Example from

- https://github.com/RodolfoFerro/pandas_twitter/blob/master/01-extracting-data.md (https://github.com/RodolfoFerro/pandas_twitter/blob/master/01-extracting-data.md) **Added some notes and explanation**

Extracting twitter data (tweepy + pandas)

In [6]:
```
# General:
import tweepy           # To consume Twitter's API
import pandas as pd     # To handle data
import numpy as np      # For number computing

# For plotting and visualization:
from IPython.display import display
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [11]:
```
#import libraries that you need
import tweepy
from tweepy import OAuthHandler, Stream

# Go to https://developer.twitter.com/en/apps to create an app and get valu
# for these credentials, which you'll need to provide in place of these
# empty string values that are defined as placeholders.
# See https://developer.twitter.com/en/docs/basics/authentication/overview/
# for more information on Twitter's OAuth implementation.

from credentials import *
#create a file called credentials.py make sure it is in the same folder as
# the credentials file will look like this
#ACCESS_TOKEN = 'xxx'
#ACCESS_SECRET = 'xx'
#CONSUMER_KEY = 'xx'
#CONSUMER_SECRET = 'xx'


auth = tweepy.OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)
auth.set_access_token(ACCESS_TOKEN, ACCESS_SECRET)

twitter_api = tweepy.API(auth,  wait_on_rate_limit=True)

# Nothing to see by displaying twitter_api except that it's now a
# defined variable

print(twitter_api)
```

```
<tweepy.api.API object at 0x7fe478874d90>
```

In [12]:
```
# We import our access keys:


# API's setup:
def twitter_setup():
    """
    Utility function to setup the Twitter's API
    with our access keys provided.
    """
    # Authentication and access using keys:
    auth = tweepy.OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)
    auth.set_access_token(ACCESS_TOKEN, ACCESS_SECRET)

    # Return API with authentication:
    api = tweepy.API(auth)
    return api
```

Tweets Extraction

In [15]:
```
# We create an extractor object:
extractor = twitter_setup()

# We create a tweet list as follows:
tweets = extractor.user_timeline(screen_name="bojangles", count=200)
print("Number of tweets extracted: {}.\n".format(len(tweets)))

# We print the most recent 5 tweets:
print("5 recent tweets:\n")
for tweet in tweets[:5]:
    print(tweet.text)
    print()
```

```
Number of tweets extracted: 198.

5 recent tweets:

When you get the text from your coworker “I’m running late. What you want
from Bojangles?” It’s Sausage &amp;amp; Egg Biscu… https://t.co/BtOtJG41tj (h
ttps://t.co/BtOtJG41tj)

@CharlotteFC WINS! https://t.co/2pqW8Rm0dW (https://t.co/2pqW8Rm0dW)

Tee off in Augusta or tea off from the couch? https://t.co/Fuk2o9YE5x (ht
tps://t.co/Fuk2o9YE5x)

And they’re off! Let’s go @CharlotteFC #forthecrown https://t.co/e2yYKNHr
Cm (https://t.co/e2yYKNHrCm)

RT @CharlotteFC_CFO: Supporters Tailgate is already poppin off. @Bojangle
s handing out breakfast biscuits and prizes too! Supporters march…
```

Creating a (pandas) DataFrame

In [16]:
```
# We create a pandas dataframe as follows:
data = pd.DataFrame(data=[tweet.text for tweet in tweets], columns=['Tweets

# We display the first 10 elements of the dataframe:
display(data.head(10))
```

	Tweets
0	When you get the text from your coworker “I’m ...
1	@CharlotteFC WINS! https://t.co/2pqW8Rm0dW
2	Tee off in Augusta or tea off from the couch? ...
3	And they’re off! Let’s go @CharlotteFC #forthe...
4	RT @CharlotteFC_CFO: Supporters Tailgate is al...
5	pov: you’re a scratch-made Cajun Filet Biscuit...
6	Next #cltfc star? Come take your shot! https:/...
7	https://t.co/s7p1TrlY0k
8	We’re back serving up scratch-made biscuits (w...
9	A bucket of golf balls or Bojangles biscuits? ...

In [17]:
```
# Internal methods of a single tweet object:
print(dir(tweets[0]))
```

```
['__class__', '__delattr__', '__dict__', '__dir__', '__doc__', '__eq__',
'__format__', '__ge__', '__getattribute__', '__getstate__', '__gt__', '__
hash__', '__init__', '__init_subclass__', '__le__', '__lt__', '__module_
_', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__repr__', '__se
tattr__', '__sizeof__', '__slots__', '__str__', '__subclasshook__', '__we
akref__', '_api', '_json', 'author', 'contributors', 'coordinates', 'crea
ted_at', 'destroy', 'entities', 'favorite', 'favorite_count', 'favorite
d', 'geo', 'id', 'id_str', 'in_reply_to_screen_name', 'in_reply_to_status
_id', 'in_reply_to_status_id_str', 'in_reply_to_user_id', 'in_reply_to_us
er_id_str', 'is_quote_status', 'lang', 'parse', 'parse_list', 'place', 'p
ossibly_sensitive', 'retweet', 'retweet_count', 'retweeted', 'retweets',
'source', 'source_url', 'text', 'truncated', 'user']
```

In [18]:
```
# We print info from the first tweet:
print(tweets[0].id)
print(tweets[0].created_at)
print(tweets[0].source)
print(tweets[0].favorite_count)
print(tweets[0].retweet_count)
print(tweets[0].geo)
print(tweets[0].coordinates)
print(tweets[0].entities)
```

```
1514263192918274057
2022-04-13 15:24:13+00:00
Sprinklr Publishing
27
6
None
None
{'hashtags': [], 'symbols': [], 'user_mentions': [], 'urls': [{'url': 'ht
tps://t.co/BtOtJG41tj', 'expanded_url': 'https://twitter.com/i/web/statu
s/1514263192918274057', 'display_url': 'twitter.com/i/web/status/1…', 'in
dices': [121, 144]}]}
```

Adding relevant info to our dataframe

In [19]:
```
# We add relevant data:
data['len']  = np.array([len(tweet.text) for tweet in tweets])
data['ID']   = np.array([tweet.id for tweet in tweets])
data['Date'] = np.array([tweet.created_at for tweet in tweets])
data['Source'] = np.array([tweet.source for tweet in tweets])
data['Likes']  = np.array([tweet.favorite_count for tweet in tweets])
data['RTs']    = np.array([tweet.retweet_count for tweet in tweets])
```

In [20]:
```
# Display of first 10 elements from dataframe:
display(data.head(10))
```

	Tweets	len	ID	Date	Source	Likes	RTs
0	When you get the text from your coworker “I’m ...	144	1514263192918274057	2022-04-13 15:24:13+00:00	Sprinklr Publishing	27	6
1	@CharlotteFC WINS! https://t.co/2pqW8Rm0dW	42	1513244822542426120	2022-04-10 19:57:35+00:00	Twitter for iPhone	11	2
2	Tee off in Augusta or tea off from the couch? ...	69	1513207690838917135	2022-04-10 17:30:02+00:00	Sprinklr Publishing	19	3
3	And they’re off! Let’s go @CharlotteFC #forthe...	75	1513187788878925825	2022-04-10 16:10:57+00:00	Twitter for iPhone	19	2
4	RT @CharlotteFC_CFO: Supporters Tailgate is al...	139	1513178005681721346	2022-04-10 15:32:04+00:00	Twitter for iPhone	0	4
5	pov: you’re a scratch-made Cajun Filet Biscuit...	117	1513175568401018884	2022-04-10 15:22:23+00:00	Twitter for iPhone	96	6
6	Next #cltfc star? Come take your shot! https:/...	62	1513173566417686538	2022-04-10 15:14:26+00:00	Twitter for iPhone	4	0
7	https://t.co/s7p1TrlY0k	23	1513170923519713291	2022-04-10 15:03:56+00:00	Twitter for iPhone	2	0
8	We’re back serving up scratch-made biscuits (w...	139	1513169376685568003	2022-04-10 14:57:47+00:00	Twitter for iPhone	56	7
9	A bucket of golf balls or Bojangles biscuits? ...	69	1512860418410270721	2022-04-09 18:30:05+00:00	Sprinklr Publishing	60	7

Visualization and basic statistics

Averages and popularity

In [21]:
```
# We extract the mean of lenghts:
mean = np.mean(data['len'])

print("The lenght's average in tweets: {}".format(mean))
```

```
The lenght's average in tweets: 82.20202020202021
```

- To extract more data, we will use some pandas' functionalities:

In [22]:
```
# We extract the tweet with more FAVs and more RTs:

fav_max = np.max(data['Likes'])
rt_max  = np.max(data['RTs'])

fav = data[data.Likes == fav_max].index[0]
rt  = data[data.RTs == rt_max].index[0]

# Max FAVs:
print("The tweet with more likes is: \n{}".format(data['Tweets'][fav]))
print("Number of likes: {}".format(fav_max))
print("{} characters.\n".format(data['len'][fav]))

# Max RTs:
print("The tweet with more retweets is: \n{}".format(data['Tweets'][rt]))
print("Number of retweets: {}".format(rt_max))
print("{} characters.\n".format(data['len'][rt]))
```

```
The tweet with more likes is:
Ready for halftime... https://t.co/vNHaMVqky1 (https://t.co/vNHaMVqky1)
Number of likes: 346
45 characters.

The tweet with more retweets is:
Bracket busted? You can still own the watch party game when you enter to
 win a Bo's gift card! We're giving away 2… https://t.co/lFYeR1ZxTN (http
s://t.co/lFYeR1ZxTN)
Number of retweets: 239
139 characters.
```

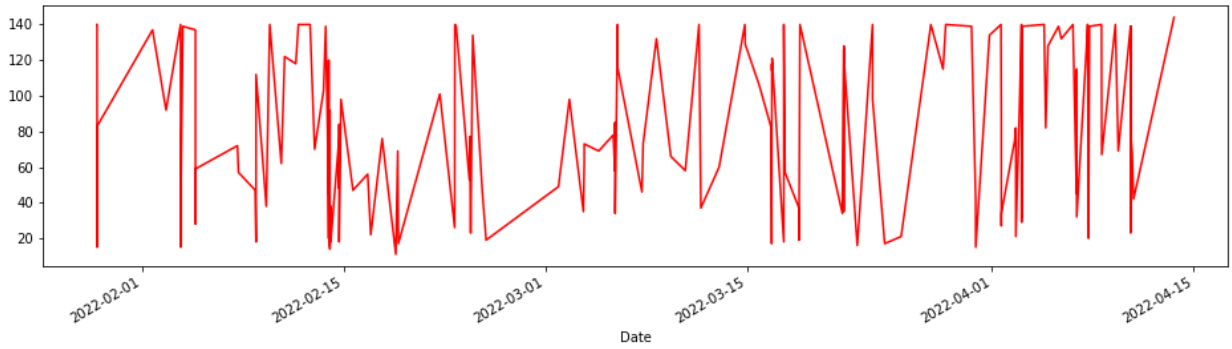
Time series

- Pandas has its own object for time series. Since we have a whole vector with creation dates, we can construct time series respect tweets lengths, likes and retweets.
- The way we do it is:

In [23]:
```
# We create time series for data:
tlen = pd.Series(data=data['len'].values, index=data['Date'])
tfav = pd.Series(data=data['Likes'].values, index=data['Date'])
tret = pd.Series(data=data['RTs'].values, index=data['Date'])
```

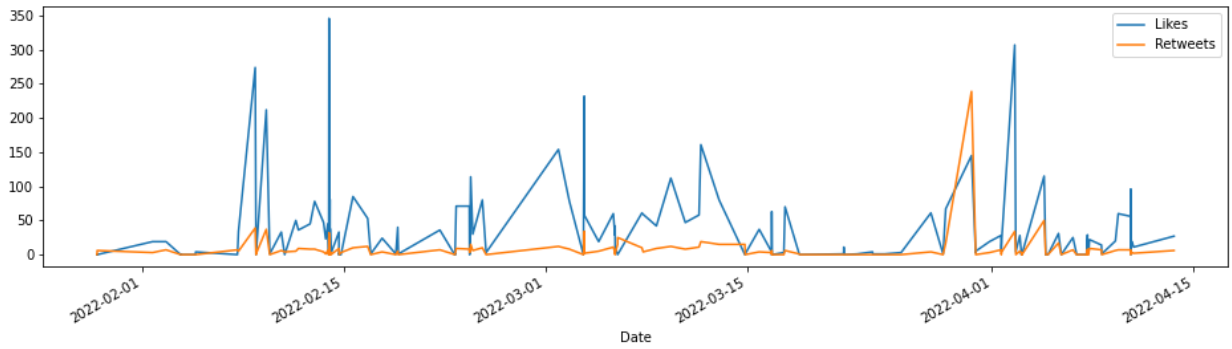
- And if we want to plot the time series, pandas already has its own method in the object. We can plot a time series as follows:

In [24]:
```
# Lenghts along time:
tlen.plot(figsize=(16,4), color='r');
```



- And to plot the likes versus the retweets in the same chart:

In [25]:
```
# Likes vs retweets visualization:
tfav.plot(figsize=(16,4), label="Likes", legend=True)
tret.plot(figsize=(16,4), label="Retweets", legend=True);
```



- Pie charts of sources

In [26]:
```
# We obtain all possible sources:
sources = []
for source in data['Source']:
    if source not in sources:
        sources.append(source)

# We print sources list:
print("Creation of content sources:")
for source in sources:
    print("* {}".format(source))
```

```
Creation of content sources:
* Sprinklr Publishing
* Twitter for iPhone
* Twitter Web App
```

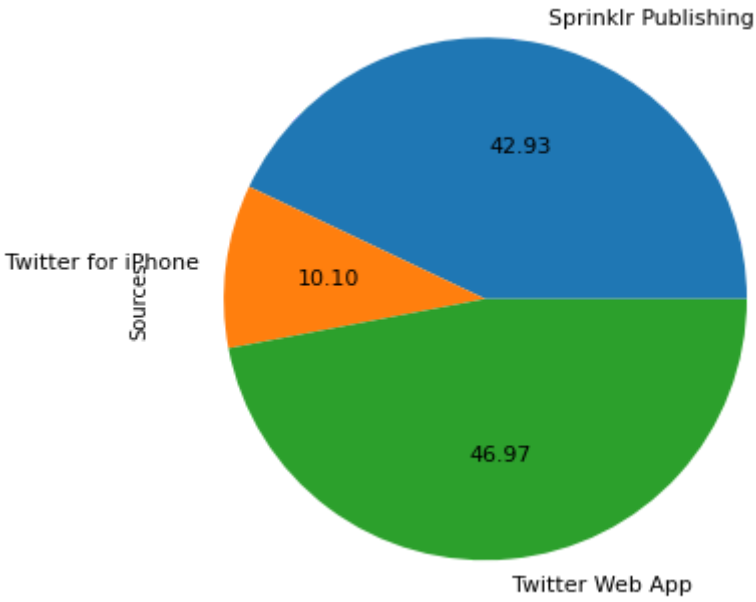
- With the following output, we realize that basically this twitter account has two sources: Creation of content sources: * Twitter for iPhone * Media Studio
- We now count the number of each source and create a pie chart. You'll notice that this code cell is not the most optimized one.

In [27]:
```
# We create a numpy vector mapped to labels:
percent = np.zeros(len(sources))

for source in data['Source']:
    for index in range(len(sources)):
        if source == sources[index]:
            percent[index] += 1
            pass

percent /= 100

# Pie chart:
pie_chart = pd.Series(percent, index=sources, name='Sources')
pie_chart.plot.pie(fontsize=11, autopct='%.2f', figsize=(6, 6));
```



Sentiment analysis

- **Importing textblob**
- textblob will allow us to do sentiment analysis in a very simple way. We will also use the re library from Python, which is used to work with regular expressions.
- For this, I'll provide you two utility functions to: a) clean text (which means that any symbol distinct to an alphanumeric value will be remapped into a new one that satisfies this condition), and b) create a classifier to analyze the polarity of each tweet after cleaning the text in it. I won't explain the specific way in which the function that cleans works, since it would be extended and it might be better understood in the official redocumentation.

In [28]:
```
from textblob import TextBlob
import re

def clean_tweet(tweet):
    '''
    Utility function to clean the text in a tweet by removing
    links and special characters using regex.
    '''
    return ' '.join(re.sub("(@[A-Za-z0-9]+)|([^0-9A-Za-z \t])|(\w+:\/\/\S+)
```

```
def analize_sentiment(tweet):
    '''
    Utility function to classify the polarity of a tweet
    using textblob.
    '''
    analysis = TextBlob(clean_tweet(tweet))
    if analysis.sentiment.polarity > 0:
        return 1
    elif analysis.sentiment.polarity == 0:
        return 0
    else:
        return -1
```

In [29]:
```
# We create a column with the result of the analysis:
data['SA'] = np.array([ analize_sentiment(tweet) for tweet in data['Tweets'
```

```
# We display the updated dataframe with the new column:
display(data.head(10))
```

	Tweets	len	ID	Date	Source	Likes	RTs	SA
0	When you get the text from your coworker “I’m ...	144	1514263192918274057	2022-04-13 15:24:13+00:00	Sprinklr Publishing	27	6	-1
1	@CharlotteFC WINS! https://t.co/2pqW8Rm0dW	42	1513244822542426120	2022-04-10 19:57:35+00:00	Twitter for iPhone	11	2	1
2	Tee off in Augusta or tea off from the couch? ...	69	1513207690838917135	2022-04-10 17:30:02+00:00	Sprinklr Publishing	19	3	0
3	And they’re off! Let’s go @CharlotteFC #forthe...	75	1513187788878925825	2022-04-10 16:10:57+00:00	Twitter for iPhone	19	2	0
4	RT @CharlotteFC_CFO: Supporters Tailgate is al...	139	1513178005681721346	2022-04-10 15:32:04+00:00	Twitter for iPhone	0	4	0
5	pov: you’re a scratch-made Cajun Filet Biscuit...	117	1513175568401018884	2022-04-10 15:22:23+00:00	Twitter for iPhone	96	6	0
6	Next #cltfc star? Come take your shot! https:/...	62	1513173566417686538	2022-04-10 15:14:26+00:00	Twitter for iPhone	4	0	0
7	https://t.co/s7p1TrlY0k	23	1513170923519713291	2022-04-10 15:03:56+00:00	Twitter for iPhone	2	0	0
8	We’re back serving up scratch-made biscuits (w...	139	1513169376685568003	2022-04-10 14:57:47+00:00	Twitter for iPhone	56	7	1
9	A bucket of golf balls or Bojangles biscuits? ...	69	1512860418410270721	2022-04-09 18:30:05+00:00	Sprinklr Publishing	60	7	0

Analyzing the results

- To have a simple way to verify the results, we will count the number of neutral, positive and negative tweets and extract the percentages.

In [30]:
```
# We construct lists with classified tweets:
pos_tweets = [ tweet for index, tweet in enumerate(data['Tweets']) if data[
neu_tweets = [ tweet for index, tweet in enumerate(data['Tweets']) if data[
neg_tweets = [ tweet for index, tweet in enumerate(data['Tweets']) if data[
```

- Now that we have the lists, we just print the percentages:

In [31]:
```
# We print percentages:

print("Percentage of positive tweets: {}%".format(len(pos_tweets)*100/len(d
print("Percentage of neutral tweets: {}%".format(len(neu_tweets)*100/len(da
print("Percentage de negative tweets: {}%".format(len(neg_tweets)*100/len(d
```

```
Percentage of positive tweets: 32.82828282828283%
Percentage of neutral tweets: 47.474747474747474%
Percentage de negative tweets: 19.696969696969695%
```

In [ ]: