Week 5. 데이터 분석 기초

데이터 분석의 순서

1. 데이터 읽기
2. 데이터 전처리
3. 데이터 특성 살피기
4. 적절한 분석 모델 또는 학습 알고리즘 선택
    - 단순한 모델 부터
    - 좀 더 복잡한 모델
    - 훈련과 테스트
5. 모델 간 비교(성능 테스트)

데이터 읽기

파이썬에서 파일을 열기 위해서는 `open` 이라는 내장 함수를 사용

`open(filename, [mode])` 은 "파일 이름"과 "읽기 방법"을 입력받아 파일 객체를 리턴하는 함수이다. 읽기 방법(mode)이 생략되면 기본값인 읽기 전용 모드(r)로 파일 객체를 만들어 리턴한다.

- 첫 번째 인자로는 파일의 경로를 입력
    - 파일의 경로를 입력할 때 주의할 점은 디렉터리의 구분자로 '\' 하나를 사용하는 것이 아니라 `'\\'` 또는 `'/'`를 사용함
- 두 번째 인자로는 파일을 열기위한 모드를 입력
    - `r` : 읽기 모드
    - `w` : 쓰기 모드
    - `a` : 추가 모드
    - `b` : 바이너리 모드
    - `t` : 텍스트 모드
- 옵션으로 `encoding=` 을 줄수 있음.

`readlines()` : 파일을 읽어서 각 라인을 리스트에 넣은 후 리스트를 리턴

`read()` : 파일의 내용 전체를 문자열로 리턴

txt 파일

In [25]:
```
f = open("data/create.txt", 'w', encoding='utf-8')

for i in range(10):
    data = "%d번째 줄입니다.\n" %(i)
    f.write(data)

f.close()
```

In [26]:
```
f = open("data/create.txt", 'r', encoding='utf-8')
print(f)

while True:
    line = f.readline()
    print(line)
    if not line:
        break

f.close()
```

```
<_io.TextIOWrapper name='data/create.txt' mode='r' encoding='utf-8'>
0번째 줄입니다.

1번째 줄입니다.

2번째 줄입니다.

3번째 줄입니다.

4번째 줄입니다.

5번째 줄입니다.

6번째 줄입니다.

7번째 줄입니다.

8번째 줄입니다.

9번째 줄입니다.
```

In [27]:
```
f = open("data/create.txt", 'r', encoding='utf-8')

lines = f.readlines()
print(lines)

# for line in lines:
#     print(line)

f.close()
```

```
['0번째 줄입니다.\n', '1번째 줄입니다.\n', '2번째 줄입니다.\n', '3번째 줄입니다.\n', '4
번째 줄입니다.\n', '5번째 줄입니다.\n', '6번째 줄입니다.\n', '7번째 줄입니다.\n', '8번째
 줄입니다.\n', '9번째 줄입니다.\n']
```

> 파일을 읽기 모드로 연 후 readline()을 이용해서 파일의 첫 번째 줄을 읽어 출력하는 경우이다.

In [28]:
```
f = open("data/create.txt", 'r', encoding='utf-8')
data = f.read()
print(data)
f.close()

new_data = data.splitlines()
print(new_data)
```

```
0번째 줄입니다.
1번째 줄입니다.
2번째 줄입니다.
3번째 줄입니다.
4번째 줄입니다.
5번째 줄입니다.
6번째 줄입니다.
7번째 줄입니다.
8번째 줄입니다.
9번째 줄입니다.

['0번째 줄입니다.', '1번째 줄입니다.', '2번째 줄입니다.', '3번째 줄입니다.', '4번째 줄입니다.', '5번째 줄입니다.', '6번째 줄입니다.', '7번째 줄입니다.', '8번째 줄입니다.', '9번째 줄입니다.']
```

파일에 새로운 내용 추가하기

쓰기 모드(`w`)로 파일을 열 때 이미 존재하는 파일을 열 경우 그 파일의 내용이 모두 사라지게 됨

원래 있던 값을 유지하면서 단지 새로운 값만 추가해야 할 경우도 있다. 이런 경우에는 파일을 추가 모드(`a`)로 염

In [ ]:
```
f = open("data/create.txt",'a')

for i in range(11, 20):
    data = "%d번째 줄입니다.\n" %(i)
    f.write(data)

f.close()
```

close()를 명시적으로.

In [29]:
```
try:
    f = open("data/create.txt", 'r', encoding='utf-8')
    for line in f:
        print(line)

finally:
    f.close()
```

```
0번째 줄입니다.

1번째 줄입니다.

2번째 줄입니다.

3번째 줄입니다.

4번째 줄입니다.

5번째 줄입니다.

6번째 줄입니다.

7번째 줄입니다.

8번째 줄입니다.

9번째 줄입니다.
```

In [30]:
```
with open("data/create.txt", "r", encoding='utf-8') as f:
    data = f.read()

print(data)
```

```
0번째 줄입니다.
1번째 줄입니다.
2번째 줄입니다.
3번째 줄입니다.
4번째 줄입니다.
5번째 줄입니다.
6번째 줄입니다.
7번째 줄입니다.
8번째 줄입니다.
9번째 줄입니다.
```

> 'with open' ==> open 과 close 사이에 try~except~finally 가 속해있는 형태

CSV 파일

In [34]:
```
import csv

data = []

f = open('data/005930.ks.csv', 'r', encoding='utf-8')
rdr = csv.reader(f)

for row in rdr:
    data.append(row)

f.close()

print(data)
```

```
[['Date', 'Open', 'High', 'Low', 'Close', 'Adj Close', 'Volume'], ['2017-05-02', '22
75000.000000', '2275000.000000', '2238000.000000', '2245000.000000', '2245000.00000
0', '281366'], ['2017-05-04', '2245000.000000', '2285000.000000', '2243000.000000',
 '2276000.000000', '2276000.000000', '273802'], ['2017-05-08', '2276000.000000', '23
51000.000000', '2267000.000000', '2351000.000000', '2351000.000000', '391651'], ['20
17-05-10', '2308000.000000', '2361000.000000', '2280000.000000', '2280000.000000',
 '2280000.000000', '468219'], ['2017-05-11', '2271000.000000', '2309000.000000', '22
61000.000000', '2275000.000000', '2275000.000000', '425557'], ['2017-05-12', '228800
0.000000', '2308000.000000', '2283000.000000', '2291000.000000', '2291000.000000',
 '188458'], ['2017-05-15', '2281000.000000', '2314000.000000', '2281000.000000', '23
05000.000000', '2305000.000000', '160028'], ['2017-05-16', '2333000.000000', '234000
0.000000', '2305000.000000', '2319000.000000', '2319000.000000', '176075'], ['2017-0
5-17', '2306000.000000', '2332000.000000', '2305000.000000', '2317000.000000', '2317
000.000000', '148489'], ['2017-05-18', '2287000.000000', '2300000.000000', '2277000.
000000', '2297000.000000', '2297000.000000', '223207'], ['2017-05-19', '2282000.0000
00', '2289000.000000', '2236000.000000', '2236000.000000', '2236000.000000', '31524
7'], ['2017-05-22', '2252000.000000', '2269000.000000', '2238000.000000', '2255000.0
00000', '2255000.000000', '352871'], ['2017-05-23', '2270000.000000', '2279000.00000
0', '2245000.000000', '2246000.000000', '2246000.000000', '252141'], ['2017-05-24',
 '2243000.000000', '2265000.000000', '2240000.000000', '2244000.000000', '2244000.00
0000', '173508'], ['2017-05-25', '2258000.000000', '2284000.000000', '2240000.00000
0', '2284000.000000', '2284000.000000', '260896'], ['2017-05-26', '2280000.000000',
 '2323000.000000', '2277000.000000', '2304000.000000', '2304000.000000', '272273'],
 ['2017-05-29', '2311000.000000', '2320000.000000', '2269000.000000', '2281000.00000
0', '2281000.000000', '174791']]
```

In [117]:
```
with open('data/005930.ks.csv', 'r', encoding='utf-8') as f:
    data = f.readlines()

print(data)
```

```
['Date,Open,High,Low,Close,Adj Close,Volume\n', '2017-05-02,2275000.000000,2275000.0
00000,2238000.000000,2245000.000000,2245000.000000,281366\n', '2017-05-04,2245000.00
0000,2285000.000000,2243000.000000,2276000.000000,2276000.000000,273802\n', '2017-05
-08,2276000.000000,2351000.000000,2267000.000000,2351000.000000,2351000.000000,39165
1\n', '2017-05-10,2308000.000000,2361000.000000,2280000.000000,2280000.000000,228000
0.000000,468219\n', '2017-05-11,2271000.000000,2309000.000000,2261000.000000,227500
0.000000,2275000.000000,425557\n', '2017-05-12,2288000.000000,2308000.000000,228300
0.000000,2291000.000000,2291000.000000,188458\n', '2017-05-15,2281000.000000,231400
0.000000,2281000.000000,2305000.000000,2305000.000000,160028\n', '2017-05-16,233300
0.000000,2340000.000000,2305000.000000,2319000.000000,2319000.000000,176075\n', '201
7-05-17,2306000.000000,2332000.000000,2305000.000000,2317000.000000,2317000.000000,1
48489\n', '2017-05-18,2287000.000000,2300000.000000,2277000.000000,2297000.000000,22
97000.000000,223207\n', '2017-05-19,2282000.000000,2289000.000000,2236000.000000,223
6000.000000,2236000.000000,315247\n', '2017-05-22,2252000.000000,2269000.000000,2238
000.000000,2255000.000000,2255000.000000,352871\n', '2017-05-23,2270000.000000,22790
00.000000,2245000.000000,2246000.000000,2246000.000000,252141\n', '2017-05-24,224300
0.000000,2265000.000000,2240000.000000,2244000.000000,2244000.000000,173508\n', '201
7-05-25,2258000.000000,2284000.000000,2240000.000000,2284000.000000,2284000.000000,2
60896\n', '2017-05-26,2280000.000000,2323000.000000,2277000.000000,2304000.000000,23
04000.000000,272273\n', '2017-05-29,2311000.000000,2320000.000000,2269000.000000,228
1000.000000,2281000.000000,174791\n']
```

In [2]:
```
import csv

data = []
f = open('data/005930.ks.csv', 'r', encoding='utf-8')
rdr = csv.reader(f)

headers = next(rdr)  # skip the headers
print(headers)

for row in rdr:
    data.append(row)
f.close()

# print(data)

# print(*lista)
zipped = list(zip(*data))
print(zipped)
```

```
['Date', 'Open', 'High', 'Low', 'Close', 'Adj Close', 'Volume']
[('2017-05-02', '2017-05-04', '2017-05-08', '2017-05-10', '2017-05-11', '2017-05-1
2', '2017-05-15', '2017-05-16', '2017-05-17', '2017-05-18', '2017-05-19', '2017-05-2
2', '2017-05-23', '2017-05-24', '2017-05-25', '2017-05-26', '2017-05-29'), ('227500
0.000000', '2245000.000000', '2276000.000000', '2308000.000000', '2271000.000000',
 '2288000.000000', '2281000.000000', '2333000.000000', '2306000.000000', '2287000.00
0000', '2282000.000000', '2252000.000000', '2270000.000000', '2243000.000000', '2258
000.000000', '2280000.000000', '2311000.000000'), ('2275000.000000', '2285000.00000
0', '2351000.000000', '2361000.000000', '2309000.000000', '2308000.000000', '231400
0.000000', '2340000.000000', '2332000.000000', '2300000.000000', '2289000.000000',
 '2269000.000000', '2279000.000000', '2265000.000000', '2284000.000000', '2323000.00
0000', '2320000.000000'), ('2238000.000000', '2243000.000000', '2267000.000000', '22
80000.000000', '2261000.000000', '2283000.000000', '2281000.000000', '2305000.00000
0', '2305000.000000', '2277000.000000', '2236000.000000', '2238000.000000', '224500
0.000000', '2240000.000000', '2240000.000000', '2277000.000000', '2269000.000000'),
 ('2245000.000000', '2276000.000000', '2351000.000000', '2280000.000000', '2275000.0
00000', '2291000.000000', '2305000.000000', '2319000.000000', '2317000.000000', '229
7000.000000', '2236000.000000', '2255000.000000', '2246000.000000', '2244000.00000
0', '2284000.000000', '2304000.000000', '2281000.000000'), ('2245000.000000', '22760
00.000000', '2351000.000000', '2280000.000000', '2275000.000000', '2291000.000000',
 '2305000.000000', '2319000.000000', '2317000.000000', '2297000.000000', '2236000.00
0000', '2255000.000000', '2246000.000000', '2244000.000000', '2284000.000000', '2304
000.000000', '2281000.000000'), ('281366', '273802', '391651', '468219', '425557',
 '188458', '160028', '176075', '148489', '223207', '315247', '352871', '252141', '17
3508', '260896', '272273', '174791')]
```

In [6]:
```
aa= [[1,2],
    [3,4],
    [5,6]]

print(list(zip(*aa)))

a = [1,2,3,4]

b = [i*2 for i in a]

b=[]
for i in a:
    b.append(i*2)


b = {i:i*2 for i in a}

print(b)
```

```
[(1, 3, 5), (2, 4, 6)]
{1: 2, 2: 4, 3: 6, 4: 8}
```

In [7]:
```
with open('data/005930.ks.csv', 'r', encoding='utf-8') as f:
    rdr = csv.reader(f)
    headers = next(rdr)  # skip the headers
    print(headers)
    # Dictionary comprehension
    data = {h:[] for h in headers}
    print(data)

    for row in rdr:
        for h, v in zip(headers, row):
            data[h].append(v)

print(data)
```

```
['Date', 'Open', 'High', 'Low', 'Close', 'Adj Close', 'Volume']
{'Close': [], 'Adj Close': [], 'Date': [], 'Volume': [], 'Open': [], 'High': [], 'Lo
w': []}
{'Close': ['2245000.000000', '2276000.000000', '2351000.000000', '2280000.000000',
 '2275000.000000', '2291000.000000', '2305000.000000', '2319000.000000', '2317000.00
0000', '2297000.000000', '2236000.000000', '2255000.000000', '2246000.000000', '2244
000.000000', '2284000.000000', '2304000.000000', '2281000.000000'], 'Adj Close': ['2
245000.000000', '2276000.000000', '2351000.000000', '2280000.000000', '2275000.00000
0', '2291000.000000', '2305000.000000', '2319000.000000', '2317000.000000', '229700
0.000000', '2236000.000000', '2255000.000000', '2246000.000000', '2244000.000000',
 '2284000.000000', '2304000.000000', '2281000.000000'], 'Date': ['2017-05-02', '2017
-05-04', '2017-05-08', '2017-05-10', '2017-05-11', '2017-05-12', '2017-05-15', '2017
-05-16', '2017-05-17', '2017-05-18', '2017-05-19', '2017-05-22', '2017-05-23', '2017
-05-24', '2017-05-25', '2017-05-26', '2017-05-29'], 'Volume': ['281366', '273802',
 '391651', '468219', '425557', '188458', '160028', '176075', '148489', '223207', '31
5247', '352871', '252141', '173508', '260896', '272273', '174791'], 'Open': ['227500
0.000000', '2245000.000000', '2276000.000000', '2308000.000000', '2271000.000000',
 '2288000.000000', '2281000.000000', '2333000.000000', '2306000.000000', '2287000.00
0000', '2282000.000000', '2252000.000000', '2270000.000000', '2243000.000000', '2258
000.000000', '2280000.000000', '2311000.000000'], 'High': ['2275000.000000', '228500
0.000000', '2351000.000000', '2361000.000000', '2309000.000000', '2308000.000000',
 '2314000.000000', '2340000.000000', '2332000.000000', '2300000.000000', '2289000.00
0000', '2269000.000000', '2279000.000000', '2265000.000000', '2284000.000000', '2323
000.000000', '2320000.000000'], 'Low': ['2238000.000000', '2243000.000000', '226700
0.000000', '2280000.000000', '2261000.000000', '2283000.000000', '2281000.000000',
 '2305000.000000', '2305000.000000', '2277000.000000', '2236000.000000', '2238000.00
0000', '2245000.000000', '2240000.000000', '2240000.000000', '2277000.000000', '2269
000.000000']}
```

In [ ]:
```
```