PANDAS PART 7

Regular expressions in pandas

filter(), contains(), findall(), replace(), extract(), extractall(), split()

[53]:
```
# pandas regular expressions and how to use them on data frames
import pandas as pd
```

[54]:
```
df = pd.read_csv('D:/aatish/datasets/carprices.csv')
df
```

[54]:

	car model	mileage	sale price	age	model_date
0	BMW X5	69000	18000	6	12-12-2000
1	BMW X5	35000	34000	3	01-10-2000
2	BMW X5	57000	26100	5	12-05-1995
3	BMW X5	22500	40000	2	20-05-1994
4	BMW X5	46000	31500	4	05-05-1992
5	Audi A5	59000	29400	5	10-10-2002

[77]:
```
# filter the columns whose names start with 'm'
df.filter(regex='^m', axis=1)
```

[77]:

	mileage	model_date
0	69000	12-12-2000
1	35000	01-10-2000
2	57000	12-05-1995
3	22500	20-05-1994
4	46000	05-05-1992
5	59000	10-10-2002
6	52000	10-10-2002

[65]:
```
# filter the columns whose names end with 'age'
df.filter(regex='age$', axis=1)
```

[65]:

	mileage	age
0	69000	6
1	35000	3
2	57000	5
3	22500	2
4	46000	4
5	59000	5
6	52000	5

---

[56]:
```
# take only car model column
df_car = df['car model']
df_car
```

[56]:
```
0                    BMW X5
1                    BMW X5
2                    BMW X5
3                    BMW X5
4                    BMW X5
5                   Audi A5
6                   Audi A5
7                   Audi A5
8                   Audi A5
9     Mercedez Benz C class
10    Mercedez Benz C class
11    Mercedez Benz C class
12    Mercedez Benz C class
Name: car model, dtype: object
```

---

[57]:
```
# find no. of chars in the car names
df_car.str.len()
```

[57]:
```
0      6
1      6
2      6
3      6
4      6
5      7
6      7
7      7
8      7
9     21
10    21
11    21
12    21
Name: car model, dtype: int64
```

---

[82]:
```
# know if Audi is there in the car names - gives boolean values
df_car.str.contains(r'Audi \w*')
```

[82]:
```
0     False
1     False
2     False
3     False
4     False
5      True
6      True
7      True
8      True
9     False
10    False
11    False
12    False
Name: car model, dtype: bool
```

---

[83]:
```
# to display only those rows with Audi car names in data frame
df[df_car.str.contains(r'Audi \w*')]
```

[83]:

	car model	mileage	sale price	age	model_date
5	Audi A5	59000	29400	5	10-10-2002
6	Audi A5	52000	32000	5	10-10-2002
7	Audi A5	72000	19300	6	25-05-2001
8	Audi A5	91000	12000	8	15-09-1997

---

[59]:
```
# display the rows having Audi - returns Series object
obj = df_car.str.findall(r'Audi')
print(obj)
```

```
0         []
1         []
2         []
3         []
4         []
5     [Audi]
6     [Audi]
7     [Audi]
8     [Audi]
9         []
10        []
11        []
12        []
Name: car model, dtype: object
```

---

[60]:
```
# Replace Audi by Bodi
df_car.str.replace(r'Audi', 'Bodi')
```

[60]:
```
0                    BMW X5
1                    BMW X5
2                    BMW X5
3                    BMW X5
4                    BMW X5
5                   Bodi A5
6                   Bodi A5
7                   Bodi A5
8                   Bodi A5
9     Mercedez Benz C class
10    Mercedez Benz C class
11    Mercedez Benz C class
12    Mercedez Benz C class
Name: car model, dtype: object
```

[61]:
```
# extract all rows with BMW X5 - gives data frame object
df1 = df_car.str.extract(r'(BMW \w\w)')
df1

# NOTE: Use extractall() above and see the difference
```

[61]:

	0
0	BMW X5
1	BMW X5
2	BMW X5
3	BMW X5
4	BMW X5
5	NaN
6	NaN
7	NaN
8	NaN
9	NaN
10	NaN
11	NaN
12	NaN

[62]:
```
# split into pieces where X5 or A5 is found
lst = df_car.str.split(r'[XA]\d')
for x in lst: print(x)
```

```
['BMW ', '']
['BMW ', '']
['BMW ', '']
['BMW ', '']
['BMW ', '']
['Audi ', '']
['Audi ', '']
['Audi ', '']
['Audi ', '']
['Mercedez Benz C class']
['Mercedez Benz C class']
['Mercedez Benz C class']
['Mercedez Benz C class']
```

[71]:
```
# grab only years from model_date
df['model_date'].str.findall(r'\d\d-\d\d-(\d\d\d\d)')
```

:[71]:
```
0     [2000]
1     [2000]
2     [1995]
3     [1994]
4     [1992]
5     [2002]
6     [2002]
7     [2001]
8     [1997]
9     [1998]
10    [1996]
11    [1999]
12    [2004]
Name: model_date, dtype: object
```

[76]:
```
# to retrieve month and years from model_date - gives series object
obj = df['model_date'].str.findall(r'\d\d-(\d\d)-(\d\d\d\d)')
obj

# use extractall() to get data frame object
```

[76]:
```
0     [(12, 2000)]
1     [(10, 2000)]
2     [(05, 1995)]
3     [(05, 1994)]
4     [(05, 1992)]
5     [(10, 2002)]
6     [(10, 2002)]
7     [(05, 2001)]
8     [(09, 1997)]
9     [(09, 1998)]
10    [(09, 1996)]
11    [(09, 1999)]
12    [(09, 2004)]
Name: model_date, dtype: object
```