南昌大学实验报告

学生姓名：丁俊　　学　　号：8003119100　　专业班级：信息安全 193　班

实验类型：☐ 验证■ 综合 ☐ 设计 ☐ 创新　实验日期：2021.12.2　实验成绩：_____

一、实验项目名称

数据探索与可视化

二、实验目的

1. 掌握描述性统计分析方法
2. 掌握 pandas 进行数据聚合和数据探索
3. 掌握 pandas 进行数据可视化（折线图、柱状图、饼图、直方图、箱型图、散点图和热力图等）

三、实验任务

参照电商数据分析案例完成数据探索与数据可视化

实验目的：通过对某医院的药品销售情况进行分析，了解该医院患者的月消费次数、月均消费金额、客单价以及消费趋势、需求量前几位的药品等。

四、主要仪器设备及耗材

软件：Anaconda 或者 pycharm 等

五、实验步骤

数据分析的基本过程包括：获取数据、数据清洗、构建模型、数据可视化分析以及消费趋势分析。

	A	B	C	D	E	F	G
1	购药时间	社保卡号	商品编码	商品名称	销售数量	应收金额	实收金额
2	2018-01-0	001616528	236701	强力VC银翘	6	82.8	69
3	2018-01-0	001616528	236701	清热解毒口	1	28	24.64
4	2018-01-0	001260282	236701	感康	2	16.8	15
5	2018-01-1	001007034	236701	三九感冒灵	1	28	28
6	2018-01-1	001015543	236701	三九感冒灵	8	224	208
7	2018-01-2	001338952	236701	三九感冒灵	1	28	28
8	2018-01-3	001014649	236701	三九感冒灵	2	56	56
9	2018-02-1	001117732	236701	三九感冒灵	5	149	131.12
10	2018-02-2	001006568	236701	三九感冒灵	1	29.8	26.22
11	2018-02-2	001338952	236701	三九感冒灵	4	119.2	104.89
12	2018-03-0	001002638	236701	三九感冒灵	2	59.6	59.6
13	2018-03-0	001022850	236701	三九感冒灵	3	84	84
14	2018-03-0	001007740	236701	清热解毒口	1	28	24.64
15	2018-03-0	001007740	236701	清热解毒口	5	140	112
16	2018-03-0	001007984	236701	清热解毒口	6	168	140
17	2018-03-1	001003132	236701	清热解毒口	2	56	49.28
18	2018-03-1	001007034	236701	清热解毒口	2	56	49.28
19	2018-03-1	001071232	236701	清热解毒口	5	140	112
20	2018-03-2	001166882	236701	清热解毒口	6	168	140
21	2018-03-2	001006638	236701	清热解毒口	1	28	28
22	2018-03-2	001021333	236701	清热解毒口	6	168	140
23	2018-03-2	001007887	236701	清热解毒口	6	168	140
24	2018-03-2	001019246	236701	清热解毒口	1	28	28
25	2018-03-2	001007523	236701	清热解毒口	6	168	140

一、 获取数据

数据是存在 Excel 中的，可以使用 pandas 的 Excel 读取文件函数将数据读到内存中。

```
## 1、获取和理解数据
```

In [3]:
```
import pandas as pd
fileName = 'F:\\LECTURE\\大三作业\\python作业\\2018出售记录.xlsx'

salesDF = pd.read_excel(fileName, sheet_name = 'Sheet1', dtype = str)
salesDF.head(10)
```

二、 查看数据整体情况

1、查看数据量：salesDF.shape，总共有 6578 行、7 列数据

In [5]:
```
# 数据的总行数和列数
salesDF.shape
```

Out[5]: (6578, 7)

2、查看数据前 10 行

```
salesDF.head(10)
```

Out[3]:

	购药时间	社保卡号	商品编码	商品名称	销售数量	应收金额	实收金额
0	2018-01-01 星期五	001616528	236701	强力VC银翘片	6	82.8	69
1	2018-01-02 星期六	001616528	236701	清热解毒口服液	1	28	24.64
2	2018-01-06 星期三	0012602828	236701	感康	2	16.8	15
3	2018-01-11 星期一	0010070343428	236701	三九感冒灵	1	28	28
4	2018-01-15 星期五	00101554328	236701	三九感冒灵	8	224	208
5	2018-01-20 星期三	0013389528	236701	三九感冒灵	1	28	28
6	2018-01-31 星期日	00101464928	236701	三九感冒灵	2	56	56
7	2018-02-17 星期三	0011177328	236701	三九感冒灵	5	149	131.12
8	2018-02-22 星期一	0010065687828	236701	三九感冒灵	1	29.8	26.22
9	2018-02-24 星期三	0013389528	236701	三九感冒灵	4	119.2	104.89

3、看下数据的大概分布：salesDF.info()、salesDF.describe()

从数据的分布情况可以看出，总共行数为 6578 行，这里所有的字段都是 6577 行，说明数据中有一行空行，且“销售时间”、“社保卡号”有缺失数据。

```
]: # 数据分布和情况
salesDF.info()
#
```

可以看出数据中一行空行、且01列有缺失数据

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6578 entries, 0 to 6577
Data columns (total 7 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   销售时间    6576 non-null   object
 1   社保卡号    6576 non-null   object
 2   商品编码    6577 non-null   object
 3   商品名称    6577 non-null   object
 4   销售数量    6577 non-null   object
 5   应收金额    6577 non-null   object
 6   实收金额    6577 non-null   object
dtypes: object(7)
memory usage: 359.9+ KB
```

In [9]:

```
salesDF.describe()
'''
top 最常见的值
freq 最常见的值的频次
unique 不相同的个数
'''
```

Out[9]:

	销售时间	社保卡号	商品编码	商品名称	销售数量	应收金额	实收金额
count	6576	6576	6577	6577	6577	6577	6577
unique	202	2426	85	78	28	443	774
top	2018-04-15 星期五	001616528	2367011	苯磺酸氨氯地平片(安内真)	2	56	50
freq	228	253	622	899	3345	361	215

4、数据描述性分析

```
[22]: salesDF.describe()
```

:[22]:

	销售数量	应收金额	实收金额
count	6552.000000	6552.00000	6552.00000
mean	2.384158	50.43025	50.43025
std	2.374754	87.68075	87.68075
min	-10.000000	-374.00000	-374.00000
25%	1.000000	14.00000	14.00000
50%	2.000000	28.00000	28.00000
75%	2.000000	59.60000	59.60000
max	50.000000	2950.00000	2950.00000

count 表示此列有多少有效值；unique 不同的值有多少个；std 是数据标准差；
25%表示四分之一位数；50%表示二分之一位数；75%表示四分之三位数；mean 是均值。

三、数据清洗和分析

1、列名重命名

使用 rename 函数将修改第一列数据为“销售时间”。

```
[7]: # 列名重命名
colNameDict = {'购药时间':'销售时间'}
salesDF.rename(columns = colNameDict, inplace=True)
'''
inplace=False,数据框本身不会改变，新创建一个改动后的数据
inplace=True,数据本身会改变
'''
salesDF.head()
```

[7]:

	销售时间	社保卡号	商品编码	商品名称	销售数量	应收金额	实收金额
0	2018-01-01 星期五	001616528	236701	强力VC银翘片	6	82.8	69
1	2018-01-02 星期六	001616528	236701	清热解毒口服液	1	28	24.64
2	2018-01-06 星期三	0012602828	236701	感康	2	16.8	15
3	2018-01-11 星期一	0010070343428	236701	三九感冒灵	1	28	28
4	2018-01-15 星期五	00101554328	236701	三九感冒灵	8	224	208

2、缺失值处理

获取的数据中可能存在缺失值，通过查看基本信息可以推断出“销售时间”和“社保卡号”这两列有缺失值，如果不处理这些缺失值会干扰后面的数据分析结果。
这里直接用 dropna 函数删除缺失数据。

2、数据处理

```
In [10]: print('删除缺失值前', salesDF.shape)
         salesDF = salesDF.dropna(subset = ['销售时间', '社保卡号'], how = 'any')
         print('删除缺失值后', salesDF.shape)
```

```
删除缺失值前 (6578, 7)
删除缺失值后 (6575, 7)
```

```
In [11]: # 删除缺失值后，用reset_index重置索引
         salesDF = salesDF.reset_index(drop = True)
```

3、数据类型转换

在导入数据时为了防止导入不进来，会默认所有数据都是 object 类型，但是实际上“销售数量”、“应收金额”、“实收金额”这些数据是 float 浮点型，因此需要对数据进行类型转换。可使用 astype 转换数据类型。

```
In [12]: # 数据类型转换
         salesDF['销售数量'] = salesDF['销售数量'].astype('float')
         salesDF['应收金额'] = salesDF['应收金额'].astype('float')
         salesDF['实收金额'] = salesDF['应收金额'].astype('float')
         print('转换后的数据类型\n', salesDF.dtypes)
```

```
转换后的数据类型
 销售时间      object
社保卡号       object
商品编码       object
商品名称       object
销售数量      float64
应收金额      float64
实收金额      float64
dtype: object
```

4、日期格式转换

在“销售时间”这一列数据中存在星期这一类的数据，但在数据分析中根本用不到，因此把销售日期和星期用 split 函数分割，分割后的数据是 Series 类型。

```
In [13]: # 转换日期数据类型
         def splitSaletime(timeCol):
             timeList = []
             for value in timeCol:
                 dateStr = value.split(' ')[0]
                 timeList.append(dateStr)
             # 将列表转为一维数组Series类型
             timeSer = pd.Series(timeList)
             return timeSer
```

```
In [15]: # 获取销售时间列
         timeSer = salesDF.loc[:,'销售时间']
         dateStr = splitSaletime(timeSer)
         dateStr[0:5]
```

```
Out[15]: 0    2018-01-01
         1    2018-01-02
         2    2018-01-06
         3    2018-01-11
         4    2018-01-15
         dtype: object
```

```
[19]: # 字符串转换日期
      # errors = 'coerce'如果原始数据不符合日期的格式，转换后的值为空值NaN
      salesDF.loc[:,'销售时间'] = pd.to_datetime(salesDF.loc[:,'销售时间'], format = '%Y-%m-%d', errors = 'coerce')
      salesDF.dtypes

ut[19]: 销售时间    datetime64[ns]
        社保卡号            object
        商品编码            object
        商品名称            object
        销售数量           float64
        应收金额           float64
        实收金额           float64
        dtype: object

[20]: salesDF = salesDF.dropna(subset=['销售时间','社保卡号'], how = 'any')
```

把字符串转成日期 datetime 格式，默认不符合日期的格式的字符串置为空值，最后再将空值 dropna 删除。

5、数据排序

此时时间是没有按顺序排列的，所以还是需要排序一下，排序之后索引会被打乱，所以也需要重置一下索引。其中 by:表示按哪一列进行排序，ascending=True 表示升序排列，ascending=False 表示降序排列。

按出售日期排序

```
In [21]: # 数据排序
         # 按销售时间排序
         salesDF = salesDF.sort_values(by='销售时间',ascending=True,na_position='first')
         print('排序后---')
         salesDF.head()
```

排序后---

Out[21]:

	销售时间	社保卡号	商品编码	商品名称	销售数量	应收金额	实收金额
0	2018-01-01	001616528	236701	强力VC银翘片	6.0	82.8	82.8
1475	2018-01-01	00107891628	861456	酒石酸美托洛尔片(倍他乐克)	2.0	14.0	14.0
1306	2018-01-01	001616528	861417	雷米普利片(瑞素坦)	1.0	28.5	28.5
3859	2018-01-01	0010073966328	866634	硝苯地平控释片(欣然)	6.0	111.0	111.0
3888	2018-01-01	0010014289328	866851	缬沙坦分散片(易达乐)	1.0	26.0	26.0

6、异常值处理

```
[22]: salesDF.describe()
```

:[22]:

	销售数量	应收金额	实收金额
count	6552.000000	6552.00000	6552.00000
mean	2.384158	50.43025	50.43025
std	2.374754	87.68075	87.68075
min	-10.000000	-374.00000	-374.00000
25%	1.000000	14.00000	14.00000
50%	2.000000	28.00000	28.00000
75%	2.000000	59.60000	59.60000
max	50.000000	2950.00000	2950.00000

销售数量、金额不能小于 0。通过描述统计信息可以看到，“销售数量”、“应收金额”、“实收金额”这三列数据的最小值出现了负数，这明显不符合常理，数据中存在异常值的干扰，因此要对数据进一步处理，以排除异常值的影响：

```
In [30]: # 删除负值、异常值记录
queryStr = salesDF.loc[:,'销售数量'] >= 1
print('删除异常值前', salesDF.shape)
salesDF= salesDF.loc[queryStr,:]
print('删除异常值后', salesDF.shape)
```

```
删除异常值前 (6552, 7)
删除异常值后 (6509, 7)
```

```
In [31]: salesDF.describe()
```

Out[31]:

	销售数量	应收金额	实收金额
count	6509.000000	6509.000000	6509.000000
mean	2.405285	50.908726	50.908726
std	2.364095	87.634645	87.634645
min	1.000000	1.200000	1.200000
25%	1.000000	14.000000	14.000000
50%	2.000000	28.000000	28.000000
75%	2.000000	59.600000	59.600000
max	50.000000	2950.000000	2950.000000

没有负值记录

七、总结数据

```
salesDF['社保卡号'].unique().size
```

```
2410
```

```
salesDF['商品编码'].unique().size
```

```
85
```

```
salesDF['商品名称'].unique().size
```

```
78
```

从代码结果可以看出，有 2410 个不同的人来药店买药，总共有 78 种药被出售，但是相同药的商品编码不一定相同。

四、数据可视化分析

月均消费次数
月均消费次数 = 总消费次数/月份数，同一天、同一个人发送的所有消费算作一次消费。
计算总消费次数

```
# 删除重复数据
# drop_duplicates去重函数
kpli_Df = salesDF.drop_duplicates(subset=['销售时间', '社保卡号'])
total = kpli_Df.shape[0]
print('总消费次数=', total)
```

```
总消费次数= 5345
```

计算月份数

```
# 计算月份数
kpli_Df = kpli_Df.sort_values(by='销售时间',ascending=True)
# 重命名行index
kpli_Df = kpli_Df.reset_index(drop=True)
# 最小时间
starttime = kpli_Df.loc[0,'销售时间']
# 最大时间
endtime = kpli_Df.loc[total-1,'销售时间']
# 天数
days = (endtime - starttime).days
months = days // 30
print('月份数', months)
```

```
月份数 6
```

计算月均消费次数

In [52]:
```
# 计算月均消费次数
kpi1_I = total // months
print('月消费次数:', kpi1_I)
```

```
月消费次数: 890
```

计算月均消费金额

In [53]:
```
# 月均消费金额 = 总消费金额 / 月份数
totalMoney = salesDF.loc[:,'实收金额'].sum()
monthMoney = totalMoney // months
print('月消费金额:',monthMoney)
```

```
月消费金额: 55227.0
```

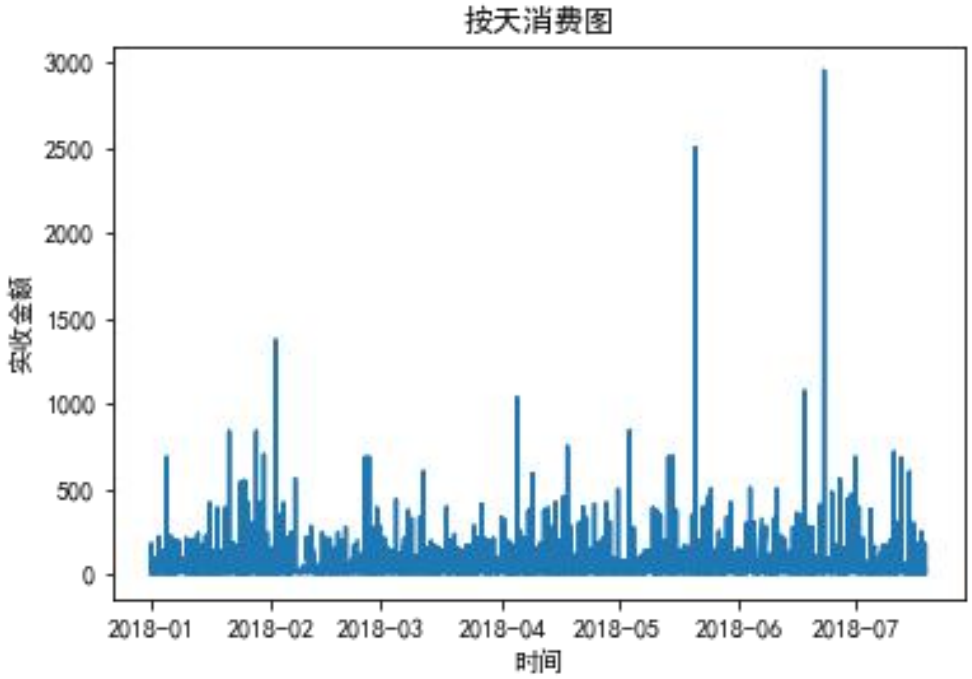
1、分析每天的消费金额

[56]:
```
# 每天的消费金额
import matplotlib.pyplot as plt
import matplotlib
from pylab import mpl
mpl.rcParams['font.sans-serif'] = ['SimHei']
# 先复制一份数据，防止影响干净的数据
groupDf = salesDF
# 让销售时间所在列的值当作行index
groupDf.index = groupDf['销售时间']
groupDf.head()
```

t[56]:

销售时间	销售时间	社保卡号	商品编码	商品名称	销售数量	应收金额	实收金额
2018-01-01	2018-01-01	001616528	236701	强力VC银翘片	6.0	82.8	82.8
2018-01-01	2018-01-01	00107891628	861456	酒石酸美托洛尔片(倍他乐克)	2.0	14.0	14.0
2018-01-01	2018-01-01	001616528	861417	雷米普利片(瑞素坦)	1.0	28.5	28.5
2018-01-01	2018-01-01	0010073966328	866634	硝苯地平控释片(欣然)	6.0	111.0	111.0
2018-01-01	2018-01-01	0010014289328	866851	缬沙坦分散片(易达乐)	1.0	26.0	26.0

[58]:
```
# 画图
plt.plot(groupDf['实收金额'])
plt.title('按天消费图')
plt.xlabel('时间')
plt.ylabel('实收金额')
plt.show()
```



从结果可以看出，每天消费总额差异较大，除了个别天出现的高额消费，大多数时间消费情况维持在 500 元之内。

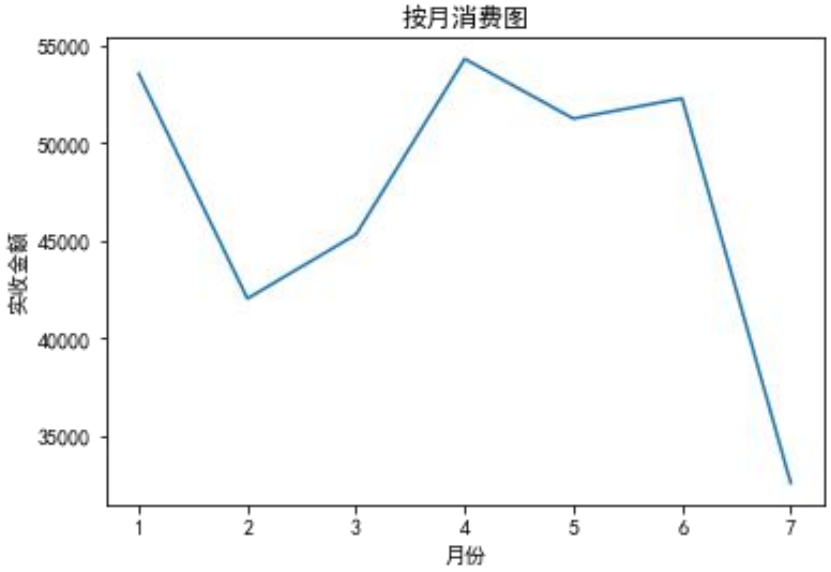
2、分析每月的消费金额

将数据集按月分组聚合，并计算每个月的消费总额

In [62]:
```
# 分析每月的消费金额
# groupDf.head(10)
gm = groupDf.groupby(groupDf.index.month) # 将销售时间聚合按月分组
monthDf = gm.sum()
print(monthDf)
```

销售时间	销售数量	应收金额	实收金额
1	2527.0	53561.6	53561.6
2	1858.0	42028.8	42028.8
3	2225.0	45318.0	45318.0
4	3010.0	54324.3	54324.3
5	2225.0	51263.4	51263.4
6	2328.0	52300.8	52300.8
7	1483.0	32568.0	32568.0

3]:
```
# 月消费金额图
plt.plot(monthDf['实收金额'])
plt.title('按月消费图')
plt.xlabel('月份')
plt.ylabel('实收金额')
plt.show()
```



结果显示，7 月消费金额最少，因为 7 月数据不完整，不具参考价值。1 月、4 月、5 月和 6 月的月消费金额差异不大，2 月和 3 月的消费金额迅速降低，这可能是 2 月和 3 月处于春节期间，大部分人都回家过年的原因。

3、分析药品销售情况

对“商品名称”和“销售数量”这两列数据进行聚合为 Series 形式，按降序排序。

[64]:
```
# 分析药品销售情况
medicine = groupDf[['商品名称', '销售数量']]
re_medicine = medicine.groupby('商品名称')[['销售数量']].sum()

# 对药品销售数量按降序排列
re_medicine = re_medicine.sort_values(by='销售数量',ascending=False)
re_medicine.head()
```

Out[64]:

商品名称	销售数量
苯磺酸氨氯地平片(安内真)	1781.0
开博通	1440.0
酒石酸美托洛尔片(倍他乐克)	1140.0
硝苯地平片(心痛定)	825.0
苯磺酸氨氯地平片(络活喜)	796.0

获取销售数量前十的商品，用条形图和饼状图展示。

```
top_medicine = re_medicine.iloc[:10,:]
print(top_medicine)
```

```
                          销售数量
商品名称
苯磺酸氨氯地平片(安内真)        1781.0
开博通                    1440.0
酒石酸美托洛尔片(倍他乐克)       1140.0
硝苯地平片(心痛定)             825.0
苯磺酸氨氯地平片(络活喜)         796.0
复方利血平片(复方降压片)         515.0
G琥珀酸美托洛尔缓释片(倍他乐克)    509.0
缬沙坦胶囊(代文)              445.0
非洛地平缓释片(波依定)           375.0
高特灵                    371.0
```

记录销售前十的药品

```
top_medicine.plot(kind='bar')
plt.title('药品前十销售情况')
plt.xlabel('药品种类')
plt.ylabel('药品数量')
plt.legend(loc = 0)
plt.show()
```

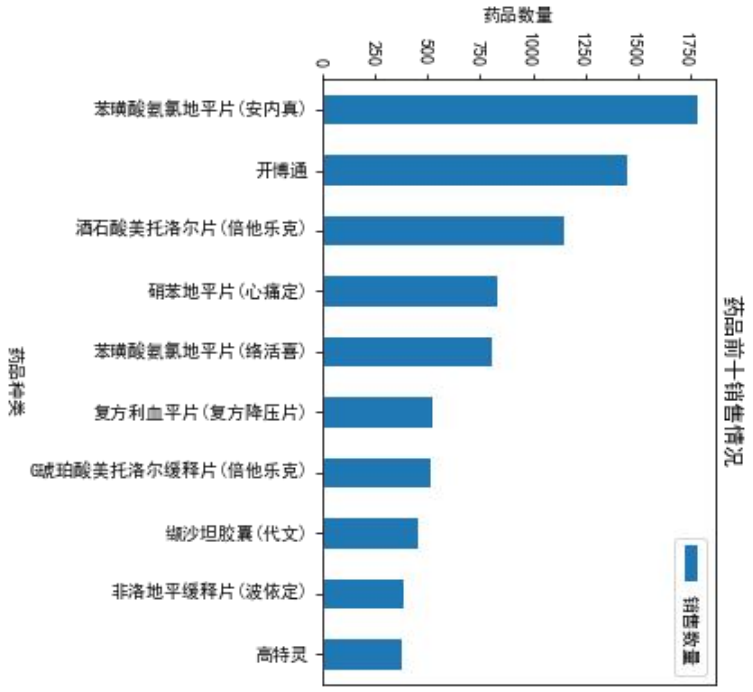
In [73]:
```
saleSum = salesDF['销售数量'].sum()
print('总销售数量:',saleSum)
```

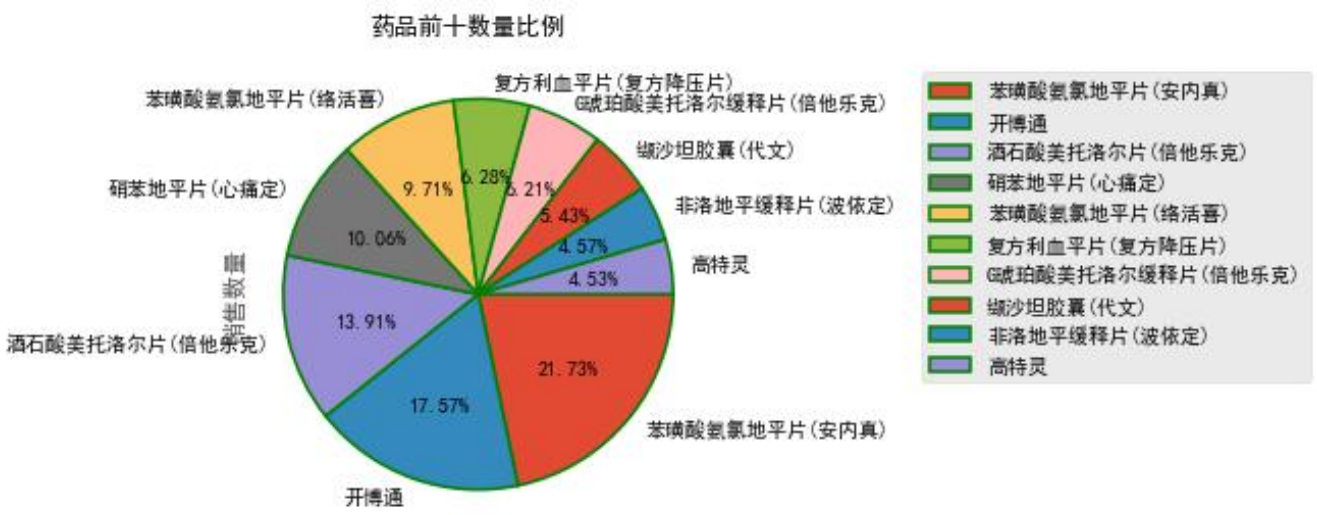
总销售数量: 15656.0

In [76]:
```
top_medicine.shape
```

Out[76]: (10, 1)

In [98]:
```
# 记录前十药品总数量的比例
top_medicine.plot(kind = 'pie',autopct = '%.2f%%', radius = 1.1, counterclock = False,
                  title = '药品前十数量比例', wedgeprops = {'linewidth':1.5, 'edgecolor':'green'},
                  textprops = {'fontsize':10, 'color':'black'},subplots=True)
plt.legend(loc=2, bbox_to_anchor=(1.5,1.0),borderaxespad = 0.)
plt.show()
```





得到销售数量最多的前十种药品信息，这些信息将会有助于加强医院对药房的管理。

分析体重身高数据

导入数据

```
In [ ]: # 分析身高体重数据
import pandas as pd
import numpy as np
classdata=pd.read_csv("D:/Pythondata/data/class.csv")
classdata.head()
```

	Name	Sex	Age	Height	Weight
0	Alfred	M	14	69.0	112.5
1	Alice	F	13	56.5	84.0
2	Barbara	F	13	65.3	98.0
3	Carol	F	14	62.8	102.5
4	Henry	M	14	63.5	102.5

数据描述统计

```
In [ ]: classdata.describe()
```

几何平均数

```
In [8]: from scipy import stats
stats.gmean(classdata['Height'])
```

几何平均数:62.133135310943146

身高和体重的相关散点图

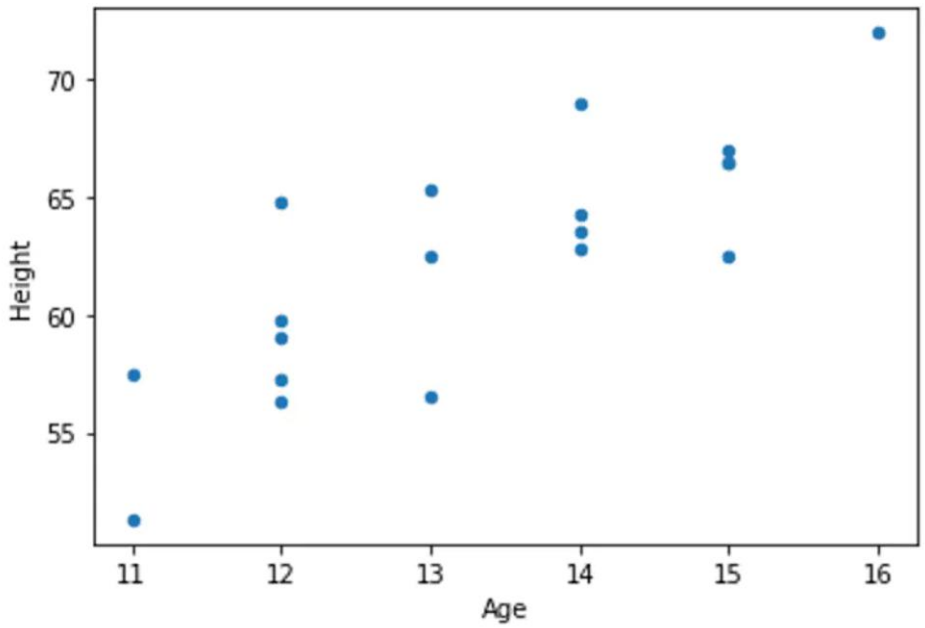
```
import matplotlib.pyplot as plt

plt.scatter(classdata['Height'],classdata['Weight'] )

plt.xlabel("Height")

plt.ylabel("Weight")

plt.show()
```



从散点图的数据分布可以看出，变量 Height 和 Weight 同样呈现很强的相关关系。

六、实验数据及处理结果

（请根据实际的数据来填写）

如步骤五图示。

七、思考讨论题或体会或对改进实验的建议

（请根据实际的数据来填写）

八、参考资料

[1] 朝乐门 著. Python 编程 从数据分析到数据科学.电子工业出版社.2019.1
[2] [美] 阿曼多·凡丹戈（Armando Fandango） 著，韩波 译. Python 数据分析(第 2 版). 人民邮电出版社.2018.6
[3] 嵩天，礼欣，黄天羽 著.Python 语言程序设计基础（第 2 版）. 高等教育出版社.2017.2