第八章 综合实践项目经典案例

本章将使用前面所学过的知识，基于项目学习学历案的方式，设计编写一些在我们生活、学习过程中的应用，通过这些应用的编写，一是可以综合应用前面章节的知识，二是可以拓展视野，激发兴趣。在这些案例中还需要使用其它第三方库，对这些库的介绍放到了本书的附录部分，请前往阅读。

8.1 项目一 绘制函数图像

8.1.1 项目学习学历案

序号	名称与课时	内容提示
1	项目名称与课时	绘制函数图像。2 课时
2	项目目标	感受 Matplotlib 库的强大绘图功能，掌握 Matplotlib 库中有关坐标系统中相关对象的设置技能，能根据要表现的数据特征合理刻画坐标轴，理解计算机绘制函数图像的原理。
3	评价任务	能用 Numpy 库及函数表达式生成坐标集，并使用 Matplotlib 库绘制相应的函数图像。在此基础上，自己随便假设一个一元函数，能优雅地绘制出这个函数图像吗？
4	学习过程	安装 Numpy 库、Matplotlib 库，阅读并实践附录中的 Numpy 库简介和 Matplotlib 库简介。然后自己尝试编写代码实现，最后参考我们提供的实现代码，比较总结。
5	作业与检测	计算机绘制函数图像的方法就是描点法，在选取点的个数时需要注意什么？除了点的个数可以改变图像的外观外，还有哪些因素也可以改变图像的外观？
6	学后反思	为了使绘制出的函数图像更逼真，可以从哪些方面加以改进？Matplotlib 库不仅能绘制一元函数图像，还能绘制二元函数图像（三维）以及其它坐标系统的图像，为了绘制这样的图像，可能需要从哪些方面着手学习？

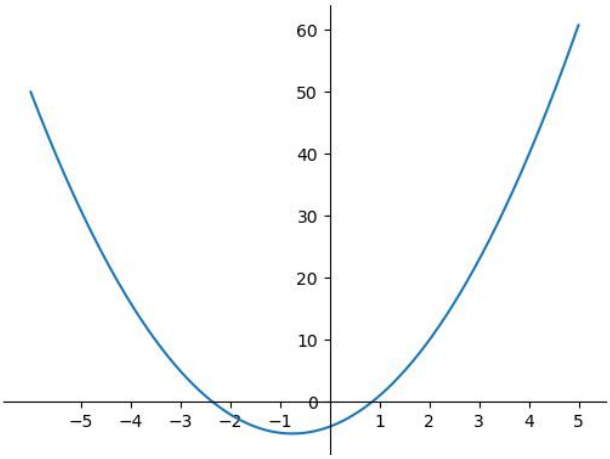
在数学课程中，需要研究函数的单调性、极值、奇偶性等性质，若从 $y=f(x)$ 出发进行研究，比较抽象。学习了计算机编程技术后，我们可以先编写画函数图像的程序，然后一边观察图像，一边从数学的角度进行验证，这样就容易得多。在 Python 中画图，若使用 numpy 和 matplotlib 第三方库将非常方便。其原理是描点法，利用计算机具有高速运算的特点，我们使两个点之间的距离足够近，就能很逼真地模拟函数图像。

8.1.2 案例解析

在 Python 中，matplotlib 库具有强大的数据表现能力，能轻松绘制绝大多数图表。绘制图表的流程可分为这几个步骤：根据函数表达式生成 x,y 坐标、设置坐标轴属性、画函数图像，显示图像。其中最关键的步骤是根据函数表达式生成 x,y 坐标。下面给出的是画函数 $y=2x^2+3x-4$ 图像的程序代码：

```
import numpy as np
import matplotlib.pyplot as plt
#根据函数表达式构造 x,y 坐标
```

```
x=np.arange(-6,5,0.01)
y=[2*i**2+3*i-4 for i in x]
#使 x，y 轴的 0 点重合
x=plt.gca()
ax.spines['right'].set_color('none')
ax.spines['top'].set_color('none')
ax.spines['bottom'].set_position(('data',0))
ax.spines['left'].set_position(('data',0))
plt.xticks([-5,-4,-3,-2,-1,1,2,3,4,5])          #设置 x，y 轴的刻度标注
plt.plot(x,y)                                    #画函数图像
plt.show()                                       #显示图像
```



8.2 项目二 爬取汽车票

8.2.1 项目学习学历案

序号	名称与课时	内容提示
1	项目名称与课时	开发一个蜘蛛程序，爬取汽车票。3 课时
2	项目目标	通过项目的实践，能从没有设置反爬功能的网站中下载 HTML 源码，初步了解 HTML 文档结构，掌握 bs4 从 HTML 文档中提取需要的信息的方法，体会用面向对象的思想编写爬虫程序的框架和主要过程。
3	评价任务	能下载携程 HTML 源代码，并能从 HTML 源码中正确分析出关于车票的 HTML 代码；能用面向对象编程的思想编写程序；能以清晰易懂的格式输出爬取

		结果。
4	学习过程	安装 requests 库、Beautiful Soup 库，阅读并实践附录中的 requests 库简介和 Beautiful Soup 库简介。然后按照自顶向下设计原则，搭建主框架，在需要实现具体功能的地方使用占位语句 pass 代替，再逐一编写类方法，最后参考我们提供的实现代码，比较总结。
5	作业与检测	写出设计爬虫程序的步骤，面向对象编程比面向过程编程有哪些优点？
6	学后反思	编写爬虫程序的难点是什么？在这个项目中我们针对携程 HTML 的结构特点爬取了车票信息，但不同网站的 HTML 结构肯定是不一致的，能否编写一个比较通用的爬虫呢？

“百度一下”也许是我们用得最多的操作。那么百度上那些资源是怎么来的呢？实际上，所有的搜索引擎都有一个称为“网络蜘蛛”的工具，这个工具会定时或不定时地从其它网站上读取信息，并对读取到的信息经过特定的分类算法保存到搜索区供用户搜索。使用 Python，也能编写“网络蜘蛛”爬取其它网站信息。在这个案例中我们编写一个从携程网爬取汽车票的“小蜘蛛”。（携程网查询汽车票的网址：http://bus.ctrip.com/busListn.html）

程序运行后，用户输入出发地名称、目的地名称、日期，程序显示爬取结果并将结果保存到文本文件（ctrip_car_info.txt）中。

输入样例：

请输入出发地、目的地、日期(日期格式:2018-10-10，用空格隔开)：德江 贵阳 2018-10-11

输出样例：

发车时间	起点站/终点站	车型	票价
08:00	德江客运站/贵阳客运东站	中型座席高一级	¥100.00
09:30	德江客运站/贵阳客运东站	中型座席高一级	¥100.00
……			

8.2.2 解析

8.2.2.1 爬虫程序编写思路

网络爬虫的基本工作流程是：获取一个 URL，读取 URL 的源码，从源码中解析出新的 URL，将这些新的 URL 放到 URL 的队列中，逐一读取 URL 的源码，再次解析出新的 URL 放入 URL 队列中，如此反复，直到 URL 队列为空，爬取过程结束。

当成功从服务器上载入 HTML 代码后，往往需要分析这些代码，分析所需要的数据信息是在哪个节点下面，然后再使用正则表达式或第三方模块去提取信息。尽管使用正则表达式具有较高的效率，但正则表达式的书写是一件很复杂的工作，所以往往是通过第三方模块来解决，这也是戏称 Python 为“胶水语言”的原因之一。在这个案例中，我们使用 BeautifulSoup 库来提取信息。（先安装 requests 和 beautifulsoup4：pip install requests pip，install beautifulsoup4）

8.2.2.2 搭建主框架

```
# _*_ coding:utf-8 _*_
import sys,os,re,requests
from bs4 import BeautifulSoup

class Ticket(object):
    def __init__(self,time,station,cartype,price):
        self.time = str(time)                #发车时间
        self.station = str(station)          #起点站/终点站
```

```
            self.cartype = str(cartype)      #车型
            self.price = str(price)          #票价

    class SpiderTicket(object):
        def __init__(self,data):
            self.url = http://bus.ctrip.com/busListn.html     #欲爬取的网址
            self.data = data                                  #参数
            self.soup = None                                  #解析后的源码
            self.state = 0                                    #程序运行状态
            self.error_info = None                            #错误信息
            self.result = []                                  #爬取的结果

        def down_data(self):
            """获取资源"""
            pass
        def parse_data(self):
            """解析 HTML 代码"""
            pass
        def disp_data(self):
            """显示数据信息"""
            pass

        def save_data(self):
             """保存信息到文本文件中"""
            pass

    def main():
        while True:
         try:
            source,to,date = input('请输入出发地、目的地、日期(日期格式:2018-10-10，用空格隔开)：
').strip().split()
            if not re.match(r"^\d{4}-\d{2}-\d{2}$",date):
               raise ValueError
            break;
         except:
            print("输入有误！请重新输入")
        st = SpiderTicket({'from':source,'to':to ,'date':date})
        st.down_data()
        st.parse_data()
        st.disp_data()
        st.save_data()
    if __name__ == "__main__":
        main()
```

8.2.2.3 完善获取资源方法

修改 SpiderTicket 类的 down_data( )方法。

```
        def down_data(self):
```

```
        """获取资源"""
        try:
            response = requests.get(self.url,self.data)
            response.raise_for_status()              #如果发生错误，引发异常
            self.soup = BeautifulSoup(response.text,features="html.parser") #构造 bs 对象
        except Exception  as e:
            self.state = 1
            self.error_info = "请求资源时发生了错误，程序终止运行。"
```

8.2.2.4 完善解析 HTML 代码方法

修改 SpiderTicket 类的 parse_data ( )方法。

知识拓展：使用浏览器的“开发者工具”可以快速找到页面元素对应的代码。

这一步是编写爬虫程序的核心步骤，需要分析被爬取网页的代码结构，设计、优化解析逻辑，编写代码。下面是从 HTML 源码中摘抄的关于车票信息的部分代码。

```
01 <table class="tb_railway_list nolayout" id="table" width="100%">
02    <tbody>
03     <tr>
04    <th width="17%"><a href="javascript:void(0);" id="" class="or_up current f_sort_list"
sort_name="from_time" onclick="return sortObj(this);">发/到时间<b class="icon_arrow_up"></b></a></th>
05    <th width="20%">发/到站</th>
06    <th width="18%">车型/耗时</th>
07    <th width="17%" class="price"><a href="javascript:void(0);" class="px_up f_sort_list"
sort_name="full_price" onclick="return sortObj(this);">票价<b class="icon_arrow_up"></b></a></th>
08    <th width="28%"></th>
09  </tr>
10  <!--推荐火车票 start-->
11  <tr class="recom" id="tuijian_train"></tr>
12  <!--推荐火车票 end-->
13  <!--推荐玩乐 start-->
14  <tr class="recom" id="tuijian_wanle"></tr>
15  <!--推荐玩乐 end-->
16 <tr>
17   <td class="time_box"> <span class="railway_time">09:30</span><br/></td>
     18   <td><span class="icon_start"></span>德江客运站<br/><span class="icon_end"></span>贵阳
  客运东站</td>
19   <td>中型座席高一级<br/></td>
20   <td class="price" style="position: relative">
21    <div class="railway_seat">
22      <div class="price_r">
23        <dfn>¥</dfn><span class="base_price base_full_price">100.00</span>
24      </div>
25    </div>
26   </td>
27  <td class="text_right">
        <input type="button" class="btn_book goBook"
data-params="{mustLogin:false,symbol:'rUaI6k6Z1hNESbmiXvmfp83E',from:'德江',fromStation:'德江客运站
```

',to:'贵阳',toStation:'贵阳
',busNo:'6038037',date:'2018-10-12',time:'16:00',hashkey:'3272fe85dc4dd875f851e4d039ba7a42',return_type:'',discount_count:'',discount_price:'',active_stop_flag:'',vendor_activity_id:'',active_start_dttm:'',active_end_dttm:'',vendor_activity_flag:'0',vendor:'',data_source:''}" value="预订">

28 </td>

29 </tr>

#省略类似 16—29 行的若干行代码

30 </tbody>

31 </table>

可以看出，需要的信息全部是在一个表格（table）中，16-29 行是一个班次的信息，如果有多个班次，那么类似 16-29 行就有多少个，因此先使用 BeautifulSoup 的 select( )方法筛选出所有行（tr)，【第一行为表头，第二行、第三行为空】，提取出第四行及以后的信息。代码如下：

```
tag_group =self.soup.select("table > tr")      #筛选第一个标签为 table 的所有子节点 tr
for index,subitem in enumerate(tag_group):
    if index <3:                               #忽略前面 3 行
        continue
```

提取每行的前 4 列内容，代码为：

```
    td_time = subitem.td
    td_station = td_time.find_next_sibling()
    td_cartype = td_station.find_next_sibling()
    td_price = td_cartype.find_next_sibling()

    time = td_time.span.string
    station ="/".join(td_station.stripped_strings)
    cartype = ''.join(td_cartype.stripped_strings)
    price = td_price.span.string  #价格
```

最后得出 parse_data( )方法的完整代码为：

```
def parse_data(self):
    """解析 HTML 代码"""
    if self.state:                                  #若请求资源时发生了错误
        return
    tag_group =self.soup.select("table > tr")    #筛选第一个标签为 table 的所有子节点 tr
    for index,subitem in enumerate(tag_group):
        if index <3:                                        #忽略前面 3 行
            continue
        td_time = subitem.td                                #时间标签
        td_station = td_time.find_next_sibling()            #起点站/终点站标签
        td_cartype = td_station.find_next_sibling()         #车型标签
        td_price = td_cartype.find_next_sibling()           #价格标签

    time = td_time.span.string
        station ="/".join(td_station.stripped_strings)
        cartype = ''.join(td_cartype.stripped_strings)
        price = td_price.span.string  #价格
        self.result.append(Ticket(time,station,cartype,'¥'+price))
    if len(self.result) == 0:
        self.state = 1
```

```
            self.error_info = "未能采集到任何数据。"
```

8.2.2.5 完善解析显示数据、保存数据的方法

修改 SpiderTicket 类的 disp_data（）、save_data（）方法。

```
    def disp_data(self):
        """显示数据信息"""
        if self.state:
            print(self.error_info)
        else:
            print("时间".center(6,' '),"起点站/终点站".center(30,' '),"车型".center(20,' '),"票价
".center(6,' '))
        for item in self.result:
            print(item.time.center(6,' '),item.station.center(30,' '),item.cartype.center(20,'
'),item.price.center(6,' '))


    def save_data(self):
            """保存信息到文本文件中"""
            if self.state:return
            with open(r'ctrip _car_info.txt','w',encoding='utf-8') as file:
                  file.write("发车时间\t 起点站/终点站\t\t\t 车型\t\t 票价\n")
                  for item in self.result:
                      file.write(item.time+"\t"+item.station+"\t"+item.cartype+"\t"+item.price+'\n
                      ')
```

这是一个非常简单的爬虫案例，真正的爬虫程序要考虑的问题还有很多，同时用 Python 做爬虫工具还有很多更好的框架，如 scrapy、crawley、PySpider 等。

8.3 项目三 用机器学习预测泰坦尼克号邮轮乘客的生死

8.3.1 项目学习学历案

序号	名称与课时	内容提示
1	项目名称与课时	用机器学习预测泰坦尼克号邮轮乘客的生死。学习这个项目，你可以了解目前机器学习的本质，站在巨人的肩膀上，写一个真正的人工智能（机器学习）程序，从而消除你对人工智能技术的恐惧感。你还可能举一反三，写几个其他的人工智能应用。3 课时
2	项目目标	期望你能通过完成这个真正的人工智能程序，了解人工智能之机器学习的基本原理，掌握开发人工智能应用项目的一般过程。提升自己的计算思维等核心素养。
3	评价任务	能用 sklearn 的“决策树”模型（无需理解内部算法，直接调用）与训练数据生成一个统计模型（预测模型）；能用测试数据检验你的预测模型。你能仿照这种方法与步骤，设计一个其他机器学习应用吗？
4	学习过程	这个项目的代码需要在 notebook 服务器环境运行。安装 sklearn 库、notebook 库、seaborn 库、numpy 库、pandas 库、warnings 库。如果你的系统已经安装 anaconda3，这些库已经安装，使用 jupyter notebook 命令启动 notebook 服务器。按照顺序阅读，并按说明用 Python 的命令对数据做各种操作，其中的训练数据文件 train.csv 与检验数据文件 test.csv 均在统一的资源包里，无需再去亚马逊云服务器下载。
5	作业与检测	目前的人工智能是机器会思考吗？它的实质是什么？用测试数据检验你的预测模型，改变一些测试数据看看准确度是达到多少？能用这种方法写一个其他的人工智能应用吗？
6	学后反思	你能总结完成这个项目的基本步骤吗？你认为这个项目最困难的地方是什么？应该特别注意的事项是什么？

8.3.2 项目简介

这个项目是用机器学习的方法判断泰坦尼克号上哪些乘客会存活，哪些乘客会遇难。我们想通过这个项目学习，消除你对人工智能技术的恐惧感，为此，我们今天要一起写一个真正的人工智能程序。

这里不是用假设性数据闹着玩，我们要用真实的数据和真实的算法，做一个真实的人工智能项目，预测泰坦尼克号邮轮上每一位乘客的生死。也就是说，我们使用的是泰坦尼克号邮轮上每一位乘客的真实数据，我们根据其中部分乘客数据，用机器学习的方法生成一个预测模型，然后就这个学习得到的模型，预测另一部分旅客中每个人是否活了下来，并与真实情况进行对比，看看准确度。

8.3.3 理论准备

像很多电影里真人一样的人工智能，叫做“广义人工智能”，这种技术在可以预见的未来都不存在。现在大家用的都是“狭义人工智能”，而狭义人工智能的本质就是人们常说的“机器学习”。

所谓“机器学习”，并不是说机器有思想，它学会了一项技能。机器学习就是用一组数据建立一个统计模型，这个统计模型能对新的数据做出预言。输入数据越多越精确，模型能做的预言就越准确，就好像是它在不断地“学习”一样。数学家管这叫“统计模型”，计算机科学家给起了个更值钱的名字叫“机器学习（Machine Learning，缩写为“ML”）”，媒体有时候管这叫“大数据”，而其实这就是现在科技大佬们在发布会上口口声声说的“人工智能”。

8.3.4 数据分析

（该项目的所有数据、实现脚本代码下载地址为：http://i.tryz.net/html/2018/python/pythonjc.rar）

我们获得的数据集中，共有 891 人，但是其中只有 714 人的年龄记录，没有年龄记录的我们已经用年龄的平均值代替。我们把标注数据集（891 人）按照随机划分 80%的记录作为训练集，20%作为验证集。其中 train.csv 为训练集（730 人），用来训练一个统计模型；test.csv 为验证集（161 人），用来检验这个模型的有效性。以下所有操作都是针对训练集进行的。

训练集中的每一行数据代表 1 位乘客的信息，每一行都有 12 项数据，分别对应每位乘客的以下属性：

乘客编号，乘客是否存活，舱位(头等舱、二等舱、三等舱)，乘客姓名，乘客的性别，年龄， 在泰坦尼克号上有没有兄弟姐妹或者配偶，在泰坦尼克号上有没有父母或者子女， 乘客的船票号码， 买的船票价格，在船上住的房间编号， 在英国哪个口岸上的船。

使用 Python，我们只要用一些简单的命令就可以对数据做各种操作。比如我们想知道各个舱位都有多少乘客，可以用如下代码实现：

```
01 import pandas as pd                          #导入 pandas 库
02 import seaborn as sns                        #导入 seaborn 库
03 train_data = pd.read_csv('D:/titanicdata/train.csv')  #装载训练集，路径根据实际情况修改
04 sns.set_style('whitegrid')                   #设置 sns 的显示风格为“白色网格”
05 train_data.head()                            #查询数据的前 5 行
```

输出结果：

```
In [11]: import pandas as pd
         import seaborn as sns
         train_data = pd.read_csv('d:/titanicdata/train.csv')
         sns.set_style('whitegrid')
         train_data.head()
```

Out[11]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

```
train_data["Pclass"].value_counts()    #对舱位进行分类统计
```

输出结果：

```
3    398            #三等舱人数
1    178            #头等舱人数
2    154            #二等舱人数
```

```
train_data["Survived"].value_counts()  #对乘客是否存活进行分类统计
```

输出结果：

```
0    445          #遇难人数
1    285          #存活人数
```

如果用频率表示概率的话，那么对于任意一位乘客，遇难的概率为 60.9%。下面我们分别统计出男性和女性的遇难情况：

```
train_data["Survived"][train_data["Sex"] == 'male'].value_counts()     #男性
0   378        #378 人遇难
1    90        #90 人存活
train_data["Survived"][train_data["Sex"] == 'female'].value_counts()    #女性
1    195       #195 人存活
0    67        #67 人遇难
```

同理，对于任意一位男乘客，遇难的概率为 81%，对于任意一位女乘客，遇难的概率 26%。

8.3.5 让机器学习算法生成模型

要想进一步提高准确度，我们就要让机器学习算法了。我们使用 Python 的机器学习库 Sklearn 来实现。由于涉及很多高深的知识，你现在不需要知道它的细节，直接调用就可以。

导入模块的语句： from sklearn import tree,preprocessing

从上面的结果中可以看出，当用性别去评估（预测）时结果截然不同。那么，还有哪些指标可能对结果产生影响呢？这里我们仅考虑舱位、性别、年龄、船票费用这 4 个指标。下面是创建预测模型的代码：

```
01 from sklearn import tree,preprocessing
02 target = train_data["Survived"].values      #获取所有乘客是否遇难的数据
03 encoded_sex = preprocessing.LabelEncoder() # 生成预处理工具
04 train_data.Sex = encoded_sex.fit_transform(train_data.Sex)
# 对标签做预处理，例如将 男 (male) 处理为 0， 将 女 (female) 处理为 1。
# 因为后面的模型处理只能处理数字，所以需要对数据做预处理
05 features_one = train_data[["Pclass","Sex","Age","Fare"]].values
06 my_tree_one = tree.DecisionTreeClassifier().fit(features_one,target)  #训练模型
```

说明：

（1）如果想了解这些指标对结果的影响的相对重要性，可以查看模型的 feature_importances_属性值。如：在这个模型中，执行 print(my_tree_one.feature_importances_)

将输出：[0.11807187 0.30965723 0.23300241 0.33926848]，分别表示 Pclass 占 11.8%，Sex 占 31%，Age 占 23.3%，Fare 占 33.9%。可看出 sex、Fare 对结果的影响更显著一些。

（2）如果想了解模型预测的准确度，可以查看模型的 score 属性值。如：在这个模型中，执行 print(my_tree_one.score)

将输出：0.9794520547945206，表示准确率为 98%。

8.3.6 用测试数据检验模型

对模型的真正考验是使用验证集数据。验证集数据在训练集中不存在，没有参与训练，是新信息。我们要把这个 test.csv 数据拿过来，用刚才训练好的模型直接预测这些我们完全没接触过的乘客的存活情况！结果是，我们这个模型用于测试数据的准确度仍然高达 80%！验证代码如下：

```
01 test_data = pd.read_csv('d:/titanicdata/test.csv') #装载验证集，路径根据实际情况修改
02 test_data.Sex = encoded_sex.fit_transform(test_data.Sex)
03 test_features = test_data[["Pclass","Sex","Age","Fare"]].values
04 test_target = test_data["Survived"].values
05 print(my_tree_one.score(test_features,test_target)) # 查看预测检验数据的准确度
```

输出：0.8074534161490683

我们使用这个模型来预测三等舱、女、20 岁、船票费用为 3455 的乘客是否会遇难：

```
01 import numpy as np
02 print(my_tree_one.predict(np.array([[3,1,20,3455]])))
```

输出：0，表示遇难。

8.3.7 算法总结

现在你已经完成了一次人工智能编程。回顾一下，整个步骤是这样的：

1. 把所有数据分成两组，一组用于训练，一组用于检验；
2. 数据都是数组，其中包括你想要预测的目标信息（是否存活），以及可能影响这个目标的各种信息；
3. 选择几个你认为最有可能影响目标的信息（舱位、性别、年龄、票价）；
4. 选择一个机器学习算法（“决策树”）；
5. 把目标和可能影响目标的几个信息作为数组变量输入算法，训练得到一个预测模型；
6. 把预测模型应用于检验数据，看看这个模型的准确度。

我们这里一点基础都没有，怎么能一下子就写一个机器学习程序呢？答案当然是因为你站在了巨人的肩膀上。现在机器学习的算法已经非常成熟，连专业人士都不需要自己从头到尾写一个算法，网上都有现成的。常用的机器学习算法大概有十几个，都做成了 Python 的库，我们可以直接调用。

8.3.8 思维拓展

学会了这个方法，使用现成的工具，只要有足够好的数据，你立即就可以搞几个人工智能应用。比如一个信用卡公司有十万个用户的详细数据，包括年龄、收入、以往的购买记录、信用得分、还款记录等等，那你就可以预测其中每一个人下个月按时还款的可能性。

不过我们今天的主要目的，还是体会现在所谓的“人工智能”到底是怎么回事儿。它仅仅是一个统计模型而已。从这个实操例子中我们可以得出两个洞见：

第一个洞见是，每个模型都会带来歧视。

泰坦尼克模型中对生死影响最重要的变量是船票价格。船票越贵，你存活的可能性就越高。那假设你是一个卖保险的，现在既然你知道买了高价票的人存活率高，将来他们不太可能找你理赔，那你为了多卖几分保险，是不是可以少收一点他们的保险费？

这就是区别定价，这就是价格歧视。你并不是歧视穷人也不是更爱富人，只不过为了多赚点钱，你就会多收穷人的保险费，少收富人的保险费，你完全是理性的。可是，请问这合理吗？这道德吗？

第二个洞见是，人工智能真的不智能。

说白了，我们今天做的就是四项指标去预测一项指标而已。你完全可以想到还有很多别的因素对在泰坦尼克号上存活很重要，但是我们根本没考虑。

比如说，当时放救生艇的时候，船长的命令是“让女人和孩子上救生艇”，但是两侧放救生艇的人对命令的理

解不一样。一个人的理解是优先让妇女儿童上，如果周围的妇女儿童都上了救生艇，还有空位，那就让男的也上。另外一个人的理解则是只让妇女儿童上救生艇，妇女儿童上完哪怕还有空位，也不让男性上，就直接把救生艇放下去。

那如此说来，一个男人能不能存活，跟他当时距离哪个救生艇的距离很有关系！但是我们根本就没有这项数据。

还有，在船即将沉没的一刹那还站在船头的人，如果你选择往远处猛跳，你就有可能存活下来；如果你不跳，沉船造成的旋涡就会把你拖下水，你就很可能遇难。所以最后一刻的跳法也决定了生死，但是这个跳法也没有包括在模型里面。

模型对泰坦尼克上发生了什么一无所知，它根本不理解自己在干什么，本质上它的一切预测都是猜的——所谓机器学习，只不过是增加它猜对的概率而已。

但是，我们居然就做到了这么高的准确率。人工智能界对此有个专门的形容词，叫“unreasonably effective”，不合理地准确。如此粗糙的模型，它居然就能做到这么准确！

你可能对此非常感慨！但是当你的手机准确识别了你的语音的时候，当 AlphaGo 赢了柯洁的时候，你也有过这样的感慨。它们的算法更复杂，但是本质原理都是一样的。

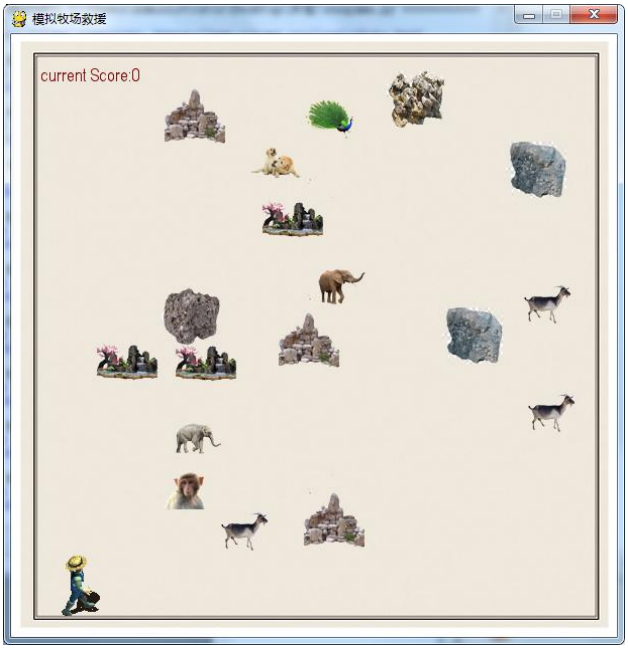
破除迷信战胜恐惧最好的办法就是，亲自做一遍，现在你至少知道你可以做到。现有的人工智能就是用统计方法增加猜测的准确度。人工智能就是机器学习。机器学习就是统计模型。

8.4 项目四 模拟牧场救援游戏

8.4.1 项目学习学历案

序号	名称与课时	内容
1	项目名称与课时	模拟牧场救援游戏。5 课时
2	项目目标	通过项目的实践，了解游戏的本质，不沉迷于游戏；理解使用 pygame 开发游戏的流程。熟悉面向对象编写程序的流程。
3	评价任务	能说出游戏中角色运动的原理；能用面向对象编程的思想编写程序；
4	学习过程	安装 pygame 库。阅读并实践附录中的 pygame 库简介，再根据项目实现的顺序一步一步地进行仿写。每完成一个步骤思考为什么要这么做？直到整个项目完成。（强烈建议你一定要亲自动手敲代码）
5	作业与检测	使用一张大小合适的图片作为角色，使用方向键控制图片的运动。
6	学后反思	游戏的本质是什么？ 使用 pygame 库编写游戏时怎样才能降低对 CPU 资源的消耗？

很多学生都喜欢玩游戏甚至沉迷于游戏，其实，游戏就是设计者在一些载体（如图片、声音等）上制定的一套规则，让玩家在遵循这套规则的条件下参与行动，从起点到达终点的过程而矣。我们用这个案例来模拟一个游戏的开发过程，以期同学们从理性上认识游戏的本质。游戏完成后界面如下图所示：



（该项目的所有数据、实现代码下载地址为：http://i.tryz.net/html/2018/python/pythonjc.rar）

角色：被救援对象（一些动物图片）、障碍物（石头等图片）、玩家（人的图片）

游戏规则：

（1）游戏初始时：随机放置被救援对象 8~10 个，障碍物 10 个，玩家 1 个，位置相互不重合，每隔 5 秒自动变换 1 个障碍物的位置；玩家起始位置为左下角。

（2）玩家操作：用键盘控制玩家角色移动的方向，当一直按住方向键不放时，玩家角色移动的速度会越来越快；当玩家角色接触到障碍物时减 10 分，并且往反方向后退一段距离（模拟反弹效果），速度降为初始速度；接触到被救援对象时，加 30 分，被救援对象消失。

（3）当所有被救援对象都救援成功时，程序结束，救援成功。一旦为负分，游戏结束，救援失败。

（4）程序结束后，显示总分和救援时间。

8.4.2 案例解析

pygame 库为 Python 开发小游戏提供了很多模块，这些模块完成了与底层开发相关的内容，使游戏开发人员不必关心怎样与硬件打交道、怎样管理事件等工作，开发人员的工作重点是游戏逻辑功能的设计。在这个案例中，我们使用 pygame 库来实现。

从技术角度来看，游戏就是具有交互功能的动画片。如果在极短时间内播放形状相似的多张图片就会感到图片在“动”，这是动画片的基本原理。因此，设计游戏的基本思想就是让程序不停地接收、处理键盘（鼠标）事件并不停地更新画面内容。

8.4.2.1 搭建主框架

分别建立 Player 类、Obstacle 类、Sos 类分别表示游戏中的玩家、障碍物、待救援对象这些角色。用 SoSGame 类表示游戏。由于这些角色可能都需要“动”的效果，所以创建一个继承于 pygame.sprite.Sprite 的 Shape 类作为基类。将这些类的定义代码保存为 RoleLibrary.py 模块，内容如下：

RoleLibrary.py 模块内容

```
import  random, pygame
from pygame.locals import *

class Shape(pygame.sprite.Sprite):
       def __init__(self,x,y,width,height,image,columns,target):
             """初始化角色"""
             pass
       def update(self,current_time, rate=500):
             """更新精灵的图像帧"""
             pass

class Sos(Shape):
      def __init__(self,target):
          pass

class Obstacle(Shape):
    def __init__(self,screen):
        pass

class Player(Shape):
      def __init__(self,screen):
         pass
```

在游戏主程序中需要完成各种角色创建、消息检测、逻辑判断等工作。构建如下框架：

```
# _*_ coding:utf-8 _*_
import sys, time, random, pygame
from pygame.locals import *
from RoleLibrary import *

class SoSGame():
      def __init__(self,screen):
            """初始化游戏"""
            self.initialization()
```

```
    def initialization(self):
        """创建角色"""
        pass
    def collision(self,obj):
        """碰撞检测"""
        pass
    def transobpos(self):
        """更换障碍物位置"""
        pass
    def set_gamestate(self,state):
        """设置游戏状态"""
        pass
    def get_gamestate(self):
        """获取游戏状态"""
        pass
    def meetsos(self,obj):
        """碰到救援对象"""
        pass
    def meetobst(self,obj):
        """碰到障碍物"""
        pass
    def update(self):
        """刷新游戏界面"""
        pass
    def displayscore(self):
        """显示当前分数"""
        pass
    def gameover(self,state):
        """游戏结束"""
        pass

def main():
    """初始化 pygame、逻辑控制"""
    pygame.init()
    screen = pygame.display.set_mode((600,600),0,32)
    pygame.display.set_caption('模拟牧场救援')
    timer = pygame.time.Clock()
    sosgame = SoSGame(screen)
    while True:
        timer.tick(30)
        for event in pygame.event.get():
            if event.type == QUIT:
                pygame.quit()
                sys.exit()
        keys = pygame.key.get_pressed()     #轮询键盘事件
        if keys[K_UP] or keys[K_w]:
            pass
```

```
            elif keys[K_RIGHT] or keys[K_d]:
                pass
            elif keys[K_DOWN] or keys[K_s]:
                pass
            elif keys[K_LEFT] or keys[K_a]:
                pass
            sosgame.update()

    if __name__ == "__main__":
        main()
```

8.4.2.2 游戏初始化、创建角色

修改 SosGame 类中的__init__( )、initialization( )、collision ( ) 方法。

```
    def __init__(self,screen):
        """初始化游戏"""
        #游戏背景图序列
            self.image_list =
['./image/background.png','./image/success.png','./image/fail.png']
            #游戏运行状态文本序列
            self.gametext = ['Rescueing…','Rescue success!','Rescue failure!']
            #游戏默认背景图
            self.background = pygame.image.load(self.image_list[0]).convert_alpha()
            self.sos_list = []              #待救援对象
            self.obstacle_list =  []        #障碍物
            self.player = None               #玩家
            self.screen = screen
            self.__gamestate = 0             #游戏状态：0 正在运行 1 援救成功 2 援救失败
            self.totalscore = 0               #当前分数
            self.gameAgainbtn = None        #再玩一次按钮
            self.gameoverbtn = None         #退出游戏按钮
            self.sostime = 0                #游戏进行时间
            self.initialization()
        def initialization(self):
            """创建角色"""
            self.sos_list.clear()             #初始化待援救对象
            self.obstacle_list.clear()         #初始化障碍物
            self.__gamestate = 0            #游戏状态正在救援
            self.totalscore = 0
            self.sostime = 0
            self.gameAgainbtn = None
            self.gameoverbtn = None

            self.player = Player(self.screen)             #创建玩家
            n = random.randint(8,10)                    #随机生成被救援对象的个数
            for i in range(0,n):                         #创建 n 个救援对象
```

```
            sos = Sos(self.screen)
            if not self.collision(sos)[0]:
                self.sos_list.append(sos)

        for i in range(0,10):                    #创建 10 个障碍物
            ob = Obstacle(self.screen)
            if not self.collision(ob)[0]:
                self.obstacle_list.append(ob)
    def collision(self,obj):
        """碰撞检测"""
        return False,None
```

说明：在实际开发中，类的成员属性一般要定义为私有属性，应使用 set 和 get 方法访问，为了使代码简短，在这里仅把表示“游戏状态”属性定义为私有属性。

8.4.2.3 把角色放到游戏窗口中

修改 SosGame 类中的 update（）方法。

```
def update(self):
"""刷新游戏界面"""
        if not self.__gamestate:                          #如果正在救援中
            ticks = pygame.time.get_ticks()
            self.screen.blit(self.background,(0,0))      #填充背景
            for obst in self.obstacle_list:
                self.screen.blit(obst.image,obst.rect)    #显示障碍物
            for sos in self.sos_list:
                sos.update(ticks)                         #更新待救援对象的图像帧
                self.screen.blit(sos.image,sos.rect)      #显示待救援对象
            self.player.update(ticks)
            self.screen.blit(self.player.image,self.player.rect)   #显示玩家角色
            self.displayscore()                           #显示成绩
        else:                                             #否则，显示游戏结束界面
            self.gameover(self.__gamestate)
        pygame.display.update()
```

说明：现在运行程序会引发：AttributeError: 'Obstacle' object has no attribute 'image'异常，这是因为“角色”还没有图像。

8.4.2.4 完善角色

修改 RoleLibrary.py 模块中的 Shape 类、Sos 类、Obstacle 类、Player 类。

```
class Shape(pygame.sprite.Sprite):
    def __init__(self,x,y,width,height,image,columns,target):
        pygame.sprite.Sprite.__init__(self)
        self.target = target
        self.master_image  = pygame.image.load(image).convert_alpha()  #读取角色主图像
        self.frame_width = width              #每一帧的宽
        self.frame_height = height            #每一帧的高
        self.rect = [x,y,width,height]        #角色在游戏窗口中的显示区域
```

```
            self.first_frame = 0                #第一帧序号
            self.frame = 0                      #当前帧序号
            self.pre_frame = -1                 #前一帧序号
            self.columns = columns              #主图像列数，即每行帧数
            rect = self.master_image.get_rect() #主图像区域
            #从主图像中取出当前帧图像
            self.image = self.master_image.subsurface((0,0,width,height))
            #最后一帧序号
            self.last_frame = (rect.width // width) * (rect.height // height) - 1
            self.last_time = 0                          #上次时间

        def update(self,current_time, rate=500):
            """更新精灵的图像帧"""
            #rate：帧图像更新的时间间隔，单位：毫秒
            if current_time > self.last_time + rate:            #如果超过间隔时间
                self.frame += 1
                if self.frame > self.last_frame:
                    self.frame = self.first_frame
                    self.last_time = current_time
            #如果当前帧序号和前一帧序号不相等，则从主图像中重新取出当前帧图像
            if self.frame != self.pre_frame:
                frame_x = (self.frame % self.columns) * self.frame_width
                frame_y = (self.frame // self.columns) * self.frame_height
                rect = ( frame_x, frame_y, self.frame_width, self.frame_height )
                self.image = self.master_image.subsurface(rect)
                self.pre_frame = self.frame

class Sos(Shape):
    def __init__(self,target):
        #target 游戏窗口对象
        x = random.randint(25,525)              #随机产生 x 坐标
        y = random.randint(25,530)              #随机产生 x 坐标
        i = random.randint(0,9)
        #生成待救援角色，参数根据图像实际情况修改
        Shape.__init__(self,x,y,50,40,'./image/sos'+str(i)+'.png',2,target)

class Obstacle(Shape):
    def __init__(self,screen):
        x = random.randint(25,525)
        y = random.randint(25,530)
        i = random.randint(0,5)
        Shape.__init__(self,x,y,60,60,'./image/obstacle'+str(i)+'.png',1,screen)

class Player(Shape):
    def __init__(self,screen):
        Shape.__init__(self,0,500,96,96,'./image/player.png',8,screen)
        self.direction = 2                          #面部方向，由玩家角色实际图像决定
```

```
        self.speed = 120                                    #当前速度阀值
        self.moving = False                                 #玩家角色是否正在行走
    def update(self,current_time):
        """更新精灵的图像帧"""
        #由于要模拟玩家行走时具有加速度的效果，所以需要重写 update( )方法，
        #动态改变时间间隔
        if not self.moving:
            self.speed = 120
            return
        self.first_frame = self.direction * self.columns
        self.last_frame = self.first_frame + self.columns-1
        if self.frame < self.first_frame:
            self.frame = self.first_frame
        if current_time > self.last_time +self.speed:
            self.frame += 1
            if self.frame > self.last_frame:
                self.frame = self.first_frame
                self.last_time = current_time
        if self.frame != self.pre_frame:
            frame_x = (self.frame % self.columns) * self.frame_width
            frame_y = (self.frame // self.columns) * self.frame_height
            rect = ( frame_x, frame_y, self.frame_width, self.frame_height )
            self.image = self.master_image.subsurface(rect)
            self.pre_frame = self.frame
            self.speed -= 1
        #每步移动的距离及方向，案例提供的图像中，上、右、下、左分别对应
        #第 0、2、4、6 行图像，读者稍加分析即可得出表达式的意义。
        step = (120-self.speed )*(-1 if self.direction % 6 ==0 else 1)
        self.move(self.direction,step)    #移动玩家角色

    def move(self,direction,step):
        """改变玩家角色的位置"""
        if direction == 2 or direction ==6:
            self.rect[0] +=step
        else:
            self.rect[1] +=step
        #处理越界
        if self.rect[0]<0 : self.rect[0]=0
        if self.rect[0]>510 : self.rect[0]=510
        if self.rect[1]<0 : self.rect[1]=0
        if self.rect[1]>500 : self.rect[1]=500
```

8.4.2.5 处理键盘事件、移动玩家角色

修改 main( )函数。

```
def main():
  pygame.init()
  screen = pygame.display.set_mode((600,600),0,32)
```

```
pygame.display.set_caption('模拟牧场救援')
timer = pygame.time.Clock()
sosgame = SoSGame(screen)
while True:
    timer.tick(30)
    for event in pygame.event.get():
        if event.type == QUIT:
            pygame.quit()
            sys.exit()
    if  not sosgame.get_gamestate():                #轮询键盘事件
        keys = pygame.key.get_pressed()
        if keys[K_UP] or keys[K_w]:
            sosgame.player.direction = 0
            sosgame.player.moving = True
        elif keys[K_RIGHT] or keys[K_d]:
            sosgame.player.direction = 2
            sosgame.player.moving = True
        elif keys[K_DOWN] or keys[K_s]:
            sosgame.player.direction = 4
            sosgame.player.moving = True
        elif keys[K_LEFT] or keys[K_a]:
            sosgame.player.direction = 6
            sosgame.player.moving = True
        else:
            sosgame.player.moving = False
    sosgame.update()
```

运行程序，玩家角色已能在区域内移动，停止。

8.4.2.6 碰撞检测

碰撞检测是指两个对象是否具体重叠区域的检测，这是开发游戏程序的核心。由于碰撞检测对象是不规则的图形区域，使用 pygame.sprite.collide_mask（）方法实现。修改 SoSGame 类中的 collision( )方法。修改后的代码如下：

```
def collision(self,obj):
    """碰撞检测"""
    #返回值为是否碰撞，碰撞对象
    if obj is not self.player and  pygame.sprite.collide_mask(self.player,obj):   #是否与玩家角色碰撞
        return True,self.player
    for sos in self.sos_list:           #是否与被救援对象碰撞
        if sos is obj:
            return False,None
        elif pygame.sprite.collide_mask(obj,sos):
            return True,sos
    for obst in self.obstacle_list:      #是否与障碍物碰撞
        if obst is obj:
            return False,None
        elif pygame.sprite.collide_mask(obj,obst):
```

```
                return True,obst
        return False,None
```

8.4.2.7 使用碰撞检测结果进行逻辑控制

修改 main()函数，SoSGame 类中的 meetsos （ ）、meetobst（） 方法。

```
def meetsos(self,obj):
   """碰到救援对象"""
         self.sos_list.remove(obj)                #从救援对象序列中删除
         self.totalscore +=30                     #加分
def meetobst(self,obj):
         """碰到障碍物"""
         self.totalscore -=10                     #减分
         direction = (self.player.direction + 4) % 8
         #模拟不能穿越障碍物，反弹效果
         self.player.move(direction,10*(-1    if    direction    %    6    ==0    else    1)    )
         self.player.speed = 120                                  #速度恢复初始值
def  main():
        #省略部分代码
          elif keys[K_LEFT] or keys[K_a]:
              sosgame.player.direction = 6
              sosgame.player.moving = True
          else:
              sosgame.player.moving = False
          f,o = sosgame.collision(sosgame.player)
          if f and isinstance(o,Sos):                             #是救援对象
               sosgame.meetsos(o)
          elif  f and isinstance(o,Obstacle):                     #是障碍物
               sosgame.meetobst(o)
       #省略部分代码
```

说明：在“模拟不能穿越障碍物，反弹效果”处存在 Bug，这是因为当玩家角色到达障碍物时速度可能已经很快，可以一步穿越障碍物，要修复此 Bug，还需要添加一些代码。请读者自行完善。

8.4.2.8 显示当前分数

修改 SoSGame 类中的 displayscore （)方法。

```
def displayscore(self):
         """显示当前分数"""
         font=pygame.font.SysFont('arial',18)
         text=font.render("current Score:"+str(self.totalscore), True, (128,10,10))
         self.screen.blit(text,(30,30))
```

8.4.2.9 检测游戏状态

修改 SoSGame 类中的 update （)、transobpos （)、set_gamestate （)、get_gamestate（） 方法，main( ) 函数。

在主模块中添加以下三个自定义消息：

```
TRANSOB = pygame.USEREVENT +1     #更换障碍物消息
```

```
GAMEOVER = pygame.USEREVENT +2    #救援失败消息
GAMESUCC = pygame.USEREVENT +3    #救援成功消息

def update(self):
  """刷新游戏界面"""
  if not self.__gamestate:                              #如果正在救援中
ticks = pygame.time.get_ticks()
      if len(self.sos_list) == 0:
          succ_event = pygame.event.Event(GAMESUCC, message="救援成功!")
          self.sostime = ticks // 1000
          pygame.event.post(succ_event)           #发送救援成功消息
      if self.totalscore < 0:
          over_event = pygame.event.Event(GAMEOVER, message="救援失败!")
          self.sostime = ticks //1000
          pygame.event.post(over_event)           #发送救援失败消息
      self.screen.blit(self.background,(0,0))        #填充背景
      #省略后面代码
class SoSGame():
    #省略部分代码
    def transobpos(self):
         """更换障碍物位置"""
         self.obstacle_list.pop()
         while True:
           ob = Obstacle(self.screen)
           if not self.collision(ob)[0]:
              p = random.randint(0,4)
              self.obstacle_list.insert(p,ob)
              break;
def set_gamestate(self,state):
        """设置游戏状态"""
           self.__gamestate = state
    def get_gamestate(self):
           """获取游戏状态"""
           return self.__gamestate
   #省略后面代码
def main():
  #省略部分代码
for event in pygame.event.get():
      if event.type == QUIT:
          pygame.quit()
          sys.exit()
    elif event.type == GAMESUCC:
          sosgame.set_gamestate(1)              #更改游戏状态为成功
    elif event.type  == GAMEOVER:
          sosgame.set_gamestate(2)              #更改游戏状态为失败
    elif event.type == TRANSOB:
          sosgame.transobpos()                  #更改障碍物位置
```

```
    #省略后面代码
```

8.4.2.10 定时发送移动障碍物消息

在 main()函数中添加：pygame.time.set_timer(TRANSOB,5000)

8.4.2.11 处理游戏结束

在 RoleLibrary.py 模块中添加 Button 类、修改 SoSGame 类中的 gameover（ ）方法、修改 main()函数

```
   class Button(object):
    def __init__(self, upimage, downimage, position):
        self.image_up = pygame.image.load(upimage).convert_alpha()
        self.image_down = pygame.image.load(downimage).convert_alpha()
        self.position = position

def is_over(self):
    """判断鼠标是否在按钮的区域"""
        point_x, point_y = pygame.mouse.get_pos()
        x, y = self.position
        w, h = self.image_up.get_size()
        x -= w/2
        y -= h/2
        in_x = x < point_x < x + w
        in_y = y < point_y < y + h
        return in_x and in_y

def render(self, surface):
    """根据鼠标的位置显示按钮的图像"""
        x, y = self.position
        w, h = self.image_up.get_size()
        x -= w/2
        y -= h/2
        if self.is_over():
            surface.blit(self.image_down, (x, y))
        else:
            surface.blit(self.image_up, (x, y))

def gameover(self,state):
         """游戏结束"""
          self.screen.fill([255,255,255])
          image = pygame.image.load(self.image_list[state]).convert_alpha()
          rect =  image.get_rect()
          self.screen.blit(image,((600-rect.width)//2,50))
          font=pygame.font.SysFont('arial',50)
          text=font.render(self.gametext[state], True, (234,162,0))
          self.screen.blit(text,(150,350))
          font=pygame.font.SysFont('arial',30)
```

```
            text=font.render('Score: '+str(self.totalscore)+'   Rescue time:
'+str(self.sostime)+'s', True, (234,162,0))
            self.screen.blit(text,(140,420))
            self.gameAgainbtn = Button('./image/again_down.png','./image/again_up.png',(400,550))
            self.gameoverbtn = Button('./image/over_down.png','./image/over_up.png',(500,550))
            self.gameAgainbtn.render(self.screen)
            self.gameoverbtn.render(self.screen)

    def main():
     #省略部分代码
             elif  f and isinstance(o,Obstacle):                #是障碍物
                  sosgame.meetobst(o)
             elif sosgame.gameoverbtn:                          #如果游戏已经结束
                    if event.type == pygame.MOUSEBUTTONUP:      #处理鼠标事件
                       if sosgame.gameoverbtn.is_over():        #在“退出游戏”上单击
                          pygame.quit()
                          sys.exit()
                       elif sosgame.gameAgainbtn.is_over():     #在“再玩一次”上单击
                          sosgame.initialization()
        sosgame.update()
```

限于篇幅，就不在此贴出源代码，源代码请从铜仁一中校园网站下载：http://i.tryz.net/html/2018/python/pythonjc.rar。本案例是使用面向对象的思维方式开发的小游戏，旨在让同学们了解游戏的本质和使用 Python 编程的乐趣。还有很多地方需要完善。

附录一 Python 库简介

Python 中只有几十个函数是随 Python 的启动就直接加载到内存中的，可以直接使用，这些函数叫做内建函数（Built_in Funtions，是标准库的一部分）。但还有更多的函数并不能直接使用，需要通过导入后才能使用。Pytthon 中的函数是以库的形式提供给我们使用。库分为标准库和第三方库。标准库是随着 Python 解释器一起安装在电脑中的，它是 Python 的一个组成部分。第三方库是由一些 Python 爱好者编写供大家免费使用的库，需要单独下载安装才能使用。python 编程的挑战很大一部分来自于对库的应用，一旦掌握了核心语言，就需要花大量时间来研究各种内建函数和库。

这里，我们根据信息技术新课标模块结构，选择几个常用库进行介绍，通过附件资源的形式提供给同学们学习。目前包含 Python3.7 内建函数、Python 库、Numpy 库、Matplotlib 库、Pygame 库、Requests 库、Beautiful Soup 库。随着新课程的实施，我们会有针对性及时更新或增加库介绍内容。下载地址：

http://i.tryz.net/html/2018/python/pythonjc.rar

附录二 各章练习题参考答案

参考答案下载：

http://i.tryz.net/html/2018/python/pythonjc.rar

附录三 参考文献

本书参考文献列表：

[1]樊磊，《中小学人工智能课程实践入门》PDF 文档

[2]秦颖，《Python 实用教程》，清华大学出版社

[3] [美]David Beazley Brian K.Jones 著 陈舸 译 《Python Cook Book》人民邮电出版社

[4] [美]Swaroop, C. H. 著 沈洁元 译 《简明 Python 教程》

[5]中华人民共和国中央人民政府.国务院关于印发〈新一代人工智能发展规划〉的通知.[DB/OL].http://www.gov.cn/zhengce/content/2017-07/20/content_5211996.htm, 2017-07-08.

[6]中华人民共和国教育部.教育部关于印发《教育信息化 2.0 行动计划》的通知.[DB/OL].http://www.moe.gov.cn/srcsite/A16/s3342/201804/t20180425_334188.html, 2018-04-18.

[7] Python 库资源大全：https://zhuanlan.zhihu.com/p/27350980

[8]Python 常用的标准库以及第三方库：
https://www.zhihu.com/question/20501628/answer/223340838

[9]Titanic Data Science Solutions：
https://www.kaggle.com/startupsci/titanic-data-science-solutions

[10]中国人功智能产业链分析：https://mp.weixin.qq.com/s/iq4wXxMnugNDoa11NyLT6A

[11]吴军，《智能时代》，中信出版集团

[12]布梅雷迪斯·布鲁萨德（Meredith Broussard），《人工不智能》

[13]尤小平，《学历案与深度学习》，华东师范大学出版社