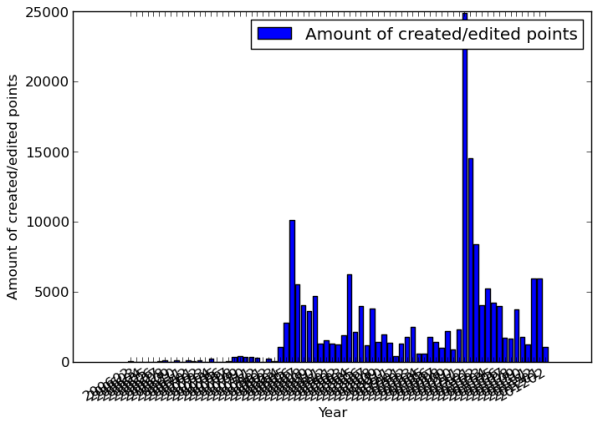
matplotlib: how to prevent x-axis labels from overlapping each other

Ask Question

30

16

I'm generating a bar-chart with matplotlib. It all works well but I can't figure out how to prevent the labels of the x-axis from overlapping each other. Here an example:



Here is some sample SQL for a postgres 9.1 database:

```
drop table if exists mytable;
create table mytable(id bigint, version smalli
insert into mytable(id, version, date_from) va

('4084036', '1', '2006-12-22 22:46:35'),
('4084938', '1', '2006-12-23 16:19:13'),
('4084938', '2', '2006-12-23 16:20:23'),
('4084939', '1', '2006-12-23 16:29:14'),
('4084954', '1', '2006-12-23 16:28:28'),
('4250653', '1', '2007-02-12 21:58:53'),
('4250657', '1', '2007-03-12 21:58:53')
;
```

And this is my python-script:

```
# -*- coding: utf-8 -*-
#!/usr/bin/python2.7
import psycopg2
import matplotlib.pyplot as plt
fig = plt.figure()

# for savefig()
import pylab

###
### Connect to database with psycopg2
###
```

```
try:
  conn_string="dbname='x' user='y' host='z'
  print "Connecting to database\n->%s" % (cc

  conn = psycopg2.connect(conn_string)
  print "Connection to database was establis
except:
  print "Connection to database failed"

###
### Execute SQL query
###

# New cursor method for sql
cur = conn.cursor()

# Execute SQL query. For more than one row u
try:
  cur.execute("""

-- In which year/month have these points bee
-- Need 'yyyymm' because I only need Months
```

Is there a way how I can prevent the labels from overlapping each other? Ideally in an automatic way, because I can't predict the amount of bars.

python matplotlib bar-chart

share improve this question

edited Nov 23 '12 at 17:34

asked Nov 22 '12 at 15:05

zehpunktbarron
393 2 7 21

1 Why not just add fewer ticks? Matplotlib will automatically find a reasonable number of ticks. As it is, you're overriding the automated positioning of the ticks. – Joe Kington Nov 22 '12 at 17:25

Thank you. Could you explain or paste a short snippet how to add fewer ticks in my code. Best whishes – zehpunktbarron Nov 22 '12 at 17:44

comment out the line that says `plt.xticks(range(len(count)), (year))` – Paul H Nov 22 '12 at 23:13

If I do so the matplotlib default-values are plotted (-10, 0, 10, 20, 30, ...). But I need my (string) values from my database-query. Any other suggestion on how to show for example only every 4th axis-label?– zehpunktbarron Nov 23 '12 at 9:48

2 also `matplotlib.dates.datestring2num()` should be useful here – Paul H Nov 23 '12 at 16:57

show **3** more comments

4 Answers

active oldest votes

10

Edit 2014-09-30

pandas now has a `read_sql` function. You definitely want to use that instead.

Original Answer

Here's how you should convert your date string into real datetime objects:

```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
data_tuples = [
    ('4084036', '1', '2006-12-22 22:46:35'),
    ('4084938', '1', '2006-12-23 16:19:13'),
    ('4084938', '2', '2006-12-23 16:20:23'),
    ('4084939', '1', '2006-12-23 16:29:14'),
    ('4084954', '1', '2006-12-23 16:28:28'),
    ('4250653', '1', '2007-02-12 21:58:53'),
    ('4250657', '1', '2007-03-12 21:58:53')]
datatypes = [('col1', 'i4'), ('col2', 'i4'), (
data = np.array(data_tuples, dtype=datatypes)
col1 = data['col1']
dates = mdates.num2date(mdates.datestr2num(dat
fig, ax1 = plt.subplots()
ax1.bar(dates, col1)
fig.autofmt_xdate()
```

Getting a simple list of tuples out of your database cursor should be as simple as...

```
data_tuples = []
for row in cursor:
    data_tuples.append(row)
```

However, I posted a version of a function that I use to take db cursors directly to record arrays or pandas dataframes here: [How to convert SQL Query result to PANDAS Data Structure?]

Hopefully that helps too.

share
improve this answer

edited May 23 '17 at 11:47
Community ♦
1 1

answered Nov 23 '12 at 19:37

Paul H
28.1k 8 94 103

-1; I'm sure that this addresses the question in some indirect way that's just not immediately obvious to me, but it ought to spell that out. As it is, this seems completely unrelated to the question asked. You don't show any images or even mention graphs or labels. – Mark Amery Jan 8 at 11:37

@MarkAmery I think you've mis-read both the answer and the question then. My question takes the list of data tuple, converts the datestrings to proper dates, plots them, and formats the x-axis to be legible. The OP asked: *It all works well but I can't figure out how to prevent the labels of the x-axis from overlapping each other* and decided this in fact*did* answer the question. – Paul H Jan 8 at 15:18

1 I haven't misread either. The issue is not that you haven't *in fact* solved the OP's issue; it's that you've done so in a way that, to a new reader arriving at the question, doesn't *appear* to be related to the OP's issue at all. The issue is that `matplotlib` doesn't know how to sensibly draw labels for an axis if the values are strings, and so that x-axis values need to be converted into a type that it can work with... but the answer doesn't *say* that, or explain itself at all, which makes it unhelpful to anyone other than the original asker (and limits its usefulness even to them). – Mark Amery Jan 8 at 15:22

The answer is essentially *Here's how you should convert your date string into real datetime objects*<does that with nearly plain English python code>. Sorry that's not clear enough. – Paul H Jan 8 at 15:27

Right, and the *question* is "how to prevent x-axis labels from overlapping?" There's no obvious relation between that question and parsing dates. – Mark Amery Jan 8 at 15:30

show **2** more comments

26

I think you're confused on a few points about how matplotlib handles dates.

You're not actually plotting dates, at the moment. You're plotting things on the x-axis with `[0,1,2,...]` and then manually labeling every point with a string representation of the date.

Matplotlib will automatically position ticks. However, you're over-riding matplotlib's tick positioning functionality (Using `xticks` is

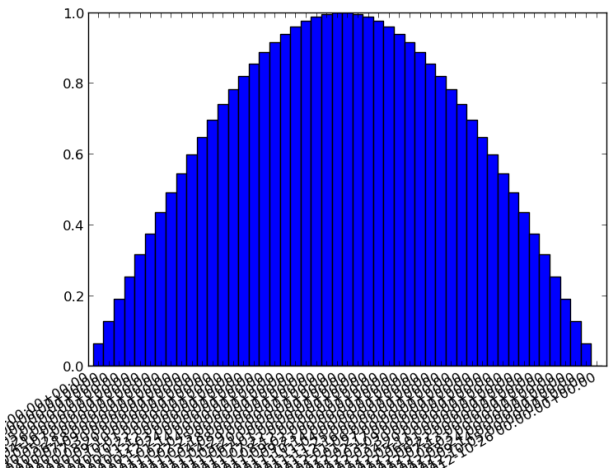
basically saying: "I want ticks in exactly these positions".)

At the moment, you'll get ticks at `[10, 20, 30, ...]` if matplotlib automatically positions them. However, these will correspond to the values that you used to plot them, not the dates (which you didn't use when plotting).

You probably want to actually plot things using dates.

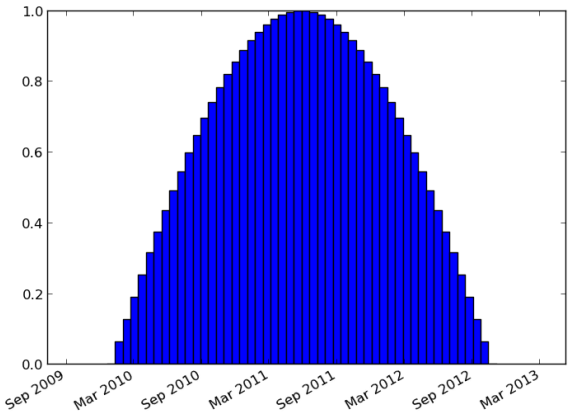
Currently, you're doing something like this:

```
import datetime as dt
import matplotlib.dates as mdates
import numpy as np
import matplotlib.pyplot as plt

# Generate a series of dates (these are in mat
dates = mdates.drange(dt.datetime(2010, 01, 01
                      dt.timedelta(weeks=3))

# Create some data for the y-axis
counts = np.sin(np.linspace(0, np.pi, dates.si

# Set up the axes and figure
fig, ax = plt.subplots()

# Make a bar plot, ignoring the date values
ax.bar(np.arange(counts.size), counts, align='

# Force matplotlib to place a tick at every ba
datelabels = mdates.num2date(dates) # Go back
ax.set(xticks=np.arange(dates.size), xticklabe

# Make space for and rotate the x-axis tick la
fig.autofmt_xdate()

plt.show()
```



Instead, try something like this:

```
import datetime as dt
import matplotlib.dates as mdates
import numpy as np
import matplotlib.pyplot as plt

# Generate a series of dates (these are in mat
dates = mdates.drange(dt.datetime(2010, 01, 01
                     dt.timedelta(weeks=3))

# Create some data for the y-axis
counts = np.sin(np.linspace(0, np.pi, dates.si

# Set up the axes and figure
fig, ax = plt.subplots()

# By default, the bars will have a width of 0.
# them quite a bit wider, so we'll make them t
# the dates. (To use the exact code below, you
# of datetimes into matplotlib's float-based d
# Use "dates = mdates.date2num(dates)" to conv
width = np.diff(dates).min()

# Make a bar plot. Note that I'm using "dates"
# "counts" against x-values of [0,1,2...]
ax.bar(dates, counts, align='center', width=wi

# Tell matplotlib to interpret the x-axis valu
ax.xaxis_date()

# Make space for and rotate the x-axis tick la
fig.autofmt_xdate()

plt.show()
```



share
improve this answer

answered Nov 23 '12 at 0:10

Joe Kington
170k 33 414 392

Thank your for answering. This works if I had dates.
But my tick-values are not formatted as date-values.

They are coming from my db formatted as strings.

So is there another way of using my code but only show for example every 4th tick on the x-axis? – zehpunktbarron Nov 23 '12 at 9:41

@C.B. If your data are dates, you should convert them to be date in your code, not string. See the datetime module which does that trivially. – Sardathrion Mar 19 '15 at 9:13

I think this is a better answer. – Hamdi May 28 '17 at 21:37

add a comment

5

As for your question on how to show only every 4th tick (for example) on the xaxis, you can do this:

```
import matplotlib.ticker as mticker

myLocator = mticker.MultipleLocator(4)
ax.xaxis.set_major_locator(myLocator)
```

share
improve this answer

answered Jan 21 '13 at 9:55

Andrea Keil
164 3 12

2 Code shown does not "show only every 4th tick". It sets up ticks on integer multiples of 4. – Russ Apr 1 '14 at 22:09

Yes i fell for that, too. It does indeed only set up ticks on integer multiples of 4 and not every 4th tick! – SmCaterpillar Oct 11 '14 at 10:44

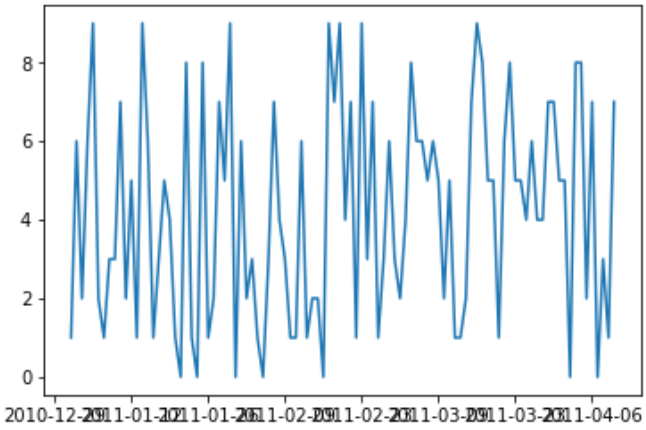
add a comment

1

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
# create a random dataframe with datetimeindex
date_range = pd.date_range('1/1/2011', '4/10/2
df = pd.DataFrame(np.random.randint(0,10,size=
```

Date ticklabels often overlap:

```
plt.plot(df.index,df['value'])
plt.show()
```



So it is useful to rotate them and right align them.

```
fig, ax = plt.subplots()
ax.plot(df.index,df['value'])
ax.xaxis_date()     # interpret the x-axis val
fig.autofmt_xdate() # make space for and rotat
plt.show()
```