《从 excel 到 python 数据分析及可视化呈现》复习知识点

1、 会读取文件

2. 读写文本文件CSV

- 文本文件是一种由若干行字符构成的计算机文件，它是一种典型的顺序文件。
- csv是一种逗号分隔的文件格式，因为其分隔符不一定是逗号，又被称为字符分隔文件，文件以纯文本形式存储表格数据（数字和文本）。

1）读取方法

- 使用read_table来读取文本文件。

  pandas.read_table(filepath_or_buffer, sep='\t', header='infer', names=None, index_col=None, dtype=None, engine=None, nrows=None)

- 使用read_csv函数来读取csv文件。

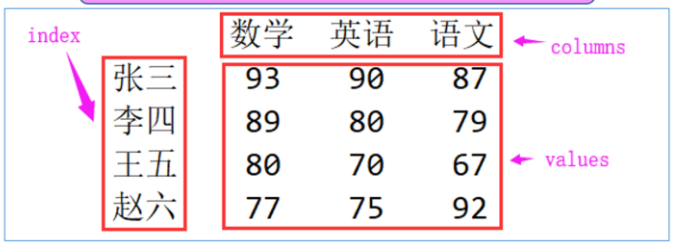
  pandas.read_csv(**filepath**_or_buffer, sep=',', **encoding**='gbk', header='infer', names=None, index_col=None, dtype=None, engine=None, nrows=None)

2、 会创建 DataFrame

2. DataFrame数据结构

每个DataFrame对象可以看作一个二维表格，由索引（index）、列名（columns）和值（value)三部分组成。

pd.DataFrame(data, index, columns)



3、 访问 DataFrame 数据、条件筛选、赋值

例子

loc访问器除了之前的作用外，还可以根据某个位置的True or False 来选定，如果某个位置的布尔值是True，则选定该row

```
data = pd.read_csv('yingyee.csv',encoding='gbk' )
data['交易额']
data[['姓名', '交易额' ]]
data.head(10)
data[1:3]
data[['姓名', '交易额']][1:3]
data.iloc[:4, 1:4]
data.loc[:4, '姓名':'交易额' ]
data.loc[(df[ '交易额' ]>1700) & (df[ '交易额' ]<1800), '等级' ] = '达标'
```

列访问

行访问

行列访问

很轻松嘛！

5）查询数据（1）

```
df[df['交易额']>1700]
df[(df['交易额']>1700) & (df['交易额']<1800)]
df[df['交易额']>1700]['交易额'].mean()
df[df['时段'] == '14：00-21：00']['交易额'].sum()
#张三下午的交易情况
df[(df['姓名'] == '张三') & (df['时段'] == '14：00-21：00')][:10]
#姓名是张三或者时间是下午的交易情况
df[(df['姓名'] == '张三') |(df['时段'] == '14：00-21：00')][:10]
```

不支持这样写：
1800>df['交易额']>1700，也不支持and
另外，注意条件中的括号也是必不可少

4、删除 DataFrame 中某行数据

8）删除某列或某行【‘数据’】

删除某列或某行数据需要用到pandas提供的方法**drop**，drop方法的用法如下。如果是删除某列，也可以直接用**del**

drop(labels, axis=0, inplace=False)

参数名称	说明
labels	接收string或array。代表删除的**行或列的标签**。无默认。
axis	接收0或1。代表操作的轴向。**默认为0。axis为0时表示删除行，axis为1时表示删除列。**
inplace	接收boolean。代表操作是否对原数据生效。默认为False。

删除列

```
df.drop(['total'], axis=1, inplace=True)
del df['total']#与上一行等价
df.columns
```

删除行

```
data.drop(data[data['交易额']<1000].index, inplace=True)
```

5、删除 DataFrame 中的空值

③ 处理缺失值

第一种方法：删除缺失值

删除法分为删除观测记录和删除特征两种，它属于利用减少样本量来换取信息完整度的一种方法，是一种最简单的缺失值处理方法。

pandas中提供了简便的删除缺失值的方法dropna，该方法既可以删除观测记录，亦可以删除特征。

DataFrame.dropna(axis=0, how='any', subset=None, inplace=False)

参数名称	说明
axis	接收0或1。表示轴向，0为删除观测记录（行），1为删除特征（列）。默认为0。
how	接收特定string。表示删除的形式。any表示只要有缺失值存在就执行删除操作。all表示当且仅当全部为缺失值时执行删除操作。默认为any。
subset	接收类array数据。表示进行去重的列/行。默认为None，表示所有列/行。
inplace	接收boolean。表示是否在原表上进行操作。默认为False。

6、为 DataFrame 添加数据列

7）为DataFrame增添数据列

DataFrame添加一列的方法非常简单，只需要**新建一个列索引**。并对该索引下的数据进行**赋值**操作即可。新增列的值都相同则直接赋一个常量值即可。

例如：

df ['total'] =5000

df['total'] = df['交易额']*5

7、对 DataFrame 进行排序

1）按不同标准对数据排序sort_values

sort_values(**by**, axis=0, **ascending**=True, **inplace**=False, na_position='last')

- 参数by用来指定依据哪个或哪些名字的列进行排序，如果只有一列则直接写出列名，**多列**的话需要放到**列表**中；
- 参数ascending=True表示升序排序，ascending=False表示降序排序，如果ascending设置为包含若干True/False的列表（必须与by指定的列表长度相等），可以为不同的列指定不同的顺序；
- 参数na_position用来指定把缺失值放在最前面（na_position='first'）还是最后面（na_position='last'）。

8、分组聚合

例子

- 可以使用agg方法一次求某数值列的中值与均值，如

```
df['交易额'].agg([np.median, np.mean])
df[['交易额', '工号']].agg(['median', 'mean'])
```

- 对于某个字段希望只做求均值操作，而对另一个字段则希望只做求和操作，可以使用字典的方式，将两个字段名分别作为key，求和与求均值的函数分别作为value，如

```
df.agg({'交易额': 'median', '工号': 'mean' })
df.agg({'交易额': ['median', 'sum'], '工号': 'mean' })
```

- 对分组结果进行聚合

```
df.groupby(by='姓名').agg({'交易额':['max', 'min', 'mean', 'median'], '日期':['max', 'min']})
```

9、绘制 matplotlib 图形，掌握折线图 plot 和 bar 图。会分析箱线图。

一、matplotlib 画图基本操作步骤

第一步、导入pyplot子库

```
import matplotlib.pyplot as plt
```

第三步：添加画布内容

第二部分是绘图的主体部分。即确定x轴和y轴内容。X、Y的数据类型可以是列表、元组、numpy数组等。

然后就可以用绘图函数绘图了。

其中添加标题，坐标轴名称，绘制图形等步骤是并列的，没有先后顺序，可以先绘制图形，也可以先添加各类标签。但是添加图例一定要在绘制图形之后。

常见绘图函数

如： plt.plot(x,y)

函数	描述
plot(x， y)	plot()是一个灵活的命令，它的参数可以是任意数量。可以用来绘制线型图
bar/barth(x, y)	柱状图
pie(x)	饼图
scatter(x, y)	散点图
boxplot(x)	箱线图
hist(x)	直方图
stackplot(x,y)	面积图
imshow(x)	热力图

给图添加元素内容

Xlim与ylim也可以直接在axis中设置
plt.axis([xmin, xmax, ymin, ymax])

函数名称	函数作用
plt.title	在当前图形中添加标题，可以指定标题的名称、位置、颜色、字体大小等参数。
plt.xlabel	在当前图形中添加x轴名称，可以指定位置、颜色、字体大小等参数。
plt.ylabel	在当前图形中添加y轴名称，可以指定位置、颜色、字体大小等参数。

plt.text()用法

plt.text()作用：画图时给图中的点 **加标签**

plt.text(x, y, s, fontsize, verticalalignment, horizontalalignment, rotation, **kwargs)

（1）x,y：标签添加的位置，注释文本内容所在位置的横/纵坐标

（2）s：标签名，字符串格式。例如str(y[0])或者'%d'%y[0]

第四步：保存与展示图形

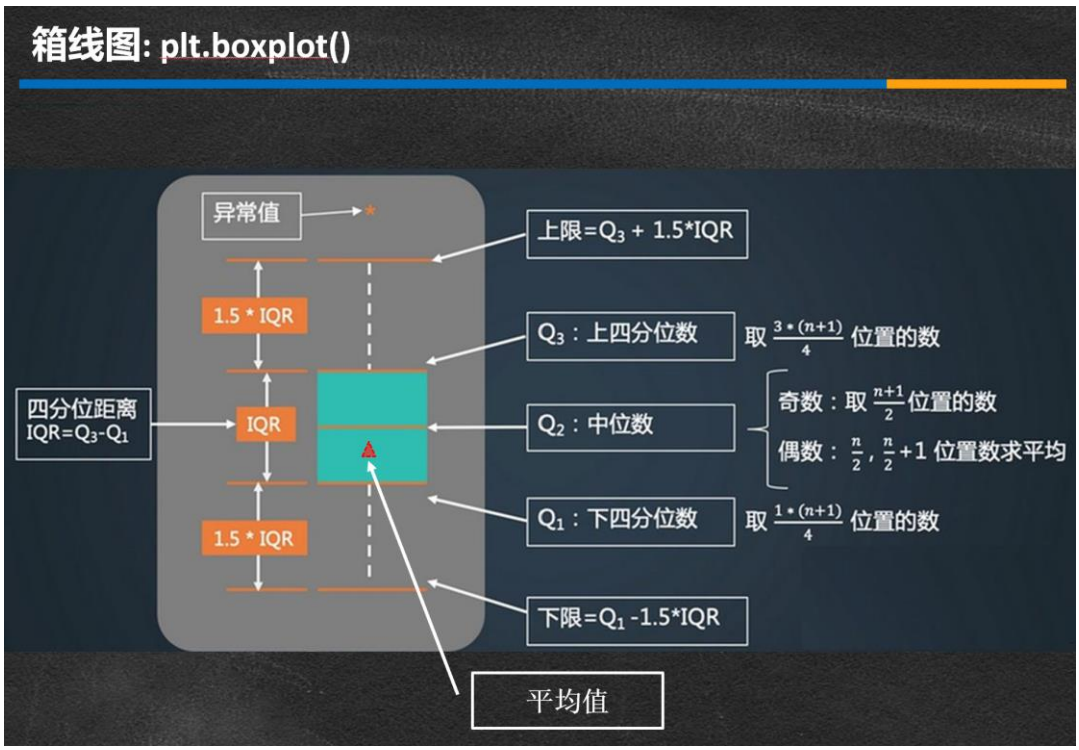
函数名称	函数作用
plt.savafig	保存绘制的图片，可以指定图片的分辨率、边缘的颜色等参数。
plt.show	在本机显示图形。

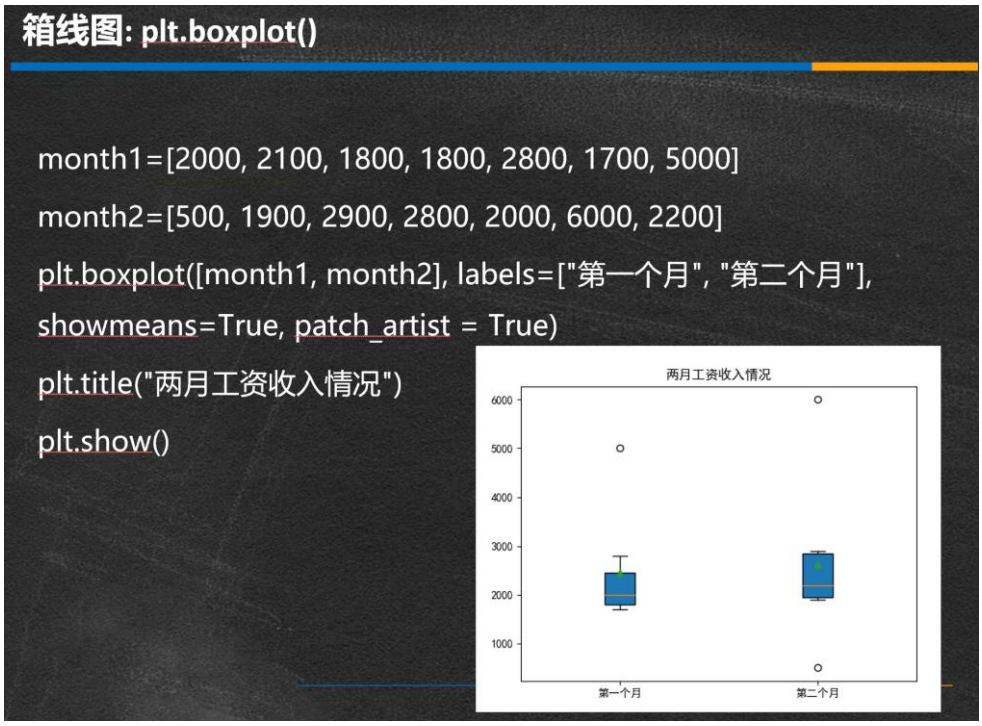
解决图中中文问题和负号问题

➢解决不显示中文

如果是mac，可以试试PingFang.ttc或者
['Arial Black ']或者
['Arial Unicode MS']

plt.rcParams['font.sans-serif'] = 'SimHei'





10、 绘制 pyecharts

第一步：

图表类型: from pyecharts.charts import *

函数	说明	函数	说明
Scatter	散点图	Funnel	漏斗图
Bar	柱状图	Gauge	仪表盘
Pie	饼图	Graph	关系图
Line	折线/面积图	Liquid	水球图
Radar	雷达图	Parallel	平行坐标系
Sankey	桑基图	Polar	极坐标系
WordCloud	词云图	HeatMap	热力图

第二步：

添加数据

条形图是
yaxis_data=y

- 散点图、折线图等二维数据图形可通过 .add_xaxis(xaxis_data=x)和.add_yaxis(series_name= ‘’ , y_axis=y)方法设置。加载y轴数据（可以多个）。
- 饼图等一维图形可通过.add(series_name= ‘’ , data_pair=[(i, j)for i, j in zip(lab, num)])方法设置参数
- pyecharts 所有方法均支持链式调用。

参数名在写的过程中可以省略，
.add_yaxis中的series_name的值不能省略，即使没有，也要用‘’写上

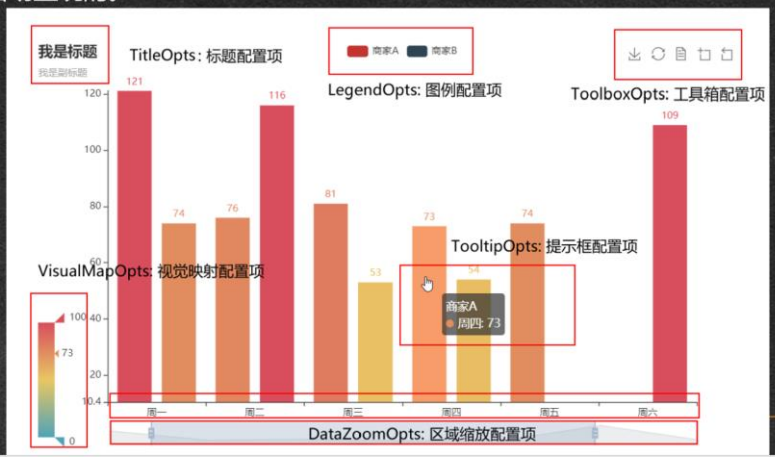
第三步：

显示、保存图表

- 图表.render()：默认将会在当前目录下生成一个 render.html 的文件，支持 path 参数，设置文件保存位置，如 render(r"e:\my_first_chart.html") 。
- 在Jupyter notebook中 直接调用图表.render_notebook ()随时随地渲染图表

全局配置组件：定制图表

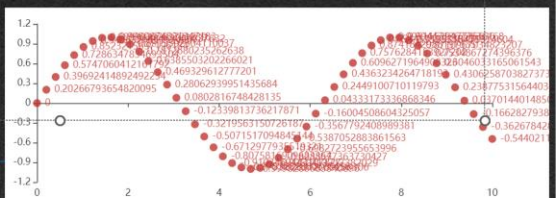
- 使用 options 配置项，在 pyecharts 中，一切皆 Options。使用的时要导入 import pyecharts.options as opts
- 全局配置项可通过 set_global_options 方法设置，其中图例、提示框配置项是自动显现的。



Pyecharts可视化案例代码演示

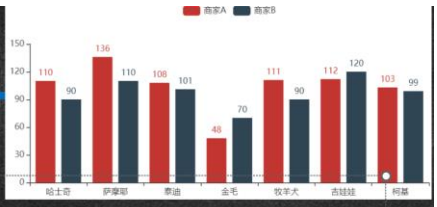
1、散点图

```
x = np.linspace(0, 10, 50)
y = np.sin(x)
scatter = (
        Scatter()
        .add_xaxis(xaxis_data=x)
        .add_yaxis(series_name='', y_axis=y
)
scatter.render_notebook()
```



Pyecharts可视化案例代码演示

4、柱状图




```
num = [110, 136, 108, 48, 111, 112, 103]
num2 = [90, 110, 101, 70, 90, 120, 99]
lab = ['哈士奇', '萨摩耶', '泰迪', '金毛', '牧羊犬', '吉娃娃', '柯基']
bar = (
    Bar()
    .add_xaxis(xaxis_data=lab)
    .add_yaxis(series_name='商家A', yaxis_data=num)
    .add_yaxis(series_name='商家B', yaxis_data=num2)
)
bar.render_notebook()
```