**病毒流行规律**

# 一、问题重述

对于给定浙江省82家大型的医院的每日报告的五种流行病毒有关的流行病的病例数的五年数据，相应地区的五年气候数据。本题拟完成以下几个任务：

1. 建立反映病毒流行病规律的数学模型，并进行相应的预测；
2. 因为病毒数据相对于气象数据有一定的滞后性，查阅有关资料，对气象数据和病毒流行病数据进行匹配，建立模型，滞后性进行定量研究；
3. 建立模型，研究气候变化和病毒流行病流行相互关系；
4. 通过查阅病毒特性的资料，建模分析五种病毒流行病的其它致病原因；
5. 依据以上研究对这五种病毒流行病的防治给出有效建议。

# 二、问题分析

## 2.1 问题一的分析

病毒流行病的发生与众多因素有关，题目给出了各个病例中的一些医疗信息，包括患者的一些基本信息，主要有患者的性别、出生日期、职业和食物种类，还给出了14年到18年浙江省各地的环境量化因素。在研究的时候可以分开两个部分讨论，一是患者的个人因素，探究性别、年龄、地区等因素是否对患病概率有显著性影响，再讨论环境因素对患病率的影响。由于数据分散再众多表格中，因此首先要对数据进行清洗汇总，根据研究目标对数据进行分组，统计出不同年龄段、不同地区、不同性别的患病人数，并根据患病人数的趋势预测以后一年的患病特征。

## 2.2 问题二的分析

病毒数据相对于气象数据有一定的滞后性，而对带有滞后性的数据做相关分析会有一定的误差，因此在研究气象数据和病毒流行病数关系是可以先不考虑滞后性的问题，对数据进行拟合，得出初步的函数关系式。再考虑滞后性的影响，把气象数据进行平滑处理，再对气象数据和患病数进行拟合处理，而平滑的间隔需要逐个分析，而最合适的平滑间隔可以根据拟合函数的拟合度或患病率与气象数据的相关性进行选择。

## 2.3 问题三的分析

问题三的解答是建立于问题二之上，如果气象数据与患病数据存在滞后性，那么对经过平滑后的气象数据和患病数据进行拟合分析，若不存在滞后性，则直接分析。拟合模型首先考虑线性回归，如果拟合度不能令人满意再考虑非线性回归或神经网络拟合。

## 2.4 问题四的分析

在患病数据中还给出了五种患病类型，受不同环境和每个人都有着不同体质的影响，患病类型会出现多样性。这是一个分类任务，支持向量机（SVM）在分类学习中有着出色的表现，俗称万能分类器。鉴于数据中存在标签变量，同时为了更形象解释出哪些因素对患病类型有显著详细，可以使用决策树模型。

## 2.5 问题五的分析

相信经过以上几个问题，可以得出病毒流行病的一些致病因素和发展规律，同时通过查阅相关资料，针对不同的人群、不同的环境因素对大众提出合理性的建议。

# 三、模型假设

1. 题目所给的数据完全真实可靠；
2. 病毒流行病只与气候因素和自身身体质素有关，不考虑其他致病对病毒流行病造成的影响；
3. 各地区的人口保持原有的发展趋势，不考虑人口迁移对各地区的人口造成的影响。

# 四、符号说明与定义

|  |  |
| --- | --- |
| 符号 | 说明 |
|  | 患病人数 |
|  | 地区总人口 |
|  | 患病率 |
|  | 患病类型标签 |
|  | 时间 |
|  | 经度 |
|  | 维度 |
|  | 平均气温 |
|  | 日最高气温 |
|  | 日最低气温 |
|  | 平均相对湿度 |
|  | 降水量 |
|  | 平均风速 |
|  | 日照相对数 |

* 患病人数为疾病数据中在某地区某一时间段中个案数；
* 患病率=，因为每个地区的人口不同，为了公平计量每个地区的患病情况的严重程度，以患病率作为预测值。

# 五、模型建立与求解

## 5.1 问题一的模型建立与求解

由于数据分散在若干个表格中，为了方便统一分析，在建模之前先利用Excel和python对数据进行清洗汇总，并根据实际需要进行分组描述统计。首先从个人基本因素对患病情况的分组，单考虑性别的影响因素（见图5.1），四年中男患者人数占了53%，而女患者占了47%。若考虑不同的职业人群（见图5.2），农民群体的患病率最高，这有可能跟农民的工作环境相对恶劣有关。

图 5.1 不同性别的患者人数

图 5.2 不同职业群体的患者人数

若针对不同年龄段的患病人数来看，最年长的患者为109岁，从图下图中可以看出，幼儿的患病人数出现尖峰，这个可能由于幼儿自身体抗力不强。而作为老年群体，患病人数虽然极少，但考虑到超年老的人群数量不多，且没有收集到浙江地区的人口分布情况，难以判断年老的患病率是高或低。

图 5.3 不同年龄段的患者人数

探讨过个人因素对患病人数的影响，以下谈论不同地区的患病率情况，从统计表来看，患病率最高的城市为舟山市，而舟山市的人口也是最少的，同样舟山市在浙江的经济水平是倒数的一个水平，高患病率可能与当地的城市保洁卫生做得不足有关。当然，这结论是基于没有传染病的假设前提下。

表 5.1 各个城市的患病率

|  |  |  |  |
| --- | --- | --- | --- |
| 城市 | 患病人数 | 总人口 | 患病率（%） |
| 丽水市 | 52393 | 2680300 | 1.954743872 |
| 台州市 | 49837 | 6001700 | 0.830381392 |
| 嘉兴市 | 43149 | 3521100 | 1.225440913 |
| 宁波市 | 69191 | 5909600 | 1.170823744 |
| 杭州市 | 86785 | 7359900 | 1.179160043 |
| 温州市 | 69542 | 8182200 | 0.849918115 |
| 湖州市 | 42673 | 2648400 | 1.611274732 |
| 绍兴市 | 40570 | 4445300 | 0.912649315 |
| 舟山市 | 28196 | 937200 | 3.008536065 |
| 衢州市 | 33288 | 2574800 | 1.292838279 |
| 金华市 | 66167 | 4811500 | 1.375184454 |

图 5.3 各个城市的患病率

以下是基于每个月的患病总人数的分析，把数据放到SPSS中作时间序列分析，可以发现患病人数呈现明显的季节性循环，每年六月份开始，患病开始暴增，一直到年底，随后开始下降，逐年循环。鉴于如此规律，在此对患病人数建立ARIMA模型来预测未来一年对患病人数。

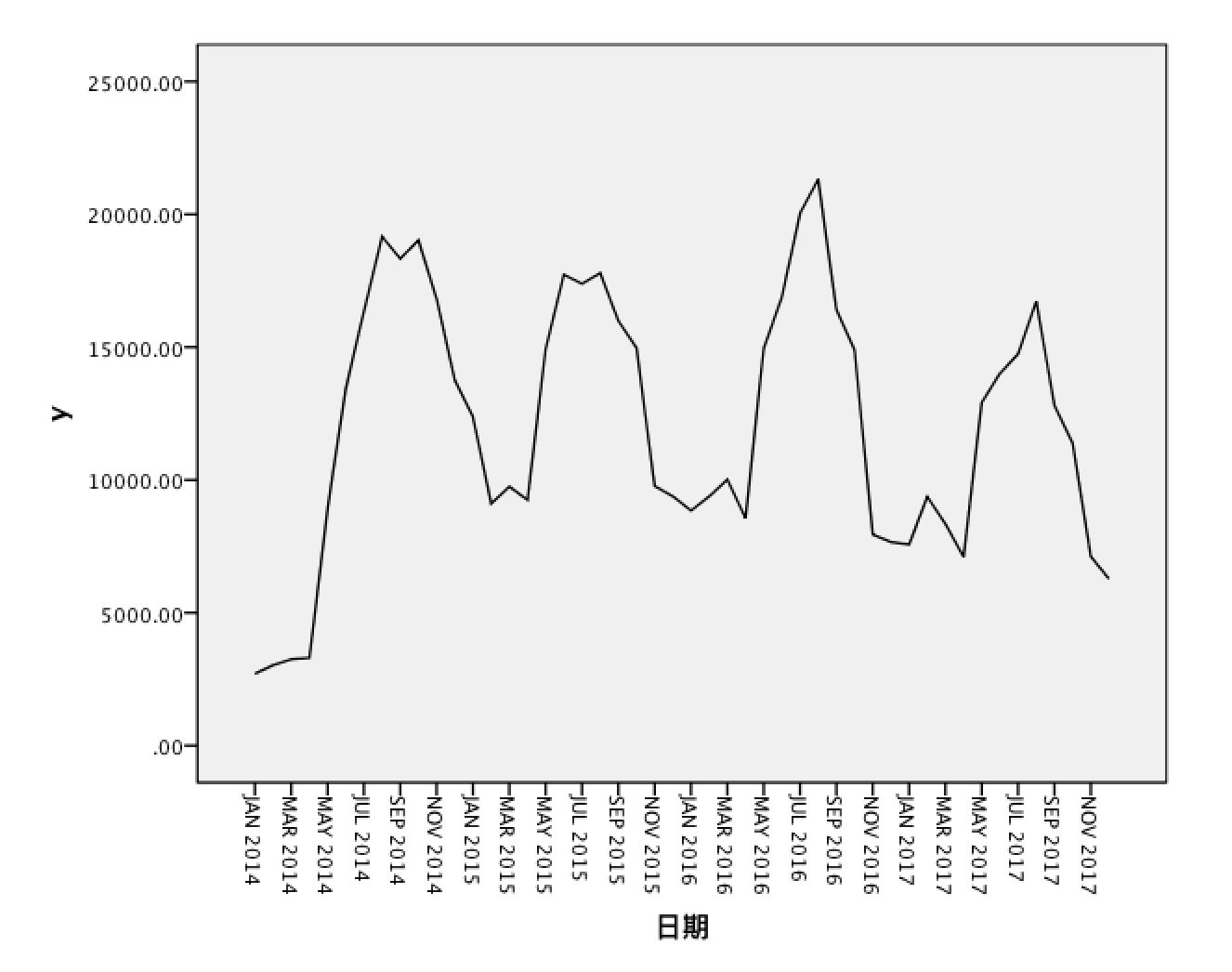


图5.4 不同时间下的患病人数

在时间序列的时域分析中，线性差分方程是极为有效的工具。事实上，任何一个

ARMA 模型都是一个线性差分方程。在此考虑到数据的周期性，建立了ARIMA（1，1，6）模型，得到R方为0.835的高可决系数，认为这次的拟合预测十分可靠。

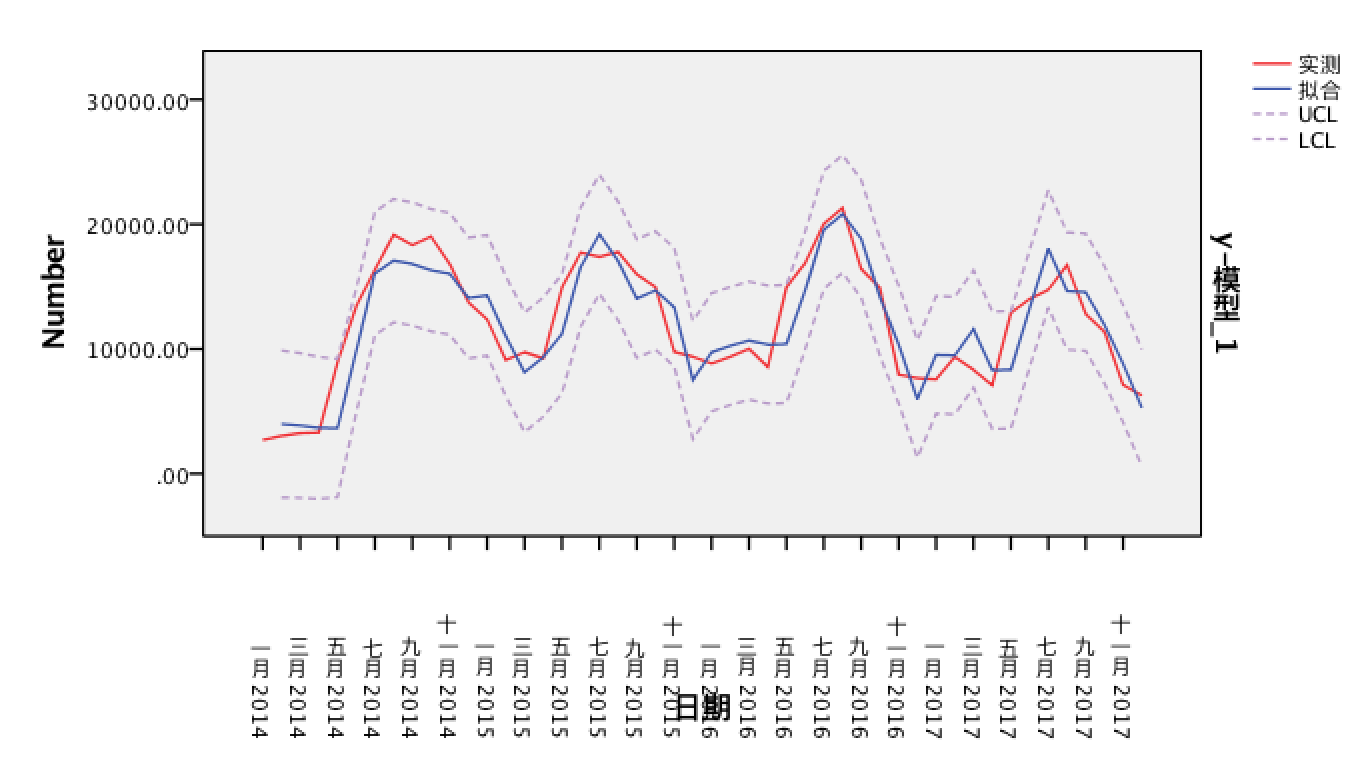


图5.5 ARIMA模型下的患病人数拟合

## 5.2 问题二、三的模型建立与求解

病毒数据相对于气象数据有一定的滞后性，因此先对各个环境因素变量和患病率进行皮尔逊相关分析，从下面的相关矩阵来看，患病率跟日最低气温、平均相对湿度、降水量有一定的相关性外，其他因素似乎没有太多关系。

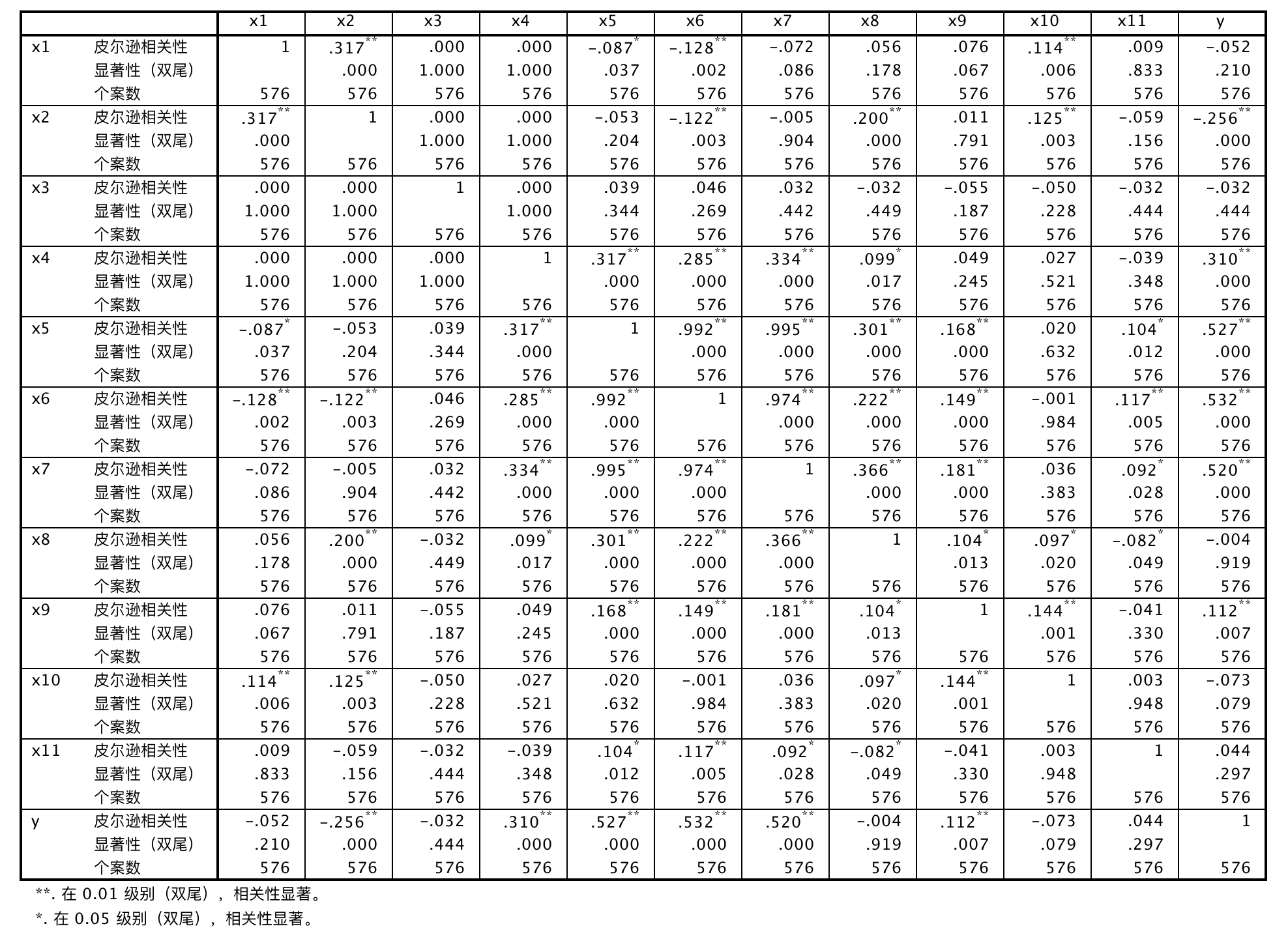


图5.6 气候数据与患病数据相关矩阵

因此在此认为病毒数据相对于气象数据日最低气温、平均相对湿度、降水量具有一定的滞后性，而其他因素则不存在滞后性。取存在滞后性的三个因素的平均相关系数作为衡量整体滞后程度，对气候数据进行向前平滑一个或几个月，再重新计算平均相关系数，若平均相关系数得到提升，则认为存在滞后性。对每个平滑周期进行测试计算，得到需要平滑的时间单位为一个月。即6月份受到恶劣气候对影响，患者不会立即患病，而是差不多等了一个月之后才开始患病。实际情况下这个延迟反应时间不会过长，在这里以月为单位是受限于气候数据以月为单位作为统计。假若把统计单位精确到天的话，想象计算精度会更高。

尽管把数据的滞后性考虑到模型中去后，然而整体的相关系数提升的幅度不大，说明建立线性回归的拟合度不会太高。在此应用BP神经网络对气候数据和患病数据进行拟合。

BP神经网络是一种多层前馈神经网络，该网络的主要特点是信号前向传递，误差反向传播。在前向传递中，输入信号从输入层经隐含层逐层处理，直至输出层。每一层的神经元状态只影响下一层神经元状态。如果输出层得不到期望输出，则转入反向传播，根据预测误差调整网络权值和阈值，从而使BP神经网络预测输出不断逼近期望输出[3]。BP神经网络的拓扑结构如下图：

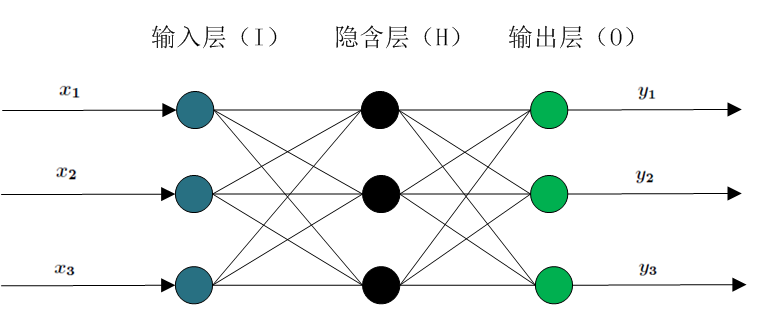


图 5.7 BP神经网络结构图

BP神经网络预测前首先要训练网络，通过训练使网络具有联想记忆和预测能力。BP神经网络的训练过程包括以下几个步骤：

Step 1:网格初始化。根据系统输入输出序列确定输入层节点数、隐含层节点数，输出层节点数，初始化输入层、隐含层和输出层神经元之间的连接权值，初始化隐含层阈值，输出层阈值，给定学习速率和神经元激励函数。

Step 2:隐含层输出计算。计算公式如下：

其中味隐含层激励函数，该函数有多种选择，在本研究中选用tanh函数。

Step 3:输出层输出计算，计算BP神经网络预测输出

BP神经网络的基本计算为前三个步骤，完整的BP网络训练还包括误差计算、权值更新、阈值更新、判断算法是否收敛等[2]，但由于计算过于复杂，在此不作描述。在本次研究中，输入的变量为经纬度和其余几个气候变量，在Matlab进行神经网络训练，除了个别异常样本点外，拟合结果和真实结果的误差非常小，其拟合效果见下图所示：

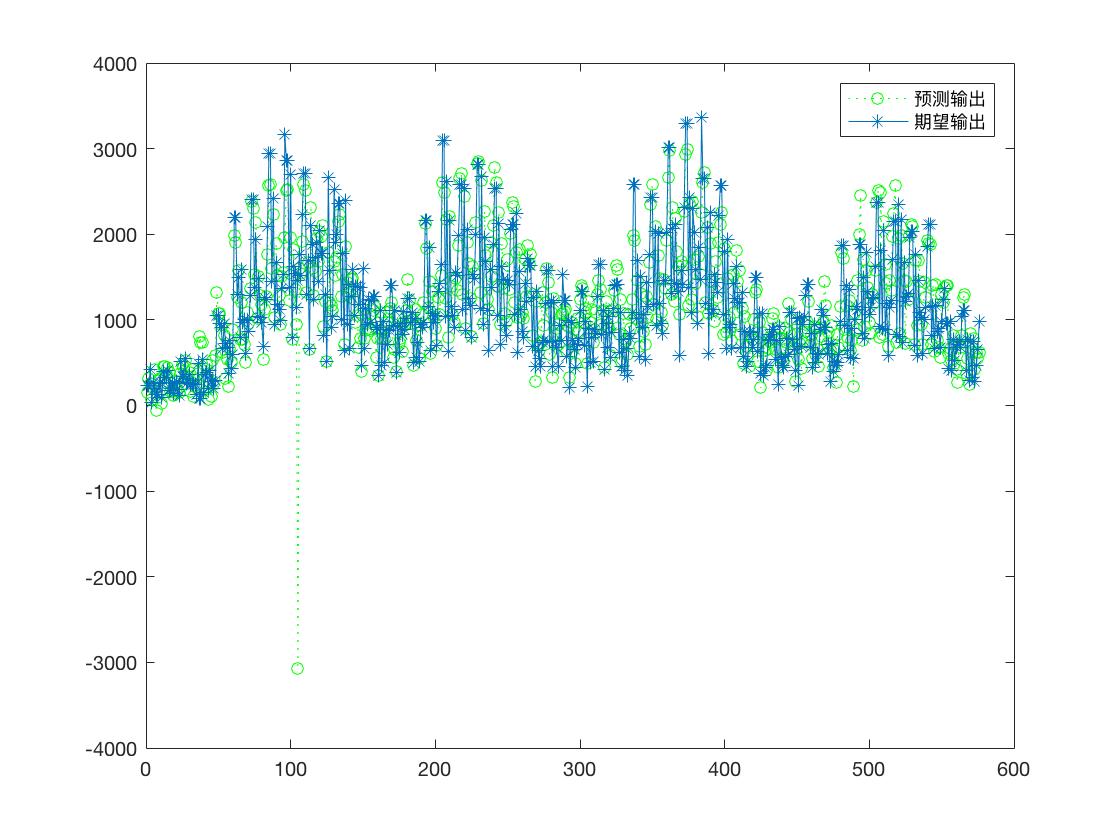


图5.8 BP神经网络的拟合效果

## 5.3 问题四模型建立与求解

在患病数据中还给出了五种患病类型，受不同环境和每个人都有着不同体质的影响，患病类型会出现多样性。鉴于数据中存在标签变量，同时为了更形象解释出哪些因素对患病类型有显著详细，可以使用决策树模型。

一般地，一棵决策树包含一个根结点、若干个内部结点和若干个叶结点。叶结点对应于决策结果，其他结点则对应于一个属性测试；每个结点包含的样本集合根据属性测试的结果被划分到子结点中；根结点包含样本全集。从根结点到每个叶结点的路径对应了一个判定测试序列。决策树学习的目的就是为了产生一棵泛化能力强，即处理未见示列能力强的决策树，其基本流程遵循简单且直观的“分而治之(divide-and-conquer)”策略[1]。算法流程如下：

---------------------------------------------------------------------

输入：训练集

属性集.

过程：函数

1:生成结点node；

2:if 中样本全属同一类别 then

3: 将node 标记为类叶结点；return

4: end if

5:if 为空集或中样本在上取值相同 then

6: 将node标记为叶结点，其类别标记为中样本数最多的类；retuen

7: end if

8: 从中选择最优划分属性；

9: for 的每一个值 do

10: 为node生成一个分支；令表示中在上取值为的样本子集；

11: if 为空集 then

12: 将分支结点标记为叶结点，其类别标记为中样本最多的类；

13: else

14: 以为分枝结点

15: end if

16: end for

输出：以node为根结点的一颗决策树

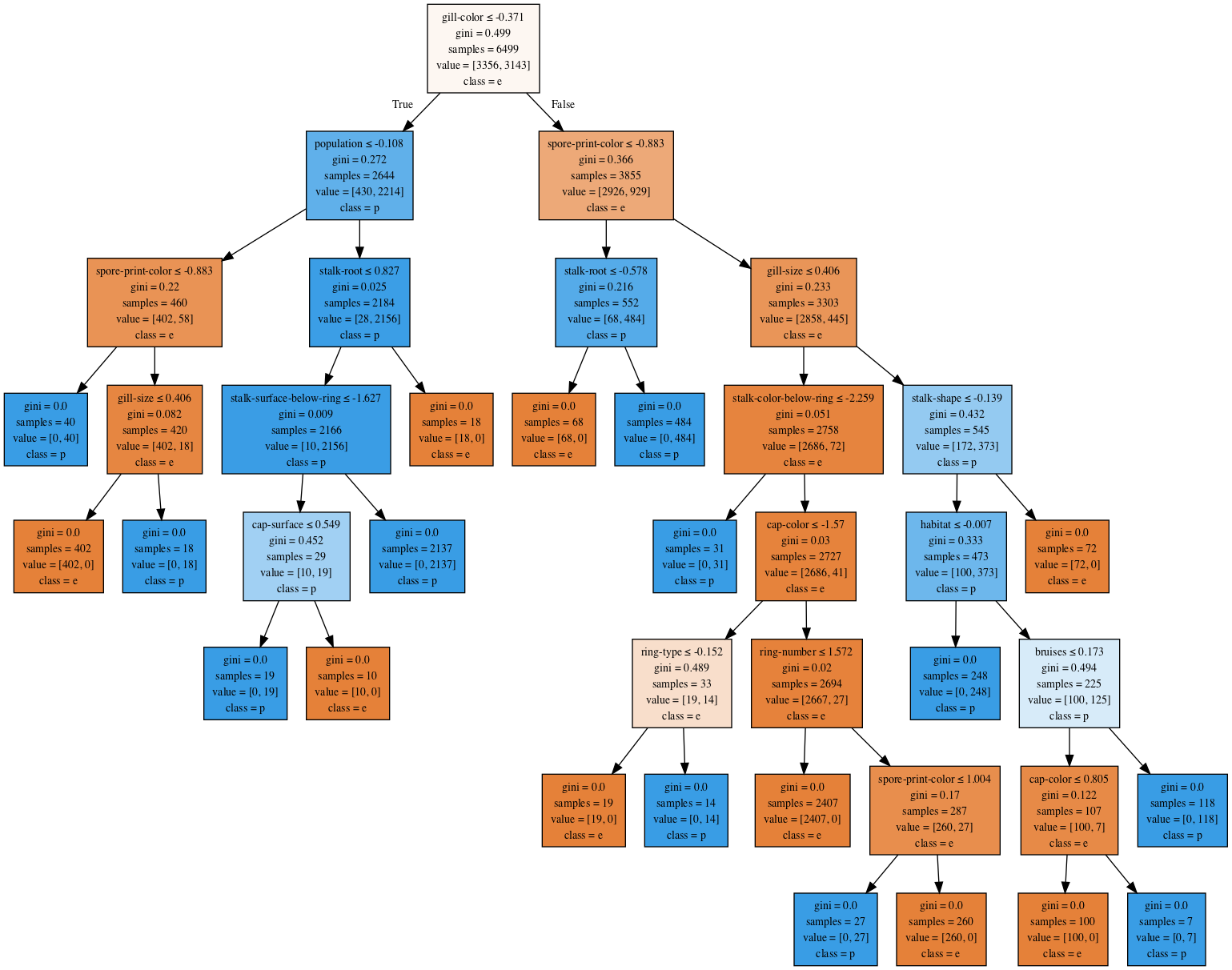


图5.9 决策树可视化

参考文献

[1] 周志华. 机器学习 : = Machine learning[M]. 清华大学出版社, 2016.

[2] 唐鹤隽. 多元回归和神经网络在武汉市房地产预测中的应用比较[D].华中科技大学,2013.

[3] MATLAB中文论坛. MATLAB神经网络30个案例分析[M]. 北京航空航天大学出版社, 2010.

附录

附录一：Matlab代码

data=xlsread('data.xlsx','A1:L576')

X=data(:,1:11)';

y=data(:,12)';

[inputn,inputps]=mapminmax(X);

[outputn,outputps]=mapminmax(y);

net=newff(inputn,outputn,12);

net.trainParam.epochs=1000;

net.trainParam.lr=0.1;

net.trainParam.goal=0.00001;

net=train(net,inputn,outputn);

inputn\_test=mapminmax('apply',X,inputps);

an=sim(net,inputn\_test);

BPoutput=mapminmax('reverse',an,outputps);

figure(1)

plot(BPoutput,':og')

hold on

plot(y,'-\*')

legend('预测输出','期望输出')

附录二：python3代码

import pandas as pd

data01=pd.read\_csv('疾病数据01.csv')

data02=pd.read\_csv('疾病数据02.csv')

data03=pd.read\_csv('疾病数据03.csv')

data04=pd.read\_csv('疾病数据04.csv')

data05=pd.read\_csv('疾病数据05.csv')

data=pd.merge(data01,data02,how='outer')

data=pd.merge(data,data03,how='outer')

data=pd.merge(data,data04,how='outer')

data=pd.merge(data,data05,how='outer')

data.to\_csv('merge\_data.csv')

year=[]

month=[]

k=1

for line in data['检测日期']:

lineArr=line.split('-')

year.append(lineArr[0])

month.append(lineArr[1])

data['year']=year

data['month']=month

data['date-city']=data['year'].map(str)+'-'+data['month'].map(str)+data['医疗机构市']

weather\_city=pd.read\_csv('weather\_city.csv')

for i in range(len(weather\_city['month'])):

if weather\_city.iloc[i,6]==1:

weather\_city.iloc[i,6]='01'

elif weather\_city.iloc[i,6]==2:

weather\_city.iloc[i,6]='02'

elif weather\_city.iloc[i,6]==3:

weather\_city.iloc[i,6]='03'

elif weather\_city.iloc[i,6]==4:

weather\_city.iloc[i,6]='04'

elif weather\_city.iloc[i,6]==5:

weather\_city.iloc[i,6]='05'

elif weather\_city.iloc[i,6]==6:

weather\_city.iloc[i,6]='06'

elif weather\_city.iloc[i,6]==7:

weather\_city.iloc[i,6]='07'

elif weather\_city.iloc[i,6]==8:

weather\_city.iloc[i,6]='08'

elif weather\_city.iloc[i,6]==9:

weather\_city.iloc[i,6]='09'

weather\_city['date-city']=weather\_city['year'].map(str)+'-'+weather\_city['month'].map(str)+weather\_city['tzm']

data=data[['患者性别','患者职业','是否为疑似食源性病例', '是否采集生物样本','初步诊断','食品名称', '食品分类','医疗机构市']]

data.to\_csv('DT\_data.csv')

patient\_data=data.groupby('date-city').count()

patient\_data.to\_csv('patient\_data.csv')

patient\_data['date-city']=patient\_data.index

totData=pd.merge(weather\_city,patient\_data,left\_on='date-city',right\_on='date-city')

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.model\_selection import GridSearchCV

from sklearn.metrics import accuracy\_score

from sklearn.metrics import roc\_auc\_score

data =pd.read\_csv('DT\_data.csv')

from sklearn.preprocessing import LabelEncoder

labelencoder=LabelEncoder()

for col in data.columns:

data[col]=labelencoder.fit\_transform(data[col])

y=data['检测项编号']

X=data.drop('检测项编号',axis=1)

from sklearn.model\_selection import train\_test\_split

X\_train,X\_test,y\_train,y\_test = train\_test\_split(X,y,train\_size=0.8,random\_state=0)

columns=X\_train.columns

from sklearn.preprocessing import StandardScaler

ss\_X =StandardScaler()

ss\_y =StandardScaler()

X\_train=ss\_X.fit\_transform(X\_train)

X\_test=ss\_X.fit\_transform(X\_test)

from sklearn.tree import DecisionTreeClassifier

model\_tree=DecisionTreeClassifier()

model\_tree.fit(X\_train,y\_train)

y\_prob=model\_tree.predict\_proba(X\_test)[:,1]

y\_pred=np.where(y\_prob>0.5,1,0)

model\_tree.score(X\_test,y\_pred)

#可视化树图

data\_=pd.read\_csv("merge\_data.csv")

data\_feature\_name=data\_.columns[1:]

data\_target\_name=np.unique(data\_['检测项编号'])

import graphviz

import pydotplus

from sklearn import tree

from IPython.display import Image

import os

os.environ['PATH']+= os.pathsep

dot\_tree=tree.export\_graphviz(model\_tree,out\_file=None,feature\_names=data\_feature\_name,class\_names=data\_target\_name,filled=True,special\_characters=True)

graph=pydotplus.graph\_from\_dot\_data(dot\_tree)

img=Image(graph.create\_png())

graph.write\_png('out.png')