1.下列哪一项是对文本关键词进行可视化的方法？（A）

A标签云B短语网络C文本流D词语树

2.针对比例数据，研究者关心的是（B）

A各比例的顺序

B各比例的分布和相互关系

C各比例的变化

D各比例的大小

3.堆叠柱形图除了可以展示离散型时间数据，还可以展示（C）

A比例的变化情况

B数据随时间变化的趋势

C多个部分到整体的关系

D一个部分到整体的关系

4.对于关系数据，我们要做的是尝试着探索事物的（D）

A因果关系B线性关系C逻辑关系D相关关系

5.下列哪一项不属于导航的基本操作？（B）

A平移B指示C缩放D旋转

6.下列选项哪一个不是数据中所包含的信息（C）

A模式B关系C原理D异常

7.我们用散点图可以推断出变量间的（B）

A分布性B相关性C因果性D结构性

8.用于发现两个变量之间的数理关系的图是（B）

A散点图B拟合曲线C折线图D气泡图

9.（B）可一次完成统计分组和次数分配，其图形直观且保留原始信息，平均值、中位数和众数均可按照原始数据准确、方便地算出。

A密度图B茎叶图C饼图D直方图

10.如果数据存在子分类，而这些分类相加有意义的话，为了展示数据的数值差异，应该选择（A）

A堆叠柱形图B柱状图C堆叠折线图D直方图

11.下列那一项属于格式塔原则（D）

A间断原则B分组原则C相邻原则D闭合原则

12.高维多元数据中高维指数据具有多个（B）属性。

A互斥B独立C不相关D相关

13.对于折线图来说，合理设置横轴长度的原因是（B）

A使得折线剧烈变化，方便观察

B正确展示折线变化趋势

C正确设置横轴刻度

D规范图的大小

14.可以在二维平面上展示三个变量关系的单一图表是（D）

A散点图矩阵B散点图C堆叠面积图D气泡图

15.饼图不太适合表示精确数据的原因是（A）

A人眼对面积的大小不敏感

B饼图的角度不能精确展示数据

C饼图上只能标记比例，不能标记数据值

D对圆的设置会影响饼图对数据的展示

16.文档散是对文本的（B）进行可视化

A短语B关键词C句子D主题

17.以宽度相等的条形高度或长度的差异来显示统计单个指标数值多少或大小的一种图形是（B）

A茎状图B柱状图C直方图D堆叠柱状图

18.（D）允许同时看到多个单独变量的分布和它们两两之间的关系。

A气泡图B雷达图C平行坐标D散点矩阵图

19.几何体表达法对数据的不确定性可视化的缺点是（B）

A需要精心选择视觉元素才能有效表达不确定性

B易污染原有的确定性数据的可视化结果

C容易产生视觉混淆问题

D理解曲线较长，易引起疲劳

20.环形图采用（B）表示各类别的占比。

A角度B弧度C颜色D宽度

21.下列哪一个不属于高维多元数据可视化中的空间映射法？（A）

A雷达图B表格透镜C气泡图D平行坐标

22.下列哪一项不是定量或定序的视觉通道（D）

A面积B角度C长度D位置

23.下列哪一项是定性或分类的视觉通道（A）

A形状B包含C相似D直线的长度

24.矩形树图可以呈现树状结构数据的（C）关系。

A因果B同异C比例D关联

25.可视化的精髓与核心是（C）

A图表B视觉编码C数据D视觉通道

26.大数据分析和可视化中分而治之的思想没有体现在下列那一层？（B）

A统计分析层B数据采集层

C数据挖掘层D数据可视化层

27.对于带有连续时间属性的比例数据，可以选择（A）进行可视化。

A堆叠面积图B堆叠折线图

C堆叠柱形图D冲击图

28.下列哪个技术可用于解决由于空间距离拉大导致数据属性关联性降低的问题？（B）

A关联技术B导航技术C过滤技术D重配技术

29.数据不确定性来源于哪里？（A）

A以上全部B可视化C测量D建模

30.平行坐标的缺点是（C）

A数据维度大于四时，会由于视觉通道太多而降低可读性

B随着维度的扩展，子图会成指数增加，在有限的屏幕空间中会降低可读性

C难同时表现多个维度间的关系

D数据细节无法清楚展示

31.下列哪一项不属于文本信息层级？（C）

A语义级B语法级C语句级D词汇级

68.√暖色调颜色常常给人兴奋、愉快、活泼、亲切的感受，适用于积极、健康、努力等方面的表现内容。

69.√数据分布是指在统计分组的基础上，将总体中各单位按组归类整理，按一定顺序排列，形成的总体

70.√时间型数据包含时间属性，不仅要表达数据随时间变化的规律，还需表达数据分布的时间规律。

71.×视觉通道的表现力和有效性中的可辨性指不同视觉通道的编码对象放置到一起，是否容易分辨。

72.×不确定性可视化对确定性数据可视化结果没有负面影响。

73.×视觉的高级感知层次与物体性质相关，包括深度、形状、边界、表面材质等。

74.√数据不确定性可视化有助于帮助用户准确地理解数据并做出正确决策。

75.√交互技术可以帮助用户处理更庞大的数据，但是用户额外的时间成本也会增加。

76.×文本主题的抽取算法大致可分为两类：基于贝叶斯的非概率模型和基于矩阵分解的概率模型。

77.√关联技术可以通过交互的方式向用户展现数据之间的联系。

78.×分组的视觉通道适合于编码分类的数据信息。

79.×异常的数据一定都是错误的数据。

80.√密度图表现与数据值对应的边界或域对象的一种理论图形表示方法。

81.×视觉感知是把看的信息加以整合、解释、赋以意义的心理活动。

82.×数据中所包含的信息，归纳起来，主要有三种：变化、关系和分类。

83.×关系数据的相关性既能告诉人们事物为何产生，也可以给人们一个事物正在发生的提醒。

84.√在数据可视化中，通常并不区分区间型数据和比值型数据。

85.√冷色系颜色是以蓝色为中心的色群，这个色群常会给人造成寒冷、清爽、收缩的感受，适用于表现商业、科技、学习等方面的表现内容。

86.×一个好的可视化必须展示所有的信息内容，以保证用户一次获取全部数据信息。

87.×人们在获取视觉感知的时候，会倾向于将事物理解为组成该事物所有部分的集合，而不是将事物解为一个整体。

88.√数据间的关系大多可分为三类：数据间的比较，数据的构成，以及数据的分布或联系。

89.×可视化之前不需要将原始数据转换成用户可以理解的模式和特征并显示出来。

90.√气泡图的优势在于它便于我们在二维平面上一次比较3-4个变量。

91.×原始的文本数据无需处理，可以直接用来进行可视化。

92.√在数据的测量、清洗、建模、可视化过程中，都可能产生数据不确定性。

93.√人们天然对大自然的色彩变化感到舒适，所以应该更多地去应用自然中的色彩过渡。

94.×传统的可视化方法可以支持对复杂数据的分析。

95.√可视化系统的目的不是单纯地向用户传递信息，而是提供平台和工具让用户探索数据。

96.√如果两个变量之间存在某种相关性，那么散点图的大部分数据点就会相对密集并呈现出某种趋势。

97.×交互延时的长短对用户体验的效果没有太大影响，设计人员可以自行设置延时时间。

98.×视觉高级感知层次与物体性质相关。

99.√当数据维度非常高的时候，我们需要使用降维的方法降低数据维度，并且保证数据的特征不变。

100.√饼图用圆饼状方式显示一个数据系列中各项的大小、各项总和以及在整体中的比例。

101.×命名实体识别属于语义级信息的提取。

102.√一个有效的可视化是可以在短时间内把数据信息以用户容易理解的方式显示出来。

****1 .**** 列表是不可变对象，支持在原处修改。 （ ）否

****2 .**** 元组是不可变的，不能直接修改元组中元素的值，也不能为元组增删元素。（ ）是

****3 .**** Python使用lambda创建匿名函数，匿名函数拥有自己的命名空间。（ ）是

****4 .**** 同一个列表中的元素的数据类型可以各不相同。（ ）是

****5 .**** 列表、元组和字符串属于有序序列，其中的元素有严格的先后顺序。 （ ）是

****6 .**** 集合中的元素没有特定顺序但可以重复。（ ）否

****7 .**** 列表推导式在逻辑上等价于一个循环语句，只是形式上更加简洁是

****8 .**** 在Python中创建一个空集合，可以直接用 set1={} 。否

****9 .**** 列表、元组和字符串都支持双向索引，有效索引的范围为[-L,L]，L为列表、元组或字符串的长度。（ ）否

****10 .**** 包含列表的元组可以作为字典的键。（ ）否

****23 .**** 列表是不可变对象，支持在原处修改。（ ）否

****24 .**** 元组是不可变的，不能直接修改元组中元素的值，也不能为元组增删元素。（ ）是

****25 .**** Python使用lambda创建匿名函数，匿名函数拥有自己的命名空间。（ ）是

****26****同一个列表中的元素的数据类型可以各不相同。（ ）是

****27 .**** 集合中的元素没有特定顺序但可以重复。（ ）否

****28 .**** 列表推导式在逻辑上等价于一个循环语句，只是形式上更加简洁。（ ）是

****29 .**** 在Python中创建一个空集合，可以直接用 set1={}。否

****30 .**** 列表、元组和字符串都支持双向索引，有效索引的范围为[-L,L]，L为列表、元组或字符串的长度。（ ）否

****31 .**** 列表是包含0个或多个对象引用的有序序列，属于序列类型是

****32 .**** 字典是存储可变数量键值对的数据结构，键和值可以是任意类型数据结构。（ ）是

****1 .****创建Series时如果指定了index，则只能用index访问数据。否

****2 .**** 创建DataFrame时会自动加上索引，且全部列会被有序排列是

****3 .****Pandas中数据对象的索引可以随时被修改。 否

****4 .**** Pandas中数据的重建索引指对索引重新排序而不是修改。是

****5 .****交叉表是一种特殊的透视表，主要用于计算分组频率。是

****6 .****两个索引不一致的series进行算数运算会出错。否

****19 .****创建Series时如果指定了index，则只能用index访问数据。 否

****20 .**** Pandas中的apply方法能将函数应用于每一列。是

****20 .****Pandas中的apply方法能将函数应用于每一列。是

****22 .****Pandas中数据的重建索引指对索引重新排序而不是修改。是

****23 .**** 交叉表是一种特殊的透视表，主要用于计算分组频率。是

****24 .****两个索引不一致的series进行算数运算会出错。否

****25 .**** Pandas中可以通过行索引或行索引位置的切片形式选取行数据。是

****26 .****Pandas中可以通过query方法查询数据。是

****27 .**** Pandas中使用loc和isin两个函数配合使用，按指定条件对数据进行提取。是

****28 .**** 线性图一般用于描述两组数据之间的趋势。Pandas中的Plot方法默认绘制线形图。是

****3 .****Pandas中利用merge函数合并数据表时默认的是内连接方式是

****4 .**** Pandas中的描述性统计一般会包括缺失数据。 （ ）否

****5 .**** 语句dataframe.dropna(thresh=len(df)\*0.9,axis=1) 表示如果某列的缺失值超过90%则删除该列。 （ ）否

****6**** 利用merge方法合并数据时允许合并的DataFrame之间没有连接键。 （ ）否

****7 .****哑变量（Dummy Variables）又称虚拟变量，是用以反映质的属性的一个人工变量。（ ）是

****15 .**** Pandas中使用isnull().sum()可以统计缺失值。（ ）是

****16 .**** Pandas中的dropna中的thresh=N时表明要求一行有N个NaN值时该数据才能保留。（ ）否

****17 .**** Pandas中利用merge函数合并数据表时默认的是内连接方是

****18 .****Pandas中的描述性统计一般会包括缺失数据。（ ）否

****19 .****语句dataframe.dropna(thresh=len(df)\*0.9,axis=1) 表示如果某列的缺失值超过90%则删除该列。（ ）否

****20 .****利用merge方法合并数据时允许合并的DataFrame之间没有连接键。（ ）否

****22 .**** DataFrame的duplicates方法可以用来删除重复数据。否

****23 .****网络关联关系在大数据中是一种常见的关系。（ ）是

## **1. 写出引入numpy、matplotlib、pandas等库的代码 *(共1分)***

**import** numpy **as** np

**import** matplotlib.pyplot **as** plt

**import** pandas **as** pd

## **2. 完成下列要求代码： （*共4分*）**

(1)创建以下多维数组a: (1分)

array([[0, 1, 2, 3, 4], [5, 6, 7, 8, 9]])

创建以下多维数组b: (1分)

array([[1, 1, 1, 1, 1], [1, 1, 1, 1, 1]])

(3)将a,b数组合并为如下数组： (2分)

array([[0, 1, 2, 3, 4], [5, 6, 7, 8, 9], [1, 1, 1, 1, 1], [1, 1, 1, 1, 1]])

*# （1）作答开始*

a **=** np.array([[0, 1, 2, 3, 4],

[5, 6, 7, 8, 9]])

*# （1）作答结束*

*# （2）作答开始*

b **=** np.array([[1, 1, 1, 1, 1],

[1, 1, 1, 1, 1]])

*# （2）作答结束*

*# （3）作答开始*

result **=** np.concatenate((a, b), axis**=**0)

print(result)

*# （3）作答结束*

**3. 完成下列问题**

(1) 创建一个10\*10的ndarray对象，且矩阵边界全为1，里面全为0；

(2) 分别创建一个5\*3、3\*2矩阵，并计算其矩阵阵法； *(3分)*

(3) A=[1,1,1]、B=[2,2,2]、C=[3,3,3]，试计算((A+B)\*(A/2))的值；

*# （1）作答开始*

arr **=** np.zeros((10, 10))

arr[0] **=** 1

arr[**-**1] **=** 1

arr[:, 0] **=** 1

arr[:, **-**1] **=** 1

*# （1）作答结束*

*# （2）作答开始*

matrix1 **=** np.random.rand(5, 3)

matrix2 **=** np.random.rand(3, 2)

matrix\_multiply **=** np.dot(matrix1, matrix2)

*# （2）作答结束*

*# （3）作答开始*

A **=** np.array([1, 1, 1])

B **=** np.array([2, 2, 2])

result **=** ((A **+** B) **\*** (A **/** 2))

print(result)

*# （3）作答结束*

## **4. 创建满足下列要求多维数组，并回答问题：**

(1) shape为(100,200)，均值为10，标准差为5的正态分布; （*3分*）

(2) 在以上多维数组的基础上，计算其均值和方差，验证其是否完全符合题目要求，如果不符合，试说明其原因。（*3分*）

(3) 利用索引/切片技术从以上多维数组中提取前200个数据，并绘制其直方图，要求直方图中直方块的数量为50个。

*# (1)代码作答开始*

normal\_dist\_array **=** np.random.normal(loc**=**10, scale**=**5, size**=**(100, 200))

*# (1)代码作答结束*

*# (2) 代码作答开始*

mean\_value **=** np.mean(normal\_dist\_array)

variance\_value **=** np.var(normal\_dist\_array)

*# (2) 代码作答结束*

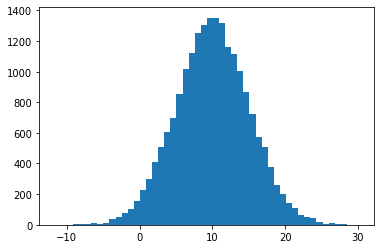
*# （3）代码作答开始*

subset\_array **=** normal\_dist\_array[:200].flatten()

plt.hist(subset\_array, bins**=**50)

plt.show()

*# (3) 代码作答结束*



**5. 使用随机数函数生成两个随机序列，并绘制其在平面直角坐标系下的散点图。*（共5分）***

*## 作答开始*

x **=** np.random.rand(50)

y **=** np.random.rand(50)

plt.scatter(x, y)

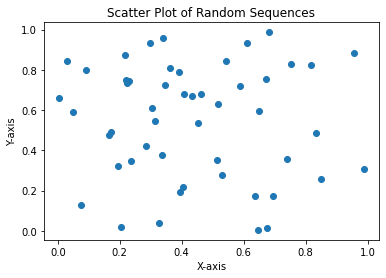
plt.xlabel('X-axis')

plt.ylabel('Y-axis')

plt.title('Scatter Plot of Random Sequences')

plt.show()

*## 作答结束*



**6. 完成以下题目**

**(1)创建如下图所示的DataFrame，并通过索引/切片方法求出北京和上海的环比数据：**



**(2) 在(1)的基础上，将DataFrame变还成如下图所示的多级索引，并通过多级索引的索引/切换方法求出北京和上海的环比数据：（*8分*）**



**（3）在（2）的基础上，计算所有城市统计指标的均值和中位数。（*3分*）**

*# （1）作答开始*

data **=** {

'环比': [101.5, 101.2, 101.3, 102.0, 100.1],

'同比': [120.7, 127.3, 119.4, 140.9, 101.4],

'定基': [121.4, 127.8, 120.0, 145.5, 101.6]

}

index **=** ['北京', '上海', '广州', '深圳', '沈阳']

df **=** pd.DataFrame(data, index**=**index)

*# 提取北京和上海的环比数据*

beijing\_shanghai **=** df.loc[['北京', '上海'], '环比']

print(beijing\_shanghai)

*# （1）作答结束*

*# （2）作答开始*

data **=** {

'北京': [101.5, 120.7, 121.4],

'上海': [101.2, 127.3, 127.8],

'广州': [101.3, 119.4, 120.0],

'深圳': [102.0, 140.9, 145.5],

'沈阳': [100.1, 101.4, 101.6]}

index **=** [['环比', '同比', '定基'], ['环比', '同比', '定基'], ['环比', '同比', '定基']]

columns **=** ['北京', '上海', '广州', '深圳', '沈阳']

df **=** pd.DataFrame(data, index**=**index, columns**=**columns)

*# 提取北京和上海的环比数据*

beijing\_shanghai\_multiindex **=** df.loc['环比', ['北京', '上海']]

print(beijing\_shanghai\_multiindex)

*# （2）作答结束*

*# （3）作答开始*

mean\_values **=** df.mean()

median\_values **=** df.median()

print("均值:\n", mean\_values)

print("\n中位数:\n", median\_values)

*# （3）作答结束*

# **6. 完成下列问题： *（共15分）***

(1) 创建一个类型为DatatimeIndex的序列(起始日期为今天的日期，时间频率为工作日); (3分)

(2) 以该序列作为索引，创建一个由随机数构成的DataFrame. (3分)

结果类似于下图:

A B C D

2019-12-23 1.331587 0.715279 -1.545400 -0.008384

2019-12-24 0.621336 -0.720086 0.265512 0.108549

2019-12-25 0.004291 -0.174600 0.433026 1.203037

2019-12-26 -0.965066 1.028274 0.228630 0.445138

2019-12-27 -1.136602 0.135137 1.484537 -1.079805

2019-12-30 -1.977728 -1.743372 0.266070 2.384967

2019-12-31 1.123691 1.672622 0.099149 1.397996

2020-01-01 -0.271248 0.613204 -0.267317 -0.549309

2020-01-02 0.132708 -0.476142 1.308473 0.195013

2020-01-03 0.400210 -0.337632 1.256472 -0.731970

(3) 选取'A', 'B'列数据 (3分)

(4) 选取如下图所示数据 (3分)

C D

2019-12-26 0.228630 0.445138

2019-12-27 1.484537 -1.079805

(5) 选取所有大于0的数据 (3分)

np.random.seed(10)

*# (1)作答开始*

start\_date **=** pd.Timestamp.today()

date\_index **=** pd.date\_range(start**=**start\_date, periods**=**10, freq**=**'B')

*# (1)作答结束*

*# (2)作答开始*

data **=** np.random.randn(10, 4)

df **=** pd.DataFrame(data, index**=**date\_index, columns**=**['A', 'B', 'C', 'D'])

*# (2)作答结束*

*# (3)作答开始*

selected\_columns **=** df[['A', 'B']]

*# (3)作答结束*

*# (4)作答开始*

specified\_dates **=** df.loc['2019-12-26':'2019-12-27', ['C', 'D']]

*# (4)作答结束*

*# (5)作答开始*

positive\_values **=** df[df **>** 0]

*# (5)作答结束*​

# **7. 泰坦尼克数据集简单分析，完成下列问题：**

(1) 读取****titanic.csv****数据集，并展示前10条数据；（*2分*）

(2) 从性别角度，统计出男女获救平均比例；（*3分*）

(3) 统计出不同性别、不同仓位获救人数的平均比例、人数、中位数；（*5分*）

(4) 从性别、年龄段(0-18, 18-80岁)角度统计获救的人数；（*5分*）

(5) ****使用DataFrame绘图方式****画出不同仓位获救人员年龄特征的箱线图.

*# (1)作答开始*

titanic\_df **=** pd.read\_csv('titanic.csv')

*# 展示前10条数据*

print(titanic\_df.head(10))

*# (1)作答结束*

*# (2)作答开始*

gender\_survival\_ratio **=** titanic\_df.groupby('sex')['survived'].mean()

print(gender\_survival\_ratio)

*# (2)作答结束*

*# (3)作答开始*

result **=** titanic\_df.groupby(['sex', 'pclass'])['survived'].agg(['mean', 'count', 'median'])

print(result)

*# (3)作答结束*

*# (4)作答开始*

age\_bins **=** [0, 18, 80]

age\_labels **=** ['0-18', '18-80']

titanic\_df['ageGroup'] **=** pd.cut(titanic\_df['age'], bins**=**age\_bins, labels**=**age\_labels)

age\_survival\_count **=** titanic\_df.groupby(['sex', 'ageGroup'])['survived'].sum()

print(age\_survival\_count)

*# (4)作答结束*

*# (5)作答开始*

**import** seaborn **as** sns

**import** matplotlib.pyplot **as** plt

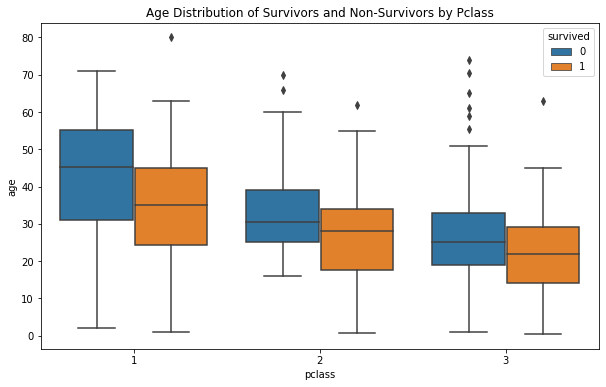
plt.figure(figsize**=**(10, 6))

sns.boxplot(x**=**'pclass', y**=**'age', hue**=**'survived', data**=**titanic\_df)

plt.title('Age Distribution of Survivors and Non-Survivors by Pclass')

plt.show()

*# (5)作答结束*



# **8. 完成下列问题：（*共20分*）**

(1) 读取****FremontBridge.csv****数据集(按小时统计经过美国西雅图弗莱蒙特桥上的自行车的数量)；（*1分*）

(2) 做初步的数据整理：修改更短的列名、增加Total列、删除缺失值、输出数据总体统计情况；（*4分*）

(3) 以'周'为统计单位，对数据进行重新采样，绘制其曲线图；（*5分*）

(4) 统计'每小时'自行车车流量均值，并绘制曲线图；（*5分*）

(5) 统计'工作日'、'周末'自行车流量均值，并绘制曲线图。（*5分*）

*# (1)作答开始*

fremont\_bridge\_df **=** pd.read\_csv('FremontBridge.csv')

*# (1)作答结束*

*# (2)作答开始*

fremont\_bridge\_df.columns **=** ['Date', 'West', 'East']

fremont\_bridge\_df['Total'] **=** fremont\_bridge\_df['West'] **+** fremont\_bridge\_df['East']

fremont\_bridge\_df.dropna(inplace**=True**)

print(fremont\_bridge\_df.describe())

*# (2)作答结束*

*# (3)作答开始*

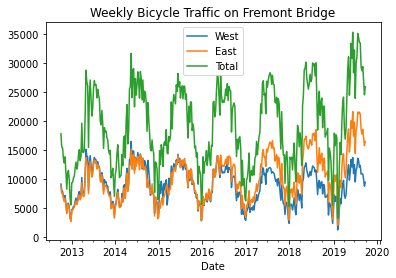
fremont\_bridge\_df['Date'] **=** pd.to\_datetime(fremont\_bridge\_df['Date'])

weekly\_resampled **=** fremont\_bridge\_df.resample('W-Mon', on**=**'Date').sum()

weekly\_resampled.plot(title**=**'Weekly Bicycle Traffic on Fremont Bridge')

plt.show()

*# (3)作答结束*



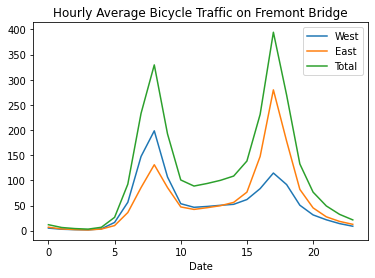
*# (4)作答开始*

hourly\_mean **=** fremont\_bridge\_df.groupby(fremont\_bridge\_df['Date'].dt.hour).mean()

hourly\_mean.plot(title**=**'Hourly Average Bicycle Traffic on Fremont Bridge')

plt.show()

*# (4)作答结束*



*# (5)作答开始*

fremont\_bridge\_df['Weekday'] **=** fremont\_bridge\_df['Date'].dt.weekday

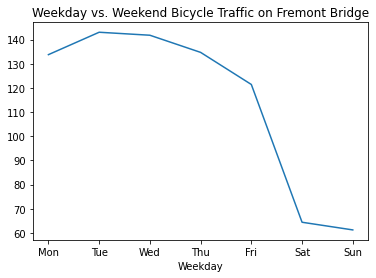
weekday\_weekend\_mean **=** fremont\_bridge\_df.groupby('Weekday')['Total'].mean()

weekday\_weekend\_mean.plot(title**=**'Weekday vs. Weekend Bicycle Traffic on Fremont Bridge')

plt.xticks(ticks**=**range(7), labels**=**['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun'])

plt.show()

*# (5)作答结束*



正余炫函数

import matplotlib.pyplot as plt

import matplotlib as mpl

import numpy as np

import pandas as pd

plt.rcParams['font.family'] = ['SimHei']

plt.rcParams['axes.unicode\_minus'] = False #负号显示的相关设置

# %matplotlib notebook

# %matplotlib inline

X = np.linspace(-np.pi, np.pi, 256,endpoint=True)

C,S = np.cos(X), np.sin(X)

plt.figure(figsize=(8,6))

plt.plot(X, C, color="blue", linewidth=2.5, linestyle="-", label="cosx")

plt.plot(X, S, color="red", linewidth=2.5, linestyle="-", label="sinx")

plt.legend(loc='upper left')

plt.xticks([-np.pi, -np.pi/2, 0, np.pi/2, np.pi],

[r'$-\pi$', r'$-\pi/2$', r'$0$', r'$+\pi/2$', r'$+\pi$'])

plt.yticks([-1, 0, +1],[r'$-1$', r'$0$', r'$+1$'])

t = 2\*np.pi/3

plt.plot([t,t],[0,np.cos(t)], color ='blue', linewidth=2.5, linestyle="--")

plt.scatter([t,],[np.cos(t),], 50, color ='blue')

plt.annotate(r'$sin(\frac{2\pi}{3})=\frac{\sqrt{3}}{2}$',

xy=(t, np.sin(t)), xycoords='data',

xytext=(+10, +30), textcoords='offset points', fontsize=16,

arrowprops=dict(arrowstyle="->", connectionstyle="arc3,rad=.2"))

plt.plot([t,t],[0,np.sin(t)], color ='red', linewidth=2.5, linestyle="--")

plt.scatter([t,],[np.sin(t),], 50, color ='red')

plt.annotate(r'$cos(\frac{2\pi}{3})=-\frac{1}{2}$',

xy=(t, np.cos(t)), xycoords='data',

xytext=(-90, -50), textcoords='offset points', fontsize=16,

arrowprops=dict(arrowstyle="->", connectionstyle="arc3,rad=.2"))

ax = plt.gca()

ax.spines['right'].set\_color('none')

ax.spines['top'].set\_color('none')

ax.xaxis.set\_ticks\_position('bottom')

ax.spines['bottom'].set\_position(('data',0))

ax.yaxis.set\_ticks\_position('left')

ax.spines['left'].set\_position(('data',0))

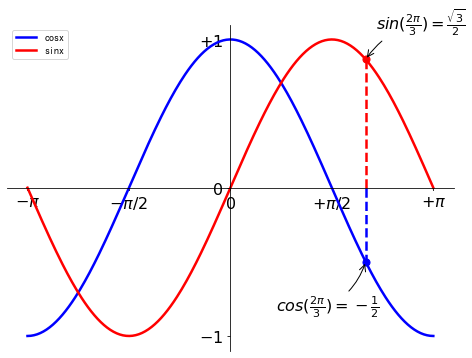
for label in ax.get\_xticklabels() + ax.get\_yticklabels():

label.set\_fontsize(16)

label.set\_bbox(dict(facecolor='white', edgecolor='None' ))

plt.savefig("绘图COS()&SIN().PNG",dpi=300)

plt.show()



Cos+sin

# 第01步：导入模块，并设置显示中文和负号的属性；

import matplotlib.pyplot as plt

import numpy as np

plt.rcParams['font.sans-serif']=['SimHei'] # 用于正常显示中文标签

plt.rcParams['axes.unicode\_minus']=False # 用来正常显示负号

# 第02步：创建x轴数据，从-pi到pi平均取256个点；

x = np.linspace(-np.pi,np.pi,256,endpoint=True) #获取x坐标

# 第03步：创建y轴数据，根据X的值，求正弦和余弦函数；

sin,cos = np.sin(x),np.cos(x) #获取y坐标

# 第04步：绘制正弦、余弦函数图，并将图形显示出来，设置正弦函数曲线的颜色为蓝色(blue)，线型为实线，线宽为2.5mm；余弦函数曲线的颜色为红色(red)，线型为实线，线宽为2.5mm。显示结果如下图所示；

plt.plot(x,sin,"b-",lw=2.5,label="正弦Sin()")

#X：x轴；sin:y轴；b-:color="blue",linestyle="-"的简写；lw:linewidth；label:线条的名称，可用于后面的图例

plt.plot(x,cos,"r-",lw=2.5,label="余弦Cos()") #cos:y轴；r-:color="red"；

plt.xlim(x.min()\*1.5,x.max()\*1.5)

plt.ylim(cos.min()\*1.5,cos.max()\*1.5)

plt.xticks([-np.pi,-np.pi/2,0,np.pi/2,np.pi],[r'$-\pi$',r'$-\pi/2$',r'$0$',r'$\pi/2$',r'$\pi$'])

plt.yticks([-1,0,1])

plt.title("绘图实例之COS()&SIN()",fontsize=16,color="green")

plt.text(+2.1,-1.4,"python\_matplotlib",fontsize=16,color="purple")

ax=plt.gca() #获取Axes对象

ax.spines['right'].set\_color('none') #隐藏右边界

ax.spines['top'].set\_color('none') #隐藏上边界

ax.xaxis.set\_ticks\_position('bottom') #x轴坐标刻度设置在坐标轴下面

ax.spines['bottom'].set\_position(('data',0)) #x轴坐标轴平移至经过零点（0,0）位置

ax.yaxis.set\_ticks\_position('left') #y轴坐标刻度设置在坐标轴下面

ax.spines['left'].set\_position(('data',0)) #y轴坐标轴平移至经过零点（0,0）位置

plt.legend(loc="upper left",fontsize=12)

t1 = 2\*np.pi/3 #设定第一个点的x轴值

t2 = -np.pi #设定第二个点的x轴值

plt.plot([t1,t1],[0,np.sin(t1)],color ='b',linewidth=1.5,linestyle="--")

#第一个列表是x轴坐标值，第二个列表是y轴坐标值

#这两个点坐标分别为（t1,0）和（t1，np.sin(t1)），根据两点画直线

plt.plot([t2,t2],[0,np.cos(t2)],color ='r',linewidth=1.5,linestyle="--")

#这两个点坐标分别为（t2,0）和（t2，np.cos(t2)），根据两点画直线

plt.scatter([t1,],[np.sin(t1),], 50, color ='b')

plt.scatter([t2,],[np.cos(t2),], 50, color ='r')

plt.annotate(r'$\sin(\frac{2\pi}{3})=\frac{\sqrt{3}}{2}$',

xy=(t1,np.sin(t1)), #点的位置

xycoords='data', #注释文字的偏移量

xytext=(+10,+30), #文字离点的横纵距离

textcoords='offset points',

fontsize=14, #注释的大小

arrowprops=dict(arrowstyle="->",connectionstyle="arc3,rad=.2")) #箭头指向的弯曲度

plt.annotate(r'$\cos(-\pi)=-1$',

xy=(t2,np.cos(t2)), #点的位置

xycoords='data', #注释文字的偏移量

xytext=(0,-40), #文字离点的横纵距离

textcoords='offset points',

fontsize=14, #注释的大小

arrowprops=dict(arrowstyle="->",connectionstyle="arc3,rad=.2")) #箭头指向的弯曲度

for label in ax.get\_xticklabels()+ax.get\_yticklabels(): #获取刻度

label.set\_fontsize(18) #设置刻度字体大小

plt.fill\_between(x,np.abs(x)<0.5,sin,sin>0.5,color='g',alpha=0.8)

#设置正弦函数的填充区域，颜色为绿色（green），其中的一种方式

plt.fill\_between(x,cos,where=(-2.5<x)&(x<-0.5),color='purple')

#设置余弦函数的填充区域，颜色为紫色（purple），另外一种方式

plt.grid()

plt.savefig("绘图实例之COS()&SIN().PNG",dpi=300)

plt.show() #显示图表

