each document.

dates = pd.date\_range(start="2024-01-01", periods=len(polarities), freq='M')

# Creating a DataFrame for easier plotting

data = pd.DataFrame({'Date': dates, 'Polarity': polarities})

# Creating the line plot

plt.figure(figsize=(10, 6))

plt.plot(data['Date'], data['Polarity'], marker='o', linestyle='-', color='purple')

plt.title('Sentiment Polarity Fluctuation Over Time')

plt.xlabel('Date')

plt.ylabel('Polarity')

plt.grid(True)

plt.xticks(rotation=45) # Rotate date labels for better readability

plt.tight\_layout() # Adjust layout to make room for the rotated date labels

plt.show()