In [4]:
```
import pandas as pd
```

In [5]:
```
tweets = pd.read_csv('tcat_TheoAraujo-20170210-20170224---------fullExport--8cef61cd09.csv')
```

Useful Functions for Data Preparation

Here you find a few useful functions for you to use in your data preparation. You can also adapt them for your own needs. The tweets dataframe is being used here as an example.

Pandas built-in functions

Converting a column with dates incorrectly stored as object to datetime

This is useful when the column is stored as object, but actually contains dates. Function: pd.to_datetime

In [6]:
```
tweets['created_at'].head()
```

Out[6]:
```
0    2017-02-20 11:37:08
1    2017-02-20 11:38:15
2    2017-02-20 11:42:33
3    2017-02-20 11:45:30
4    2017-02-20 11:50:11
Name: created_at, dtype: object
```

In [7]:
```
tweets['created_at'] = tweets['created_at'].apply(pd.to_datetime)
```

In [8]:
```
tweets['created_at'].head()
```

Out[8]:
```
0   2017-02-20 11:37:08
1   2017-02-20 11:38:15
2   2017-02-20 11:42:33
3   2017-02-20 11:45:30
4   2017-02-20 11:50:11
Name: created_at, dtype: datetime64[ns]
```

Converting a column with numbers incorrectly stored as object to numeric

Function: pd.to_numeric Note: this may not always work - especially if the column also contains some rows with text.

In [9]:
```
tweets['retweet_count'] = tweets['retweet_count'].apply(pd.to_numeric)
```

In [ ]:
```

```

Some special functions for us

These are some special functions created for our course, of leveraged from elsewhere (credit is mentioned when appropriate). To use them, you need to add them to your code first. Sometimes the usage requires the `.apply()` to be written with additional options. See carefully below.

Checking if a text column contains a certain word

Note:

- This function checks if the word is present or not (returning a 1 or 0)
- This function checks for the full word (i.e., if you search for apple, it won't return apples)
- This function considers mentions or hashtags as normal words (i.e., it ignores @ or # signs).
- This function also is case insensitive.
- The word is included inside the "args" area

In [10]:
```
def word_present(text, query):
    import re
    text = str(text).lower()
    query = str(query).lower()

    tokens = re.findall(r"[\w']+|[.,!?;$@#]", text)
    if query in tokens:
        return 1
    return 0
```

In [11]:
```
tweets['microsoft_present'] = tweets['text'].apply(word_present, args=('microsoft',))
```

In [12]:
```
tweets[['text', 'microsoft_present']].head()
```

Out[12]:

	text	microsoft_present
0	Microsoft flying high in BI & Analytics - Base...	1
1	Sentiment analysis in the age of Digital Trans...	0
2	Eyeview Digital - Manager, Client Analytics - ...	0
3	Leading Digital Marketing Company Eminent Info...	0
4	RT @JimMarous: Top Areas of #Martech Investmen...	0

Checking if an object column contains some text

Credit to Thilo for finding a part of this solution out!

Keep in mind that this function is checking if the query you are searching for appears in any place of the string/text, not necessarily as a separate word. This means that if you search for `apple`, it will return a positive result for a text containing `apple`, but also `apples`, `applebees`, and `pineapple` for example. This can be especially handy for checking if two words are present (in order) in a sentence.

In [13]:
```
def words_present_anywhere(text, query):
    import re
    text = str(text).lower()
    query = str(query).lower()

    if query in text:
        return 1
    return 0
```

In [14]:
```
tweets['flyinghigh_present'] = tweets['text'].apply(words_present_anywhere,
                                                    args=('flying high',))
```

In [15]:
```
tweets[['text', 'flyinghigh_present']].head()
```

Out[15]:

	text	flyinghigh_present
0	Microsoft flying high in BI & Analytics - Base...	1
1	Sentiment analysis in the age of Digital Trans...	0
2	Eyeview Digital - Manager, Client Analytics - ...	0
3	Leading Digital Marketing Company Eminent Info...	0
4	RT @JimMarous: Top Areas of #Martech Investmen...	0

Checking if a set of words appears in the text

This function checks if words (in a list) are also present in the text. This function does not care for word order (i.e., `top areas` and `areas top` are the same for this function), and it does not care if there are other words between the two words being queried.

In [16]:
```
def wordlist_present(text, query):
    import re
    text = str(text).lower()
    newquery = []
    for word in query:
        newquery.append(str(word).lower())
    tokens = re.findall(r"[\w']+|[.,!?;$@#]", text)

    if set(newquery).issubset(tokens):
        return 1
    return 0
```

In [17]:
```
tweets['sentiment_age_present'] = tweets['text'].apply(wordlist_present,
                                                       args=(['sentiment', 'age'],))
```

In [18]:
```
tweets[['text', 'sentiment_age_present']].head()
```

Out[18]:

	text	sentiment_age_present
0	Microsoft flying high in BI & Analytics - Base...	0
1	Sentiment analysis in the age of Digital Trans...	1
2	Eyeview Digital - Manager, Client Analytics - ...	0
3	Leading Digital Marketing Company Eminent Info...	0
4	RT @JimMarous: Top Areas of #Martech Investmen...	0

Checking if any of the words in a list appears in the text

This function checks if any of the words (in a list) are also present in the text. If a word is found (even if the other words in the list are not found), the function returns a 1. If none of the words are found, it returns a 0.

In [19]:
```
def wordlist_any_present(text, query):
    import re
    text = str(text).lower()
    newquery = []
    for word in query:
        newquery.append(str(word).lower())
    tokens = re.findall(r"[\w']+|[.,!?;$@#]", text)

    for word in newquery:
        if word in tokens:
            return 1
    return 0
```

In [20]:
```
tweets['flying_or_top_present'] = tweets['text'].apply(wordlist_any_present,
                                                       args=(['flying', 'top'],))
```

In [21]:
```
tweets[['text', 'flying_or_top_present']].head()
```

Out[21]:

	text	flying_or_top_present
0	Microsoft flying high in BI & Analytics - Base...	1
1	Sentiment analysis in the age of Digital Trans...	0
2	Eyeview Digital - Manager, Client Analytics - ...	0
3	Leading Digital Marketing Company Eminent Info...	0
4	RT @JimMarous: Top Areas of #Martech Investmen...	1

Checking how often a certain word appears

Note: this works with the same conditions as the previous function, except for what it returns.

In [22]:
```
def word_frequency(text, query):
    import re
    text = str(text).lower()
    query = str(query).lower()

    tokens = re.findall(r"[\w']+|[.,!?;$@#]", text)
    counter = 0
    for token in tokens:
        if query == token:
            counter += 1
    return counter
```

In [23]:
```
tweets['client_frequency'] = tweets['text'].apply(word_frequency, args=('client',))
```

In [24]:
```
tweets[['text', 'client_frequency']].head()
```

Out[24]:

	text	client_frequency
0	Microsoft flying high in BI & Analytics - Base...	0
1	Sentiment analysis in the age of Digital Trans...	0
2	Eyeview Digital - Manager, Client Analytics - ...	2
3	Leading Digital Marketing Company Eminent Info...	0
4	RT @JimMarous: Top Areas of #Martech Investmen...	0

Checking if a hashtag is present

Works the same way as the previous functions.

In [25]:
```
def hashtag_present(text, query):
    import re
    text = str(text).lower().replace('#','__')
    query = str(query).lower().replace('#','__')

    tokens = re.findall(r"[\w']+|[.,!?;$@]", text)
    if query in tokens:
        return 1
    return 0
```

In [26]:
```
tweets['ai_hashtag_present'] = tweets['text'].apply(hashtag_present, args=('#AI',))
```

Checking if the tweet is a retweet

You need to always apply this to the text column. This function works only for the DMI-TCAT data.

In [27]:
```
def is_rt(text):
    text = text.split()
    if text[0] == 'RT':
        return 1
    return 0
```

In [28]:
```
tweets['is_retweet'] = tweets['text'].apply(is_rt)
```

In [29]:
```
tweets[['text', 'is_retweet']].head()
```

Out[29]:

	text	is_retweet
0	Microsoft flying high in BI & Analytics - Base...	0
1	Sentiment analysis in the age of Digital Trans...	0
2	Eyeview Digital - Manager, Client Analytics - ...	0
3	Leading Digital Marketing Company Eminent Info...	0
4	RT @JimMarous: Top Areas of #Martech Investmen...	1

Checking if the tweet is a reply

This one also only works for DMI-TCAT data, and specifically with the column in_reply_to_status_id

In [30]:
```
def is_reply(in_reply_to_status_id):
    try:
        int(in_reply_to_status_id)
        return 1
    except:
        return 0
```

In [31]:
```
tweets['is_reply'] = tweets['in_reply_to_status_id'].apply(is_reply)
```

In [32]:
```
tweets[['text', 'is_reply']].head()
```

Out[32]:

	text	is_reply
0	Microsoft flying high in BI & Analytics - Base...	0
1	Sentiment analysis in the age of Digital Trans...	0
2	Eyeview Digital - Manager, Client Analytics - ...	0
3	Leading Digital Marketing Company Eminent Info...	0
4	RT @JimMarous: Top Areas of #Martech Investmen...	0

Checking if tweet has a hashtag

In [33]:
```
def has_hashtag(text):
    if '#' in text:
        return 1
    return 0
```

In [34]:
```
tweets['has_hashtag'] = tweets['text'].apply(has_hashtag)
```

In [35]:
```
def count_hashtag(text):
    text = str(text)
    counter = 0
    for char in text:
        if char == '#':
            counter += 1

    return counter
```

In [36]:
```
tweets['total_hashtags'] = tweets['text'].apply(count_hashtag)
```

In [37]:
```
tweets[['text', 'has_hashtag', 'total_hashtags']].head()
```

Out[37]:

	text	has_hashtag	total_hashtags
0	Microsoft flying high in BI & Analytics - Base...	1	7
1	Sentiment analysis in the age of Digital Trans...	1	4
2	Eyeview Digital - Manager, Client Analytics - ...	1	5
3	Leading Digital Marketing Company Eminent Info...	0	0
4	RT @JimMarous: Top Areas of #Martech Investmen...	1	7

Checking the most frequent words

This function works with any text column, but keep in mind it just returns a report (i.e., it does not create a new column, and you don't **apply** it to a column). The whole code needs to be executed (you can change the column name though).

In [38]:
```
import re
from collections import Counter
```

In [39]:
```
texts = tweets['text'].values.tolist()
```

In [40]:
```
total_words = Counter()
for text in texts:
    # making text lower case
    text = text.lower()
    # removing URLS
    text = text.split(' ')
    newtext = []
    for item in text:
        if 'http' not in item:
            newtext.append(item)

    newtext = ' '.join(newtext)

    # splitting the text in words (tokens)
    tokens = re.findall(r"[\w']+|[.,!?;$@]", newtext)
    for token in tokens:
        total_words[token] += 1
```

In [41]:
```
total_words.most_common(100)
```

Out[41]:
```
[('analytics', 2492),
 ('@', 2217),
 ('digital', 1967),
 ('rt', 1287),
 ('.', 1265),
 (',', 1261),
 ('marketing', 664),
 ('data', 603),
 ('the', 600),
 ('digitaltransformation', 529),
 ('in', 466),
 ('for', 463),
 ('bigdata', 419),
 ('to', 412),
 ('!', 361),
 ('of', 358),
 ('iot', 334),
 ('and', 328),
 ('your', 276),
 ('a', 243),
 ('how', 227),
 ('seo', 221),
 ('via', 204),
 ('?', 203),
 ('is', 194),
 ('business', 169),
 ('transformation', 166),
 ('cmo', 160),
 ('with', 159),
 ('cx', 155),
 ('can', 154),
 ('microsoft', 154),
 ('launches', 148),
 ('focused', 145),
 ('windows', 145),
 ('ex', 145),
 ('an', 140),
 ('by', 134),
 ('manager', 133),
```

```
('cloud', 133),
('firm', 133),
('digitalmarketing', 125),
('infórmate', 122),
('app', 122),
('google', 121),
('media', 114),
('strategy', 114),
('fjuri', 111),
('s', 109),
('adwords', 108),
('mikeflache', 106),
('ai', 105),
('datascience', 105),
('using', 104),
('mobile', 103),
('on', 100),
('helps', 99),
('cursos', 95),
('experience', 91),
('into', 89),
('are', 88),
('tech', 88),
('new', 87),
('2017', 86),
('store', 85),
('retailers', 84),
('socialmedia', 84),
('virtual', 83),
('tools', 83),
('social', 81),
('top', 80),
('wordpress', 79),
('connecting', 79),
('fabric', 79),
('at', 78),
('blog', 78),
('5', 78),
('you', 78),
('it', 77),
('community', 77),
('6', 77),
```

```
('learn', 75),
('cisco', 75),
('why', 75),
('integration', 75),
('ease', 75),
('bridgei2i', 75),
('key', 74),
('machinelearning', 73),
('3', 73),
('out', 69),
('customers', 68),
('technology', 68),
('improve', 67),
('supplychain', 67),
('cainc', 66),
('our', 66),
('derekinthecloud', 65),
('2', 64),
('4', 63)]
```

Creating a column based on multiple conditions

This function needs to be applied to the whole dataframe (see example below), and can be configured to create new columns based on multiple conditions. Keep in mind that you are working with each row separately.

In [42]:
```
def categorise_df(row):
    # Here I am creating a new column (called user_type) and giving it a default value (regular user)
    row['user_type'] = 'regular user'
    # I can put the conditions I want here
    if (row['from_user_followercount'] > 2000) and (row['from_user_friendcount'] < 2000):
        row['user_type'] = 'celebrity'

    if (row['from_user_followercount'] <= 2000) and (row['from_user_friendcount'] >= 2000):
        row['user_type'] = 'bot'

    # You can modify this function as much as you want (above), but it must always return the row
    return row
```

In [43]:
```
tweets2 = tweets.apply(categorise_df, axis=1)
```

In [44]:
```
tweets2[['from_user_followercount', 'from_user_friendcount', 'user_type']].head()
```

Out[44]:

	from_user_followercount	from_user_friendcount	user_type
0	6	0	regular user
1	594	745	regular user
2	13	0	regular user
3	81	34	regular user
4	2327	3	celebrity

In [45]:
```
tweets.to_pickle('tweets_Theo.pkl')
```

In [ ]:

In [ ]: