Regex+with+Pandas+and+Named+Groups

July 1, 2020

*You are currently looking at **version 1.0** of this notebook. To download notebooks and datafiles, as well as get help on Jupyter notebooks in the Coursera platform, visit the Jupyter Notebook FAQ course resource.*

1 Working with Text Data in pandas

[1]:
```
import pandas as pd

time_sentences = ["Monday: The doctor's appointment is at 2:45pm.",
                  "Tuesday: The dentist's appointment is at 11:30 am.",
                  "Wednesday: At 7:00pm, there is a basketball game!",
                  "Thursday: Be back home by 11:15 pm at the latest.",
                  "Friday: Take the train at 08:10 am, arrive at 09:00am."]

df = pd.DataFrame(time_sentences, columns=['text'])
df
```

[1]:
```
                                                text
0     Monday: The doctor's appointment is at 2:45pm.
1  Tuesday: The dentist's appointment is at 11:30...
2  Wednesday: At 7:00pm, there is a basketball game!
3  Thursday: Be back home by 11:15 pm at the latest.
4  Friday: Take the train at 08:10 am, arrive at ...
```

[2]:
```
# find the number of characters for each string in df['text']
df['text'].str.len()
```

[2]:
```
0    46
1    50
2    49
3    49
4    54
Name: text, dtype: int64
```

[3]:
```
# find the number of tokens for each string in df['text']
df['text'].str.split().str.len()
```

[3]:
```
0     7
1     8
2     8
3    10
4    10
Name: text, dtype: int64
```

[4]:
```
# find which entries contain the word 'appointment'
df['text'].str.contains('appointment')
```

[4]:
```
0     True
1     True
2    False
3    False
4    False
Name: text, dtype: bool
```

[6]:
```
# find how many times a digit occurs in each string
df['text'].str.count(r'\d')
```

[6]:
```
0    3
1    4
2    3
3    4
4    8
Name: text, dtype: int64
```

[7]:
```
# find all occurances of the digits
df['text'].str.findall(r'\d')
```

[7]:
```
0                   [2, 4, 5]
1                [1, 1, 3, 0]
2                   [7, 0, 0]
3                [1, 1, 1, 5]
4    [0, 8, 1, 0, 0, 9, 0, 0]
Name: text, dtype: object
```

[10]:
```
# group and find the hours and minutes
df['text'].str.findall(r'(\d?\d):(\d\d)')
```

[10]:
```
0               [(2, 45)]
1              [(11, 30)]
2               [(7, 00)]
3              [(11, 15)]
4    [(08, 10), (09, 00)]
Name: text, dtype: object
```

[11]:
```
# replace weekdays with '???'
df['text'].str.replace(r'\w+day\b', '???')
```

[11]:
```
0          ???: The doctor's appointment is at 2:45pm.
1       ???: The dentist's appointment is at 11:30 am.
```

```
2          ???: At 7:00pm, there is a basketball game!
3         ???: Be back home by 11:15 pm at the latest.
4    ???: Take the train at 08:10 am, arrive at 09:...
Name: text, dtype: object
```

[12]:
```
# replace weekdays with 3 letter abbrevations
df['text'].str.replace(r'(\w+day\b)', lambda x: x.groups()[0][:3])
```

[12]:
```
0          Mon: The doctor's appointment is at 2:45pm.
1       Tue: The dentist's appointment is at 11:30 am.
2          Wed: At 7:00pm, there is a basketball game!
3         Thu: Be back home by 11:15 pm at the latest.
4    Fri: Take the train at 08:10 am, arrive at 09:...
Name: text, dtype: object
```

[13]:
```
# create new columns from first match of extracted groups
df['text'].str.extract(r'(\d?\d):(\d\d)')
```

```
/opt/conda/lib/python3.6/site-packages/ipykernel_launcher.py:2: FutureWarning:
currently extract(expand=None) means expand=False (return
Index/Series/DataFrame) but in a future version of pandas this will be changed
to expand=True (return DataFrame)
```

[13]:
```
    0   1
0   2  45
1  11  30
2   7  00
3  11  15
4  08  10
```

[14]:
```
# extract the entire time, the hours, the minutes, and the period
df['text'].str.extractall(r'((\d?\d):(\d\d) ?([ap]m))')
```

[14]:
```
                0   1   2   3
  match
0 0        2:45pm   2  45  pm
1 0      11:30 am  11  30  am
2 0        7:00pm   7  00  pm
3 0      11:15 pm  11  15  pm
4 0      08:10 am  08  10  am
  1       09:00am  09  00  am
```

[17]:
```
# extract the entire time, the hours, the minutes, and the period with group␣
↪names
df['text'].str.extractall(r'(?P<time>(?P<hour>\d?\d):(?P<minute>\d\d) ?(?
↪P<period>[ap]m))')
```

[17]:
```
             time hour minute period
  match
0 0        2:45pm    2     45     pm
```

```
1 0       11:30 am   11      30      am
2 0         7:00pm    7      00      pm
3 0       11:15 pm   11      15      pm
4 0       08:10 am   08      10      am
  1        09:00am   09      00      am
```