**广 州 商 学 院**

**实验报告（第 4 次）**

实验名称 智能停车场运营分析系统 实验时间 2022.05.25

同组同学 王弦 小组分工 独自完成

**一、实验目的**

（1）掌握数据分析方法

（2）掌握项目数据分析流程

（3）掌握使用Pygame搭建应用窗体

**二、实验仪器设备或材料**

1.学生PC机 57 台

2.教师机 1 台

3.教师机安装有极域教师机端、学生PC机安装有极域学生端

4.教学机和学生机需要安装 PyCharm和Anaconda3

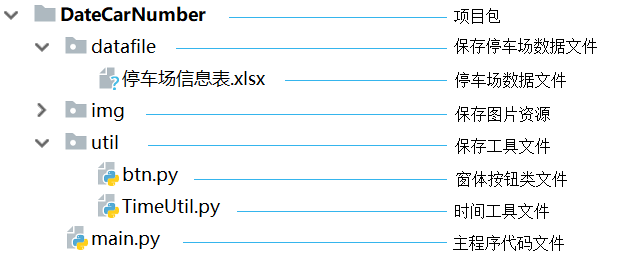
**三、实验原理**

1、教学机与学生机需要安装PyCharm和Anaconda3

2、教学机与学生机需要安装第三方模块pygame、matplotlib、pandas

**四、实验内容与步骤**

**1、项目文件结构**

****

1. **源代码**

**main.py**

import pygame # 导入pygame模块

from util import btn # 导入自定义的按钮

import matplotlib.pyplot as plt # 导入绘制图表的模块

from matplotlib.ticker import FuncFormatter # 绘图模块格式化类

from util.TimeUtil import \* # 导入自定义的时间处理模块

import pandas as pd # 导入pandas模块

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

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

# 窗体大小

size = 340, 484

# 定义颜色

WHITE = (255, 255, 255)

BLUE = (72, 61, 139)

excelFile = r'datafile/停车场信息表.xlsx'

# 读取文件内容

pi\_table = pd.read\_excel(excelFile, sheet\_name='data')

# pygame初始化

pygame.init()

# 设置窗体名称

pygame.display.set\_caption('智能停车场运营分析系统')

# 图标

ic\_launcher = pygame.image.load('img/ic\_launcher.png')

# 设置图

pygame.display.set\_icon(ic\_launcher)

# 设置窗体大小

screen=pygame.display.set\_mode(size)

# 设置背景颜色

screen.fill(WHITE)

# 车位每天利用率

def lyl():

# 获取列表中state(车辆状体)列1为出停车场

tcdf = pi\_table.loc[pi\_table['state'] == 1]

# 循环的开始与结束时间

start = '2018-01-01'

end = '2018-03-31'

# 转换开始与结束时间类型

datestart = datetime.datetime.strptime(start, '%Y-%m-%d')

dateend = datetime.datetime.strptime(end, '%Y-%m-%d')

VALUE = [] # 数据列表

DATE = [] # 日期列表

while datestart <= dateend:

# 判断当前天 出车库的车辆多少

kk = tcdf[tcdf['timeout'].str.contains(datestart.strftime('%Y-%m-%d'))]

DATE.append(datestart.strftime('%Y-%m-%d')) # 将日期添加至列表中

yh =100- kk['rps'].mean() # 计算每天车位使用率

VALUE.append(yh) # 添加至数据列表中

# 按照天循环日期

datestart += datetime.timedelta(days=1)

# 绘制折线图 填充数据

plt.plot(DATE, VALUE)

# yticks格式化方法

def to\_percent(temp, position):

return '%1.0f' % (temp) + '%'

# 格式化yticks，以百分比的方式显示

plt.gca().yaxis.set\_major\_formatter(FuncFormatter(to\_percent))

plt.xticks([]) # 隐藏x轴刻度

plt.xlabel('2018-01-01至2018-03-31') # 显示日期范围

plt.title('车位利用率') # 设置标题

plt.show() # 显示图表

pass

# 周繁忙统计

def fmtj():

# 获取列表中rps（车位剩余）列为0的所有数据

fmdfs = pi\_table.loc[pi\_table['state'] ==1]

# 转换数据为列表

fmdfs=fmdfs.values

# x轴数据

WEEK = ['周一','周二','周三','周四','周五','周六','周日']

WEEK1 = 0 # 星期一

WEEK2 = 0 # 星期二

WEEK3 = 0 # 星期三

WEEK4 = 0 # 星期四

WEEK5 = 0 # 星期五

WEEK6 = 0 # 星期六

WEEK7 = 0 # 星期日

# 循环数据列表

for fmdf in fmdfs:

# 判断数据是星期几

week\_numbeer= get\_week\_numbeer(fmdf[1])

# 星期一

if week\_numbeer==0:

WEEK1 = WEEK1+1

pass

# 星期二

if week\_numbeer==1:

WEEK2 = WEEK2+1

pass

# 星期三

if week\_numbeer==2:

WEEK3 = WEEK3+1

pass

# 星期四

if week\_numbeer==3:

WEEK4 = WEEK4+1

pass

# 星期五

if week\_numbeer==4:

WEEK5 = WEEK5+1

pass

# 星期六

if week\_numbeer==5:

WEEK6 = WEEK6+1

pass

# 星期日

if week\_numbeer==6:

WEEK7 = WEEK7+1

pass

pass

# 数据信息

WEEK\_VAULE=[WEEK1,WEEK2,WEEK3,WEEK4,WEEK5,WEEK6,WEEK7]

plt.title("周繁忙统计") # 设置标题

plt.pie(WEEK\_VAULE, labels=WEEK, autopct='%1.1f%%') # 绘制饼图

plt.axis('equal') # 该行代码使饼图长宽相等

# 显示图例

plt.legend(loc="upper right", fontsize=10, bbox\_to\_anchor=(1.1, 1.05), borderaxespad=0.3)

plt.show() # 显示图表

# 每日接待车辆统计

def cljd():

# 获取列表中state(车辆状体)列1为出停车场

tcdf = pi\_table.loc[pi\_table['state'] == 1]

# 循环的开始与结束时间

start = '2018-01-01'

end = '2018-03-31'

# 转换开始与结束时间类型

datestart = datetime.datetime.strptime(start, '%Y-%m-%d')

dateend = datetime.datetime.strptime(end, '%Y-%m-%d')

VALUE=[] # 数据列表

DATE=[] # 日期列表

# 循环日期

while datestart <= dateend:

# 判断当前天 出车库的车辆多少

kk = tcdf[tcdf['timeout'].str.contains(datestart.strftime('%Y-%m-%d'))]

# 设置x轴数据按照天划分

DATE.append(datestart.strftime('%Y-%m-%d'))

# 日期对应的出车库车辆数

VALUE.append(len(kk))

# 按照天循环日期

datestart += datetime.timedelta(days=1)

#绘制折线图 填充数据

plt.plot(DATE,VALUE)

plt.xticks([])#隐藏x轴刻度

plt.xlabel('2018-01-01至2018-03-31') # 显示日期范围

# 设置标题

plt.title('每日接待车辆统计')

plt.show() # 显示图表

pass

# 停车高峰时间

def tcgf():

# 图表标题

plt.title("停车高峰时间所占比例")

# 设置x轴数据

labels = ['0-3点','3-6点','6-9点','9-12点','12-15点','15-18点',

'18-21点','21-00点']

# 根据时间获取y轴数据 判断包含

kk0 = pi\_table[pi\_table['timein'].str.contains(' 00:')]

kk1 = pi\_table[pi\_table['timein'].str.contains(' 01:')]

kk2 = pi\_table[pi\_table['timein'].str.contains(' 02:')]

kk3 = pi\_table[pi\_table['timein'].str.contains(' 03:')]

kk4 = pi\_table[pi\_table['timein'].str.contains(' 04:')]

kk5 = pi\_table[pi\_table['timein'].str.contains(' 05:')]

kk6 = pi\_table[pi\_table['timein'].str.contains(' 06:')]

kk7 = pi\_table[pi\_table['timein'].str.contains(' 07:')]

kk8 = pi\_table[pi\_table['timein'].str.contains(' 08:')]

kk9 = pi\_table[pi\_table['timein'].str.contains(' 09:')]

kk10 = pi\_table[pi\_table['timein'].str.contains(' 10:')]

kk11 = pi\_table[pi\_table['timein'].str.contains(' 11:')]

kk12 = pi\_table[pi\_table['timein'].str.contains(' 12:')]

kk13 = pi\_table[pi\_table['timein'].str.contains(' 13:')]

kk14 = pi\_table[pi\_table['timein'].str.contains(' 14:')]

kk15 = pi\_table[pi\_table['timein'].str.contains(' 15:')]

kk16 = pi\_table[pi\_table['timein'].str.contains(' 16:')]

kk17 = pi\_table[pi\_table['timein'].str.contains(' 17:')]

kk18 = pi\_table[pi\_table['timein'].str.contains(' 18:')]

kk19 = pi\_table[pi\_table['timein'].str.contains(' 19:')]

kk20 = pi\_table[pi\_table['timein'].str.contains(' 20:')]

kk21 = pi\_table[pi\_table['timein'].str.contains(' 21:')]

kk22 = pi\_table[pi\_table['timein'].str.contains(' 22:')]

kk23 = pi\_table[pi\_table['timein'].str.contains(' 23:')]

# 设置数据信息

x = [(len(kk0)+len(kk1)+len(kk2)),(len(kk3)+len(kk4)+len(kk5)),

(len(kk6)+len(kk7)+len(kk8)),( len(kk9)+len(kk10)+len(kk11)),

(len(kk12)+len(kk13)+len(kk14)),(len(kk15)+len(kk16)+len(kk17)),

(len(kk18)+len(kk19)+len(kk20)),(len(kk21)+len(kk22)+len(kk23))]

# 设置饼图,autopct保留1位小数点

plt.pie(x, labels=labels, autopct='%1.1f%%')

plt.axis('equal') # 该行代码使饼图长宽相等

plt.legend(loc="upper right", fontsize=10, bbox\_to\_anchor=(1.1, 1.05), borderaxespad=0.3) # 显示图例

plt.show() # 显示图表

pass

# 收入分析（月）

def ysrfx():

srdf = pi\_table.loc[pi\_table['state'] == 1]

# 筛选每月数据

kk1 = srdf[srdf['timeout'].str.contains('2018-01')]

kk2 = srdf[srdf['timeout'].str.contains('2018-02')]

kk3 = srdf[srdf['timeout'].str.contains('2018-03')]

# 计算价格和

price1 = kk1['price'].sum()

price2 = kk2['price'].sum()

price3 = kk3['price'].sum()

labels\_x = ['1月', '2月', '3月']

y = [price1, price2, price3]

# 设置柱状图

plt.bar(labels\_x,y)

# 为每一个图形加数值标签

for x, y in enumerate(y):

plt.text(x, y + 300, str(y)+'元', ha='center')

# x,y轴显示文字

plt.xlabel('月份')

plt.ylabel('元')

# 设置标题

plt.title("2018年1-3月收入分析-总收入："+str(price1+ price2+price3)+"元")

plt.show() # 显示图表

# 停车时间分布

def sjfb():

# 图表标题

plt.title("停车时间分布图")

# 设置x轴信息

labels\_x = ['1小时','2小时','3-5小时','6-10小时','11-12小时','12小时以上']

# 获取表中数据判断车辆停车时间

df1 = pi\_table.loc[(pi\_table['price'] == 3)] # 停车1小时

df2 = pi\_table.loc[(pi\_table['price'] == 6)] # 停车2小时

# 停车3-5小时

df3 = pi\_table.loc[(pi\_table['price']>6)&(pi\_table['price']<=15)]

# 停车6-10小时

df4 = pi\_table.loc[(pi\_table['price']>15)&(pi\_table['price']<=30)]

# 停车11-12小时

df5 = pi\_table.loc[(pi\_table['price']>30)&(pi\_table['price']<=36)]

df6 = pi\_table.loc[(pi\_table['price']>36)] # 停车12小时以上

# 停车各时间段停车数量

y=[len(df1),len(df2),len(df3),len(df4),len(df5),len(df6)]

plt.bar(labels\_x,y) # 绘制条形图

# 为每一个图形加数值标签

for x, y in enumerate(y):

plt.text(x, y + 30, str(y)+'台', ha='center')

plt.show() # 显示条形图窗体

# 主线程

Running =True

while Running:

# 创建停车时间分布按钮

btn1 = btn.Button(screen, (90, 50), 140, 60, BLUE, WHITE, "停车时间分布", 20)

# 绘制停车时间分布的按钮

btn1.draw\_button()

# 创建停车高峰时间按钮

btn2 = btn.Button(screen, (90, 130), 140, 60, BLUE, WHITE, "停车高峰时间", 20)

# 绘制停车高峰时间的按钮

btn2.draw\_button()

# 创建周繁忙统计按钮

btn3 = btn.Button(screen, (90, 210), 140, 60, BLUE, WHITE, "周繁忙统计", 20)

# 绘制周繁忙统计的按钮

btn3.draw\_button()

# 创建月收入分析按钮

btn4 = btn.Button(screen, (250, 50), 140, 60, BLUE, WHITE, "月收入分析", 20)

# 绘制月收入分析的按钮

btn4.draw\_button()

# 创建接待车辆统计按钮

btn5 = btn.Button(screen, (250, 130), 140, 60, BLUE, WHITE, "接待车辆统计", 20)

# 绘制接待车辆统计的按钮

btn5.draw\_button()

# 创建车位利用率按钮

btn6 = btn.Button(screen, (250, 210), 140, 60, BLUE, WHITE, "车位利用率", 20)

# 绘制车位利用率的按钮

btn6.draw\_button()

# 更新主窗口

pygame.display.update()

for event in pygame.event.get():

# 关闭页面游戏退出

if event.type == pygame.QUIT:

plt.close()

# 退出

pygame.quit()

exit()

# 判断点击

elif event.type == pygame.MOUSEBUTTONDOWN:

# 鼠标点击位置，判断单击“停车时间分布”按钮

if 20 <= event.pos[0] and event.pos[0] <= 90+70 \

and 20 <= event.pos[1] and event.pos[1] <= 50+30:

sjfb() # 停车时间分布

pass

# 判断单击“月收入分析”按钮

elif 180 <= event.pos[0] and event.pos[0] <= 250+70 \

and 20 <= event.pos[1] and event.pos[1] <= 50+30:

ysrfx() # 收入分析（月）

pass

# 判断单击“停车高峰时间”按钮

elif 20 <= event.pos[0] and event.pos[0] <= 90+70 \

and 100 <= event.pos[1] and event.pos[1] <= 130+30:

tcgf() # 停车高峰时间

pass

# 判断单击“接待车辆统计”按钮

elif 180 <= event.pos[0] and event.pos[0] <= 250+70 \

and 100 <= event.pos[1] and event.pos[1] <= 130 + 30:

cljd() # 每日接待车辆统计

pass

# 判断单击“周繁忙统计”按钮

elif 20 <= event.pos[0] and event.pos[0] <= 90+70 \

and 180 <= event.pos[1] and event.pos[1] <= 210 + 30:

fmtj() # 周繁忙统计

pass

# 判断单击“车位利用率”按钮

elif 180 <= event.pos[0] and event.pos[0] <= 250+70 \

and 180 <= event.pos[1] and event.pos[1] <= 210 + 30:

lyl() # 车位每天利用率

pass

**btn.py**

import pygame

# 自定义按钮

class Button():

# msg为要在按钮中显示的文本

def \_\_init\_\_(self,screen,centerxy,width, height,button\_color,text\_color, msg,size):

"""初始化按钮的属性"""

self.screen = screen

# 按钮宽高

self.width, self.height = width, height

# 设置按钮的rect对象颜色为深蓝

self.button\_color = button\_color

# 设置文本的颜色为白色

self.text\_color = text\_color

# 设置文本为默认字体，字号为20

self.font = pygame.font.SysFont('SimHei', size)

# 设置按钮大小

self.rect = pygame.Rect(0, 0, self.width, self.height)

# 创建按钮的rect对象，并设置按钮中心位置

self.rect.centerx = centerxy[0]

self.rect.centery= centerxy[1]

# 渲染图像

self.deal\_msg(msg)

def deal\_msg(self, msg):

"""将msg渲染为图像，并将其在按钮上居中"""

# render将存储在msg的文本转换为图像

self.msg\_img = self.font.render(msg, True, self.text\_color, self.button\_color)

# 根据文本图像创建一个rect

self.msg\_img\_rect = self.msg\_img.get\_rect()

# 将该rect的center属性设置为按钮的center属性

self.msg\_img\_rect.center = self.rect.center

def draw\_button(self):

# 填充颜色

self.screen.fill(self.button\_color, self.rect)

# 将该图像绘制到屏幕

self.screen.blit(self.msg\_img, self.msg\_img\_rect)

**TimeUtil.py**

# 引入模块

import datetime

# 返回 星期几标记 0代表星期一 1代表星期二...6代表星期天

def get\_week\_numbeer(date):

date = datetime.datetime.strptime(date, "%Y-%m-%d %H:%M:%S")

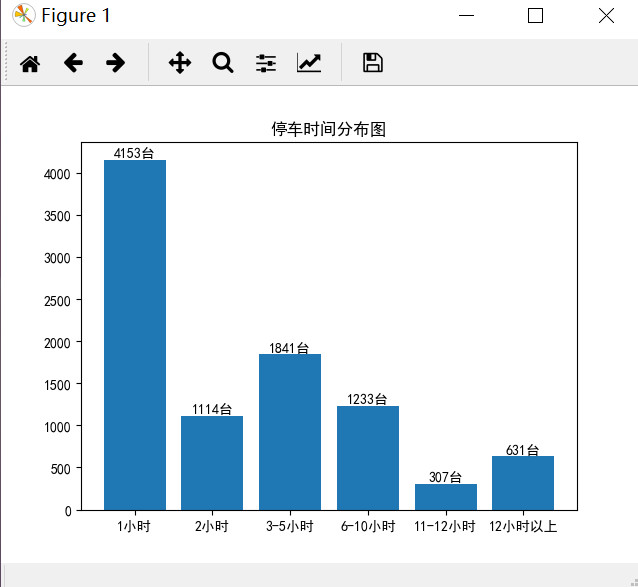
day = date.weekday()

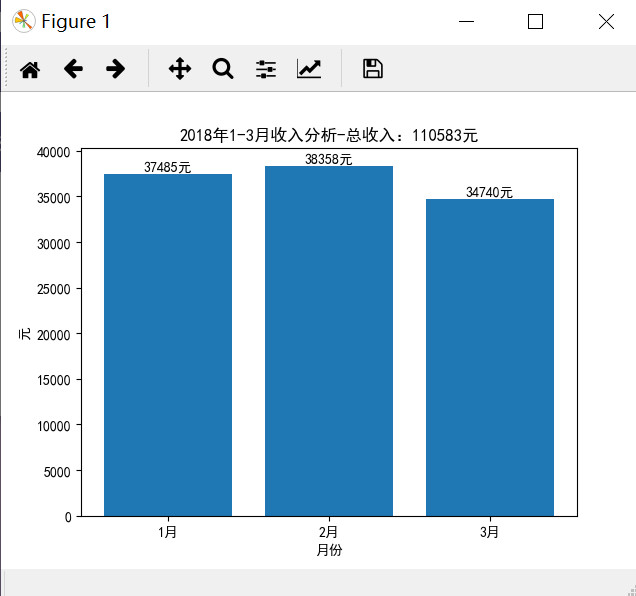
return day

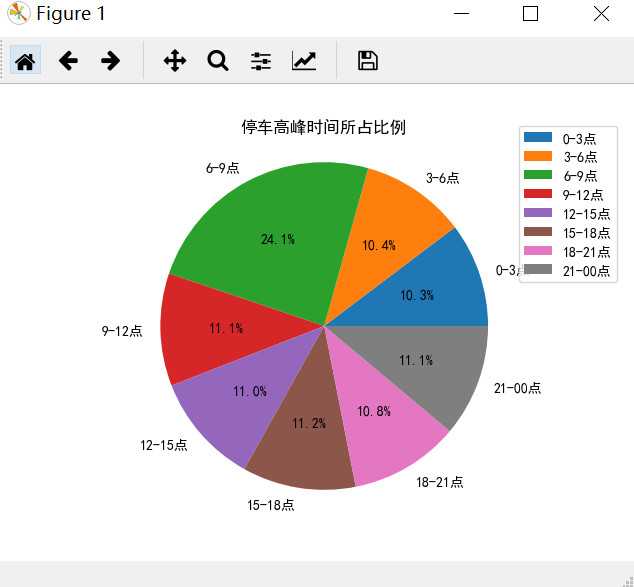
1. **系统预览**

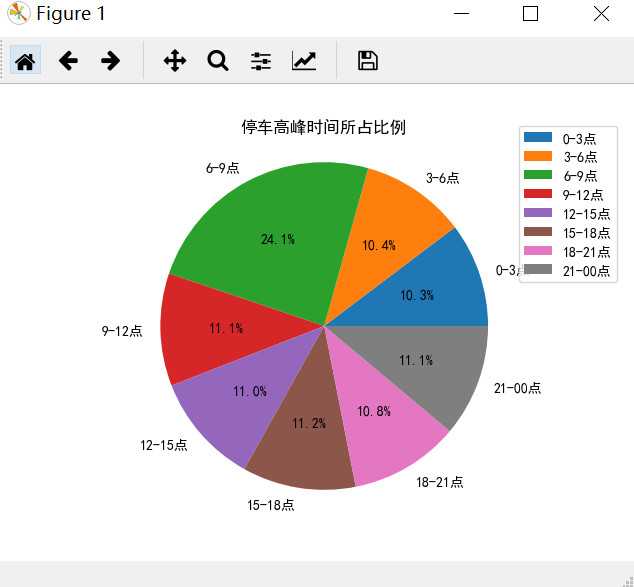
****

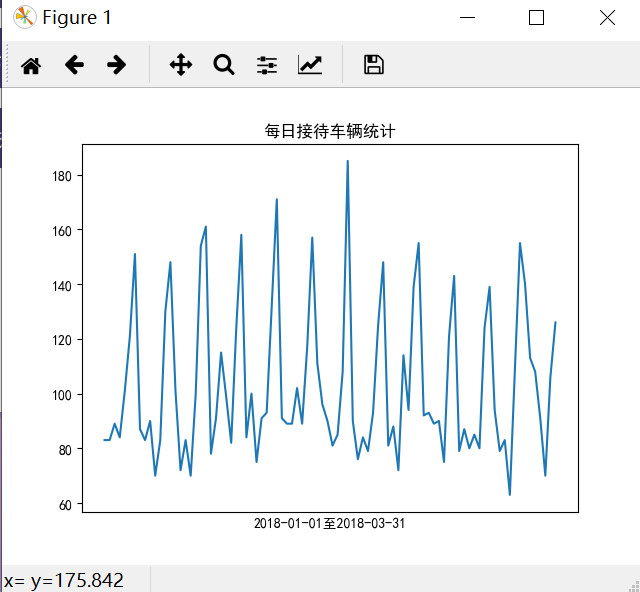
**五、实验结果与分析**

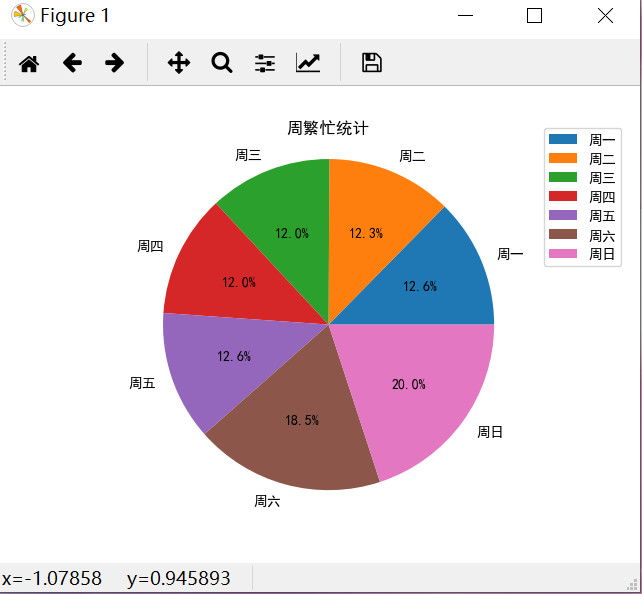
****

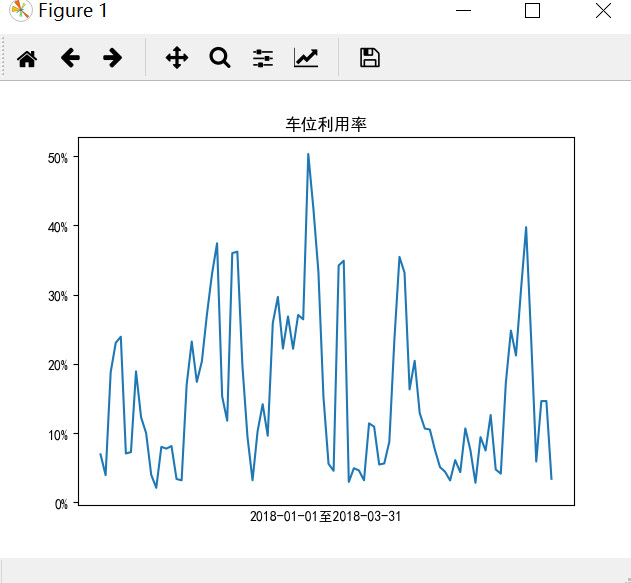
****

****

****

****

****

****

**六、结论与体会**

结论：学习了用Pygame搭建应用窗体。

体会：不够熟练要多练习。

**七、教师评语**