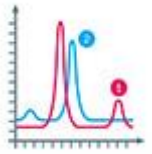
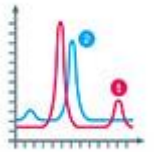


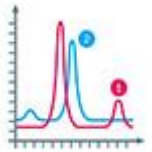
NHẬP MÔN LẬP TRÌNH KHOA HỌC DỮ LIỆU

Bài 10: Thư viện Pandas (2)

Nội dung



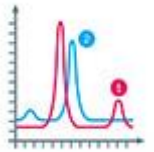
1. Chữa bài tập buổi trước
2. Làm việc với panel
3. Chọn và nhóm phần tử
4. Sử dụng pandas trong bài toán thực tế
5. Bài tập



Phần 1

Chữa bài tập buổi trước

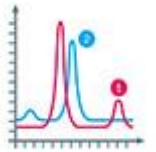
Bài tập



Nhập dữ liệu từ file **k59.csv** (file kèm với bài giảng)

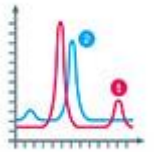
1. In dữ liệu ra màn hình
2. In 5 dòng đầu tiên và 5 dòng cuối cùng của dữ liệu ra màn hình
3. Thống kê xem lớp có bao nhiêu bạn điểm loại giỏi (điểm từ 8 trở lên)
4. Thông kê xem lớp có bao nhiêu bạn trượt môn (điểm dưới 4 hoặc không có điểm)
5. Vẽ đồ thị histogram minh họa phân bổ điểm số của lớp (trục giá trị từ 0 đến 10, không có điểm tính là 0)

Bài chữa



```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

d = pd.read_csv("k59.csv", index_col = 0)

print(d)                          # câu 1: in dữ liệu ra màn hình

print(d.head(5))                  # câu 2: in 5 dòng đầu tiên
print(d.tail(5))                  # câu 2: in 5 dòng cuối cùng

print(len(d[d.Diem >= 8]))        # câu 3: thống kê loại giỏi
```

Bài chữa

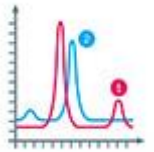


```
# câu 4: thống kê trượt môn
print(len(d[(d.Diem < 4) | (d.Diem.isnull())]))

# câu 5: vẽ đồ thị histogram phân bổ điểm
d.Diem.plot(kind='hist', bins=10)
plt.show()

# gán cho những dòng thiếu điểm thành điểm 0
d.Diem.fillna(0, inplace=True)
# thống kê theo loại điểm (so sánh xem khác histogram ở điểm nào?)
# cách khác: d.groupby('Diem').count()['MaSV'].plot(kind='bar')
d.Diem.value_counts().sort_index().plot('bar')
plt.show()
```

Phần 2

Làm việc với panel

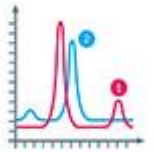
Cấu trúc panel



- Panel được sử dụng nhiều trong kinh tế lượng
- Dữ liệu có 3 trục:
  - Items (trục 0): mỗi item là một dataframe bên trong
  - Major axis (trục 1 – trục chính): các dòng
  - Minor axis (trục 2 – trục phụ): các cột
- Không được phát triển tiếp (thay bởi MultiIndex)

		Open	Close
Major	Minor		
3/31/2015	IBM	23.602	132.903
	APPL	421.412	212.665
	CVX	568.055	409.201
	BHP	487.414	515.413
4/30/2015	IBM	150.868	457.895
	APPL	204.729	957.179
	CVX	90.679	888.687
	BHP	831.527	714.202
5/31/2015	IBM	788.582	922.422
	APPL	329.716	304.964
	CVX	36.578	981.508
	BHP	313.848	882.293

Tạo panel

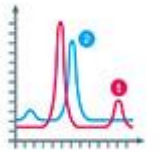


- Cú pháp:

  pandas.Panel(data, items, major_axis, minor_axis, dtype, copy)

- Trong đó:
  - ‘data’ có thể nhận các kiểu dữ liệu sau: ndarray, series, map, lists, dict, hằng số và cả dataframe khác
  - ‘items’ là axis = 0
  - ‘major_axis’ là axis = 1
  - ‘minor_axis’ là axis = 2
  - ‘dtype’ là kiểu dữ liệu mỗi cột
  - ‘copy’ nhận giá trị True/False để khởi tạo dữ liệu có chia sẻ memory hay không

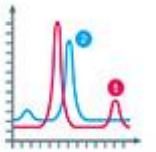
Tạo panel



```
import pandas as pd
import numpy as np
data = np.random.rand(2,3,4)
p = pd.Panel(data)
print(p)
```

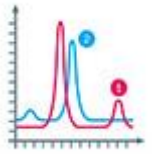
```
<class 'pandas.core.panel.Panel'>
Dimensions: 2 (items) x 3 (major_axis) x 4 (minor_axis)
Items axis: 0 to 1
Major_axis axis: 0 to 2
Minor_axis axis: 0 to 3
```

Tạo panel



```
p.to_frame()
```

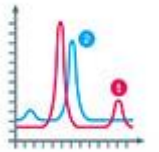
		0	1
major	minor		
0	0	0.335571	0.010409
	1	0.267106	0.843688
	2	0.840885	0.211749
	3	0.049653	0.722182
1	0	0.755207	0.282777
	1	0.674844	0.543207
	2	0.634314	0.433802
	3	0.290120	0.613040
2	0	0.322059	0.263548
	1	0.341035	0.702612
	2	0.634411	0.917126
	3	0.281678	0.809592



Phần 3

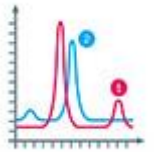
Chọn và nhóm phần tử

Chọn với iloc, loc và ix



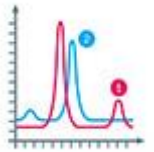
- Pandas có 3 phương pháp chọn phần tử
  1. Dùng iloc: chọn theo chỉ số hàng và cột
     - Cú pháp: data.iloc[<row selection>, <column selection>]
     - Tham số có thể là số nguyên, list các số nguyên, slice object với các số nguyên (ví dụ 2:7), mảng boolean,…
  2. Dùng loc: chọn theo nhãn hàng hoặc nhãn cột
     - Cú pháp: data.loc[<row selection>, <column selection>]
     - Tham số là nhãn (chứ không phải chỉ số)
  3. Dùng ix: lai giữa 2 cách trên, nếu truyền tham số là số nguyên thì nó làm việc như iloc, truyền kiểu giá trị khác thì nó làm việc như loc

Nhóm phần tử



```
df2 = pd.DataFrame({'X' : ['B', 'B', 'A', 'A'], 'Y' : [1, 2, 3, 4]})

df2.groupby(['X']).sum()
   Y
X
A  7
B  3

df2.groupby(['X'], sort=False).sum()
   Y
X
B  3
A  7
```

Nhóm phần tử



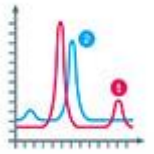
```
df3 = pd.DataFrame({'X' : ['A', 'B', 'A', 'B'], 'Y' : [1, 4, 3, 2]})
```

```
df3.groupby(['X']).get_group('A')
```

```
   X  Y
0  A  1
2  A  3
```

```
df3.groupby(['X']).get_group('B')
```

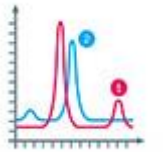
```
   X  Y
1  B  4
3  B  2
```



Phần 4

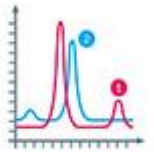
Sử dụng pandas trong bài toán thực tế

Dữ liệu kết quả xổ số



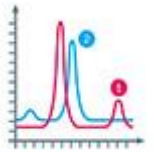
- Dữ liệu kết quả xổ số (độc đắc) từ ngày 1-1-2000 đến ngày 21-5-2018 (hôm qua)
- Lưu ở định dạng csv, 2 cột:
  - Cột 1: ngày ra số
  - Cột 2: số độc đắc
    - Dạng số (nếu không đủ 5 chữ số thì có nghĩa là đã bị xóa các chữ số 0 ở đầu)
    - Có thể không có dữ liệu (mỗi năm có 4 ngày không quay xổ số)
- Bài toán (vui + khoa học): phân tích các chiến lược chơi số đề mà người dân hay theo

Đọc và tiền xử lý dữ liệu



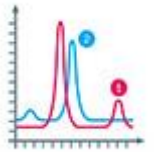
```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np

# đọc dữ liệu từ file csv, chuyển dữ liệu cột 1 sang date
df = pd.read_csv("kqxs.csv", index_col = 0, parse_dates=True)

# xóa bỏ các dòng không có dữ liệu
df.dropna(inplace=True)

# thêm cột mới là 2 số cuối của giải độc đắc
df['Cuoi'] = df.So % 100
```

Khảo sát dữ liệu



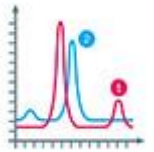
```
# trích xuất cột mới thành dữ liệu series để dễ xử lý
s = pd.Series(df.Cuoi, dtype='int64')

# xem phân bổ dữ liệu: biểu đồ histogram, 100 nhóm
s.plot('hist', bins=100)
plt.show()

# một dạng phân bổ dữ liệu khác: biểu đồ bar, đếm tần suất
s.value_counts().sort_index().plot('bar')
plt.show()
```

Viết hàm tính số tiền thu về



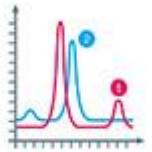
```
# thử bộ số myNums, kết quả về là result, số tiền chơi là money
def one_day(myNums, result, money):
      pay = len(myNums) * money
      get = money * 70 if result in myNums else 0
      return get-pay

# chơi nhiều ngày bộ số myNums, kết quả về là results
def many_day(myNums, results, money):
      total = 0
      for x in results:
            total += one_day(myNums, x, money)
      return total
```

Chiến lược: nuôi một số



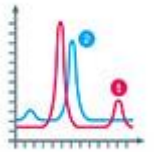
```
money = 1000

# thử chiến lược chơi: nuôi một con
print("Chơi con 76 toàn năm 2000:", many_day([76], s[0:367], money))
print("Chơi con 76 toàn bộ các năm:", many_day([76], s, money))

# thử chiến lược chơi: nuôi nhiều con
print("Nuôi nhiều số toàn năm 2000:", many_day([76, 92, 3, 10, 51,
45], s[0:367], money))
print("Nuôi nhiều số toàn bộ các năm:", many_day([76, 92, 3, 10, 51,
45], s, money))
```

Chiến lược: thống kê



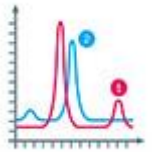
```
# thống kê con ra nhiều nhất rồi chơi
x = s[0:362].value_counts().idxmax()
y = s.value_counts().idxmax()
print("Chơi theo số ra nhiều nhất năm 2000:", x, many_day([x], s, money))
print("Chơi theo số ra nhiều nhất các năm:", y, many_day([y], s, money))
```

Chiến lược: ngẫu nhiên



```
# chơi ngẫu nhiên, mỗi ngày một con
total = 0
for d in s:
      total -= money
      m = np.random.randint(100)
      if (m == d): total += 70 * money

print("Chơi ngẫu nhiên:", total)
```

BẠN CÓ THỂ THỬ
VÀI CHIẾN LƯỢC THÔNG DỤNG KHÁC
VÀ LUÔN NHẬN ĐƯỢC KẾT LUẬN
CHƠI XỔ SỐ THÌ LUÔN THUA



Phần 5

Bài tập

Bài tập

Dựa trên bộ dữ liệu xổ số, hãy thử một vài chiến lược khác dưới đây:

1. Chơi ngẫu nhiên chẵn lẻ: mỗi lần đánh cả 50 số chẵn (hoặc 50 số lẻ), chọn ngẫu nhiên
2. Chơi nuôi đầu-cuối: chọn 1 chữ số, chẳng hạn số 7, đánh cả loạt các số có đầu và cuối chứa số 7 (07,17,27,..., 97, 70,71,...,79)
3. Chơi số xuất hiện ít nhất: thống kê xem số nào xuất hiện ít nhất từ ngày đầu tiên đến trước ngày mở thưởng thì chơi số đó