Python 应用之一:基本财务分析

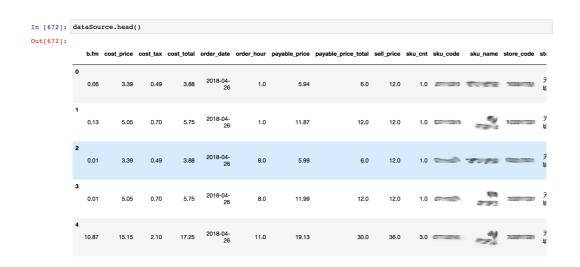
一、结果

每一种产品在某一周的工作日店均日均销售数量、销售成本、销售收入、销售毛利和销售毛利率

643]:						C1	
			avgSkuCnt	avgPayPriceTotal	avgCost	avgProfit	avgProfitHate
	weekNum	sku_name					
	17	a de la companya de l	21.60	181.37	89.44	91.92	50.68%
		64.5	25.02	269.68	148.12	121.57	45.08%
	18	***	15.87	136.11	61.57	74.54	54.76%
		F-0	19.34	205.11	111.02	94.08	45.87%
	19	-	16.57	141.01	65.23	75.78	53.74%
			20.70	222.75	118.98	103.77	46.59%
	20	センチョ	11.58	101.87	45.57	56.30	55.27%
			11.94	122.84	68.59	54.25	44.16%
			7.62	78.24	27.34	50.91	65.06%
	21	5-5-	6.75	58.42	26.57	31.85	54.53%

二、数据源

- 商品销售明细,包含字段:订单日期,商品名称,订单价格,商品销售数量,商品成本等

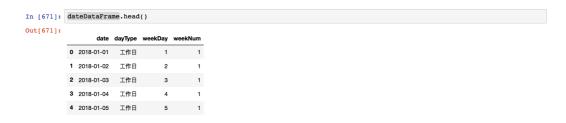


- 店铺明细,包含字段:店铺名称,开店日期,闭店日期等



三、其他数据

- 日期Mapping,包含字段:日期,日期处于第几周,日期是星期几,工作日/节假日



四、使用的模块

- pandas
- xlrd
- datetime
- time

五、涉及知识点

- 读取Excel文件里的数据并将数据转化为DataFrame
- 合并不同Excel文件的数据(相当于Vlookup)
- 清理DataFrame里的无效和重复字段
- 转换日期字符串为datetime对象
- 对数据进行聚合计算(相当于PivotTable)
- DataFrame转换index为column
- 调整输出数据的格式

知识点一: 读取Excel文件里的数据并将数据转化为DataFrame

```
def read excel workbook(path):
   workbook = xlrd.open workbook(path)
   sheet = workbook.sheet by index(0)
   data = \{\}
   for i in range(sheet.ncols):
       # 按列读取数据,第一个数据是列的的名字,剩下的数据是列的值
       h, *rest = sheet.col values(i)
       data[h] = rest
   frame = pd.DataFrame(data)
   return frame
知识点二:合并不同Excel文件的数据(相当于Vlookup)
dataSource = read excel workbook('/Volumes/Transcend/Blianlifeng/
专题数据/果汁/datasource.xlsx')
storeMapping = read excel workbook('/Volumes/Transcend/
Blianlifeng/专题数据/果汁/store Mapping.xlsx')
# pd.merge: 前两个参数是待合并数据, <how>规定了合并的类型, <on>规定了关联
的字段,这个字段同时存在于两个数据源,如果字段名不一致,可以用
<left on>,<right on>
dataSource = pd.merge(dataSource, storeMapping[['store code', '批
次']],how='inner',on='store code')
dataSource =
pd.merge(dataSource, dateDataFrame, how='inner', on=None, left on='or
der date',right on='date')
知识点三: 清理DataFrame里的无效和重复字段
dataSource.drop(['type','date'],axis='columns',inplace=True)
知识点四:转换日期字符串为datetime对象
def ChangeDateFormat(x):
   if x.count('/')>0:
       dt = time.strptime(x, '%Y/%m/%d')
   elif x.count('-')>0:
       dt = time.strptime(x,'%Y-%m-%d')
   dtStamp = time.mktime(dt)
   dtDateTime = datetime.datetime.fromtimestamp(dtStamp)
```

知识点五: 对数据进行聚合计算(相当于PivotTable)

```
sum1 = dataSource[dataSource['dayType']=='工作日']
[['cost_total','sku_cnt','payable_price_total']].groupby([dataSou
rce['weekNum'],dataSource['order_date'],dataSource['sku_name']]).
sum()
sum2 =
sum1[['avgSkuCnt','avgPayPriceTotal','avgCost']].groupby([sum1['w
eekNum'],sum1['sku_name']]).mean()
知识点六: DataFrame转换index为column
# 将合并计算后的DataFrame的index转化为column,然后与其他数据关联
sum1 =
pd.merge(sum1.reset index(),dateDataFrame[['date','storeCount']],
how='inner',left_on='order_date',right_on='date')
知识点七: 调整输出数据的格式
# 定义函数:
float format = lambda x : '%.2f' % x
percent format = lambda x : '%.2f%%' % (x * 100)
# DataFrame使用applymap, Series使用map
sum2[['avgSkuCnt','avgCost','avgPayPriceTotal','avgProfit']] =
sum2[['avgSkuCnt','avgCost','avgPayPriceTotal','avgProfit']].appl
ymap(float_format)
sum2['avgProfitRate'] = sum2['avgProfitRate'].map(percent format)
```