# Assumptions and Justifications

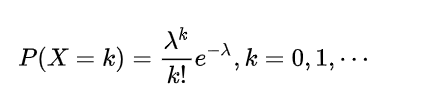
1. 上传数据大体上是真实的，符合游戏规则的
2. 数据代表的都是真实的人，他们解题的思路没有发生重大的变化

# model 1

报告的结果数量每天都有所不同。开发一个模型来解释这种变化，*并使用您的模型创建一个关于2023年3月1日报告结果数量的预测区间*。

1报告总人数和时间的关系比例

尝试1：



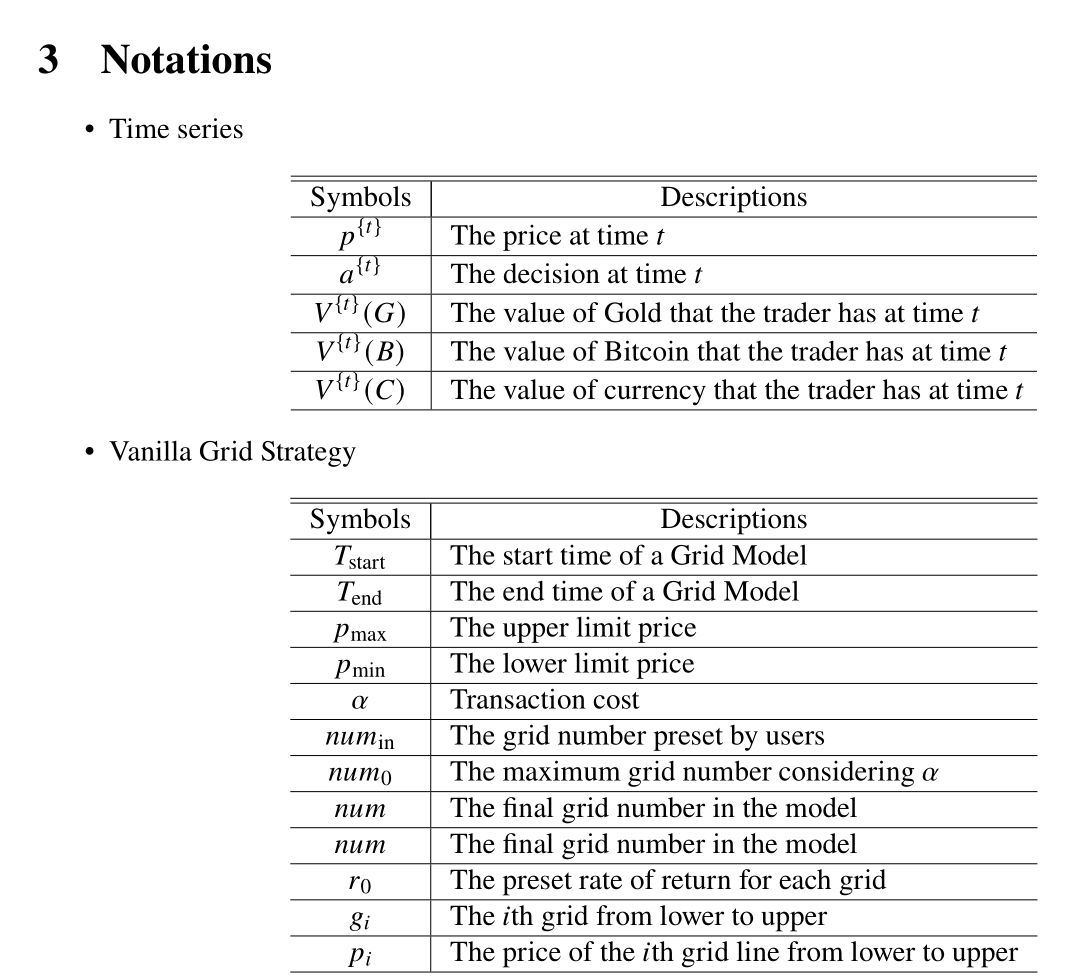
利用泊松分布模型的概率函数，创建线性回归（）

尝试1可能的问题：倍数相差太小（？2w与35w相比也有17倍）

不够拟合

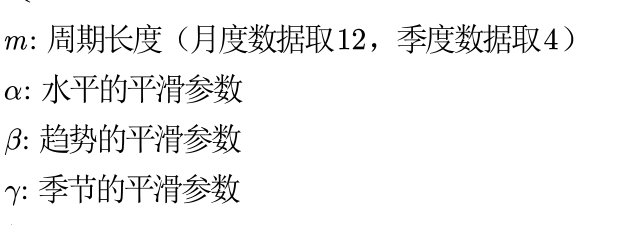
非线性回归

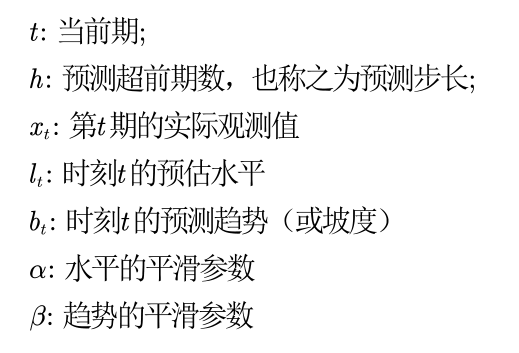
**暂时确定：**



**记号部分类似如此，分模型书写**

记号：





* 模型总览：

为预测报告结果数量的变化，我们使用回归分析和时间序列分析两大方向，并比较两者利弊。观察报告结果数量数据的可视化视图，我们使用对数线性回归模型，得到了拟合系数较高的结果，但会有过拟合的结果。分析影响报告结果数量的实际因素后，我们认为一周七天中报告数量的变化有一定的周期性规律，即存在季节性波动因素。于是我们使用Holt改良后的指数平滑模型，即温特乘法模型。计算ACF和PACF得到拟合情况，计算置信区间（写的不好）

* 回归分析（先没写）
* 温特乘法模型

para1：判断应该用时间序列预测（按实际意义）

时间序列的因素：



para2：判断模型存在季节性（季节性分解+实际意义-周）

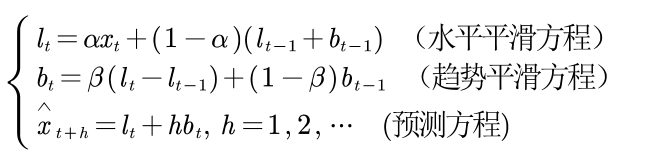


m取7（待补充）

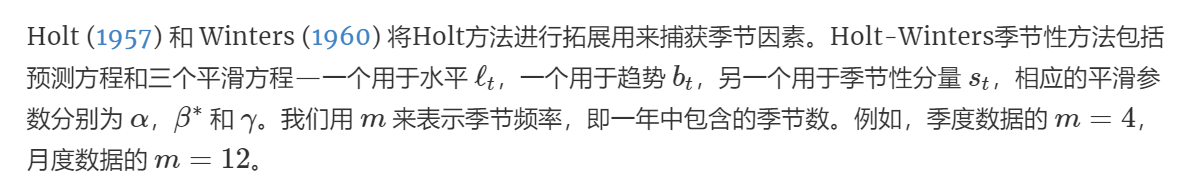
~~para2.9 holt简单季节性模型（winter模型基础）~~

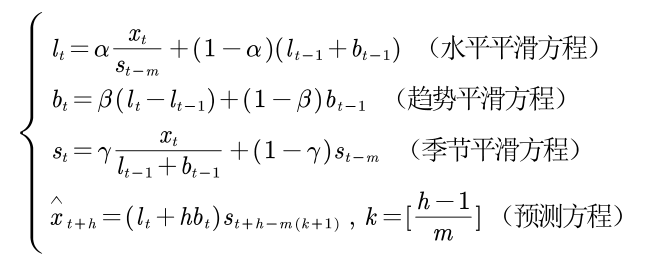
~~Holt 在1957年把简单的指数平滑模型进行了延伸，能够预测包含趋势的数据，~~

~~该方法包含一个预测方程和两个平滑方程（一个用于水平，另一个用于趋势）~~

~~~~

para3：介绍winter模型





解释：

注意到lt xt是接近预测数据的

bt相当于是lt的差分

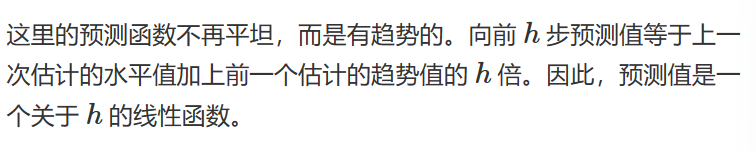
st是指数，自己与自己波动

当前时刻的预估水平由此期的实际观测值除以前一周期季节性**指数**和前一数据的【预估水平趋势和】决定。具体决定使用加权平均数，权值为水平平滑参数α，α的确定由最大化拟合系数决定（最优拟合）

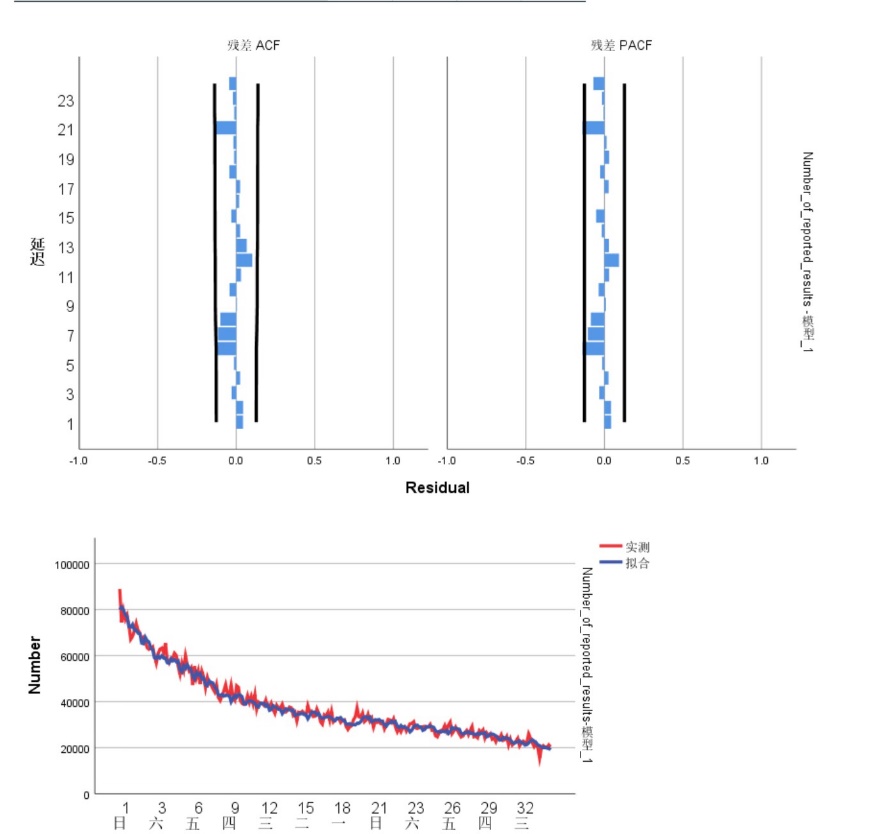
当前时刻的预估趋势由lt的差分与上一时刻的预估趋势决定。具体决定使用加权平均数，权值为趋势平滑参数β，β的确定由最大化拟合系数决定（最优拟合）

当前时刻的预估季节指数由【此期的实际观测值与【预估水平趋势和】的比】与上一时刻的预估季节指数决定。具体决定使用加权平均数，权值为季节平滑参数γ，γ的确定由最大化拟合系数决定

当前时刻的预测值为【当前时刻的预估水平趋势和】与前一期的预估季节指数的乘积，这也是为什么成为“乘法模型”



后面的具体要交给编程手



区间确定：

~~直接拿e^（kx+b）[+x]拟合（k为负数）~~

~~会过拟合~~

~~在确定预测模型后，我们根据每日的波动，确定预测区间~~

# model 2

是否有单词的属性会影响报告结果的比例？

对于未来日期的**给定解决方案单词**，开发一个模型，使您可以**预测报告结果的分布**。换句话说，预测未来日期的相关百分比（1、2、3、4、5、6、X）的分布。*您对您模型的预测有多自信？请举一个关于2023年3月1日单词EERIE的预测的具体例子。*

*您的模型和预测有哪些不确定性？*

思路部分：

注意到这里与时间无关

——需要我们自己寻找确定所谓的词的属性，所以一定存在某种属性影响这个比例

那我们需要对单词做出一个分析，大概是单词->特征向量->报告分布

即特征工程

通过分析，我们使用如下四个特征

* **辅音频率（可分裂为常见辅音+不常见辅音）**
* **音节数**
* **常用度**
* **单词信息熵**

首先需要处理出这四个特征的具体数据

辅音频率——why辅音：元音过于普遍，信息熵很低，无法作为因素

具体：使用26个字母跑主成分分析（使用excel统计：似乎软件可以写入）

音节数——使用python库pronouncing处理大部分数据。部分单词未收录产生异常人工标记数据

常用度——

<https://github.com/hermitdave/FrequencyWords/tree/master>

&c++统计

信息熵——

考虑所有3^5种猜测情况能给猜测过程带来的信息

一种猜测的概率：灰4/5 黄一个得4/25，绿一个得1/25

——改进，加入频率

？

遍历所有的猜测，灰1分，黄4分，绿20分

但是我们以x遍历结果情况（3^5），——改进，加入词语频率

x/8000

i）26^5种，每种计分

ii）\_word词，每种计分

iii）\_word词，每种记信息熵，后根据词语频率算平均数

E=sigma(词数) 频率\*词信息熵

词信息熵=结果情况得分熵形式：

~~按每个字母同等：4/5 ；4/25；1/25~~

1. 按单词，词-词比得到情况， **logp的相差**（O（nm k））
2. 按字母，词-词比得到情况， Iogp的相差 五个相乘

按每个字母不同等：共 5\* \_word字母，

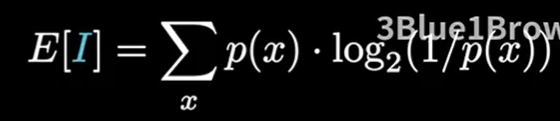
对位置val\_pos/\_word

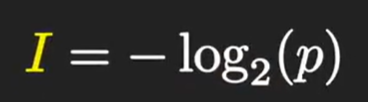
错位置 ——1-词没有-错位置

“词没有”——1-有的词/\_word

黄了一个e比黄了一个w要容易，所以信息熵也是前者小于后者

以此统计即可





~~在letter频率下，出现一个zzzzz单词的可能性非常小，信息熵很大~~

难度得分函数：（贡献法）

按猜测成功获得1分。

1次猜测就成功则每次猜测的贡献值为1分

2次猜测就成功则每次猜测的贡献值为1/2分

3次猜测就成功则每次猜测的贡献值为1/3分

。。。

6次都失败则不得分，每次猜测的贡献值为0

~~所以统计所有猜测次数的贡献值的平均数，则得到难度得分函数~~

多元线性回归/BP神经网络

敏感度分析：对词频与信息熵产生高斯白噪音，重新跑多元回归分析，以拟合度指标建图

**以下为论文提纲：**

记号：（都用下标表示）

f\_word\_x 单词x的词频

syl\_x 单词x的音节数

ent \_x 单词x的猜测信息熵

f\_eletter\_x 单词x的简化字母组出现频率

f\_dletter\_x 单词x的繁化字母组出现频率

f\_repletter\_x 单词x的重复字母对出现频率

con\_x 单词x的单次猜测贡献值

* 模型总览：

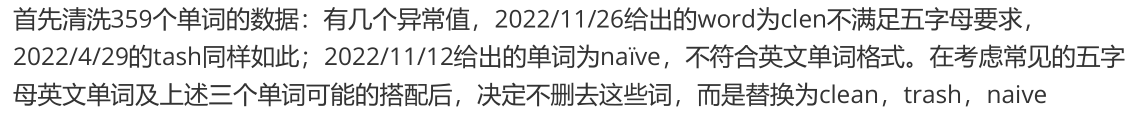
为了寻找会影响报告结果的比例的属性，我们首先需要对每个单词进行特征工程，即挖掘出单词特定属性的特征值，从而将单词映射为特有的特征向量。我们决定使用音节数、词频、猜测信息熵、简繁化字母组合出现频率、重复字母频率等特征。我们通过均摊假设计算出单次猜测的贡献值，依次作为单词猜测难度的评价标准，单词越难，单次猜测的贡献值越小。我们进行了这些因素对单次猜测贡献值的多元线性回归分析，确定了这些因素对单词综合难度的具体影响。之后我们使用BP神经网络模型，模拟**给定单词下**报告结果的相关百分比的分布。最后进行了敏感性分析。

* 第一部分 特征工程

【提供一张特征工程的流程图】

）预备：

首先我们进行了数据的重清洗，其中出现问题的数据以及其纠正如下（表）



2022/10/5 marxh单词未收录进柯林斯词典中

（<https://www.collinsdictionary.com/>）

词库也未收入，由于不确定真实单词，将其作为异常值，词频记为0

其次我们将单词猜测的难度量化，使用指标为单次猜测对猜测成功的贡献值。均摊假设即，假设在单人单日的一局游戏中，每一次猜测对于结果的贡献是相等的。也即下表：

1次猜测就成功则每次猜测的贡献值为1分

2次猜测就成功则每次猜测的贡献值为1/2分

3次猜测就成功则每次猜测的贡献值为1/3分

。。。

6次猜测就成功则每次猜测的贡献值为1/6分

6次都失败则不得分，每次猜测的贡献值为0

通过均摊假设，计算出依次单次猜测的贡献值，作为单词猜测难度的评价标准。单词越难，单次猜测的贡献值越小。

）选择理由

--考虑到单词在语言中的常见程度在猜测中的重要性，我们选择词频作为单词的特征属性之一。

--考虑到单词读音在猜测引导中的重要性，我们选择音节数作为单词的特征属性之一。

--考虑到字母频率会很大程度上，我们选择简繁化字母组出现频率、重复字母对出现频率作为单词的特征属性之一。

--考虑到字母位置与字母频率对猜测起到的综合影响，我们选择猜测信息熵作为单词的特征属性之一。

-综上，我们选择音节数、词频、猜测信息熵、简繁化字母组合出现频率、重复字母频率作为单词的特征属性

）数据挖掘

为了得到词库所有单词的常用度f\_word，我们使用c++进行统计。统计资源来自 based on www.opensubtitles.org for English and other languages These data are reused by various widely used opensource projects, among which Wikipedia, input methods and autocomplete keyoards, etc.

【附录】<https://github.com/hermitdave/FrequencyWords/tree/master>

为了得到词库所有单词的音节数syl，我们使用python外置库pronouncing进行数据的初筛。由于部分单词未收录入库，产生异常。此时剩下的数据由人工标记。

import pronouncing  
def get\_pronouncing\_num(word):  
    &apos;&apos;&apos;计算单词音节数&apos;&apos;&apos;  
    # https://pronouncing.readthedocs.io/en/latest/tutorial.html#counting-syllables  
    try:  
        pronunciation\_list = pronouncing.phones\_for\_word(word)  
        num = pronouncing.syllable\_count(pronunciation\_list[0])  
    except Exception as e:  
        print(&apos;计算音节数异常:异常单词:"&apos; + word + &apos;"&apos;)  
        #print(1)  
    else:  
        #print(&apos;单词{}的音节数为：{}&apos;.format(word,num))  
        print(num)  
  
llst = [  
    "manly","molar","havoc","impel","condo","judge","extra","poise","aorta","excel",  
    "lunar","third","slate","taper","chord","probe","rival","usual","spoke","apply",  
    "naive","knock","braid","infer","joust","amber","woken","adore","torso","chafe",  
    "undue","tepid","happy","clean","itchy","feast","drive","prime","axiom","brave",  
    "avert","glyph","there","baker","snarl","maple","inane","valet","medal","unite",  
    "rainy","spell","begin","stale","dream","photo","aloud","inept","piney","aptly",  
    "waltz","libel","sneak","carry","flout","foggy","fault","mummy","spiel","grove",  
    "denim","quirk","exist","stein","spade","catch","floor","equal","ionic","valid",  
    "enjoy","howdy","vigor","dandy","sloth","marxh","bough","sting","twine","leave",  
    "scorn","scald","usurp","soggy","brisk","admit","grate","glory","saint","recap",  
    "alike","trice","stick","chute","parer","doubt","thyme","alpha","booze","tibia",  
    "lofty","theme","class","leery","taunt","whoop","inter","gully","charm","fungi",  
    "prize","onset","chief","gauze","ruder","irony","clown","needy","woven","merit",  
    "waste","treat","shrug","twang","twice","gruel","poker","khaki","hunky","label",  
    "glean","cling","patty","unfit","smear","alien","buggy","rhyme","youth","coyly",  
    "quart","cramp","bluff","upset","stomp","motto","cinch","elope","power","midge",  
    "tryst","aphid","trite","angry","flock","wacky","roomy","wedge","liver","bland",  
    "night","madam","berth","stead","voice","agape","fluff","field","sever","lilac",  
    "egret","pinto","hutch","gawky","droll","retro","rusty","beady","smite","brink",  
    "awful","gloat","input","loser","cacao","blown","apron","primo","atone","donor",  
    "float","goose","piety","girth","trait","flood","gloom","depth","froth","peach",  
    "showy","creak","manor","atoll","bayou","crept","tiara","asset","vouch","album",  
    "hinge","money","scrap","gamer","glass","scour","being","delve","yield","metal",  
    "tipsy","slung","farce","gecko","shine","canny","midst","badge","homer","train",  
    "hairy","story","forgo","larva","trash","zesty","shown","heist","askew","inert",  
    "olive","plant","oxide","cargo","foyer","flair","ample","cheek","shame","mince",  
    "chunk","royal","squad","black","stair","scare","foray","comma","natal","shawl",  
    "fewer","trope","snout","lowly","stove","shall","found","nymph","epoxy","depot",  
    "chest","purge","slosh","their","renew","allow","saute","movie","cater","tease",  
    "smelt","focus","today","watch","lapse","month","sweet","hoard","cloth","brine",  
    "ahead","mourn","nasty","rupee","choke","chant","spill","vivid","bloke","trove",  
    "thorn","other","tacit","swill","dodge","shake","caulk","aroma","cynic","robin",  
    "ultra","ulcer","pause","humor","frame","elder","skill","aloft","pleat","shard",  
    "moist","those","light","wrung","could","perky","mount","whack","sugar","knoll",  
    "crimp","wince","prick","robot","point","proxy","shire","solar","panic","tangy",  
    "abbey","favor","drink","query","gorge","crank","slump"]  
for s in llst:  
    get\_pronouncing\_num(s)

可以作为附录

为了获得简繁化字母组频率、重复字母对出现频率，我们使用excel2022进行统计。

我们对于重复字母频率，采用“重复字母对”的统计，如

skill f\_repletter=1

vivid f\_repletter=2

mummy f\_repletter=3

如表所示为26个字母频率与单次猜测贡献值的相关系数co\_letter

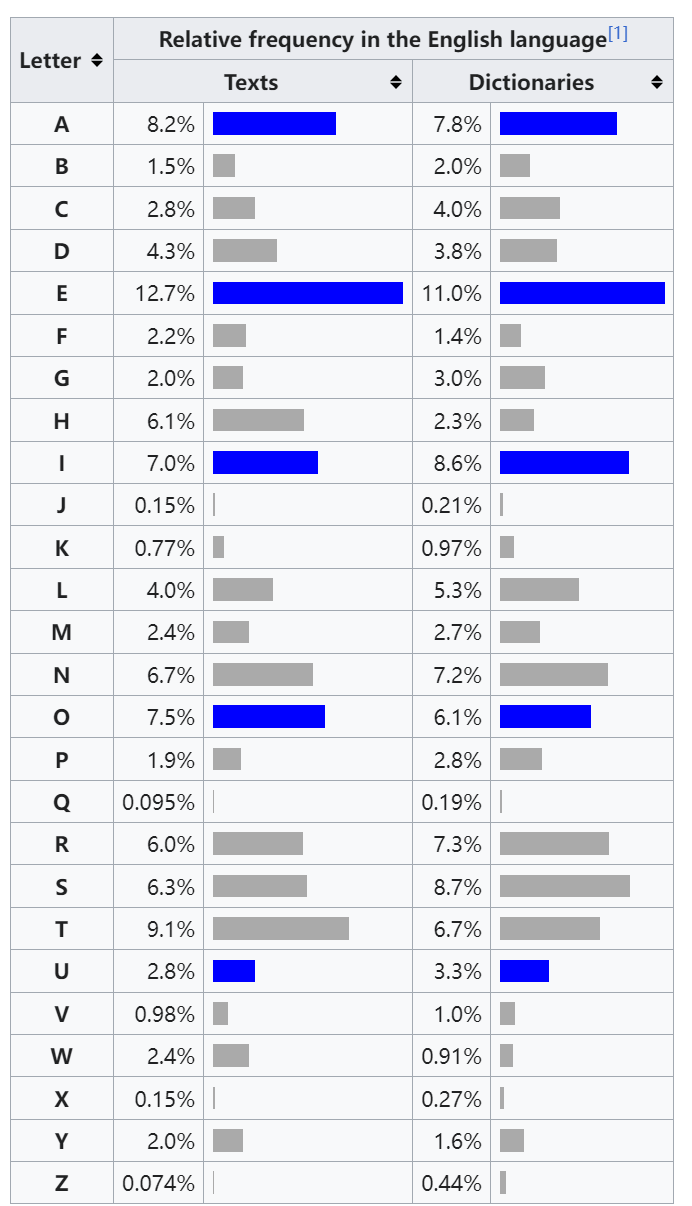
【表】

|  |  |
| --- | --- |
| a | 0.098631 |
| b | -0.03597 |
| c | -0.03554 |
| d | -0.00061 |
| e | -0.01012 |
| f | -0.0949 |
| g | -0.18798 |
| h | 0.08951 |
| i | 0.151684 |
| j | -0.10597 |
| k | -0.07327 |
| l | -0.06052 |
| m | -0.03051 |
| n | 0.097594 |
| o | -0.06458 |
| p | 0.078043 |
| q | -0.03442 |
| r | 0.048877 |
| s | 0.156533 |
| t | 0.282048 |
| u | -0.08956 |
| v | -0.12187 |
| w | -0.17932 |
| x | -0.07346 |
| y | -0.2231 |
| z | -0.16252 |

此外，字母本身的频率也会影响到组合的选取。我们参照数据来自

https://en.wikipedia.org/wiki/Letter\_frequency

1. Mička, Pavel. [*"Letter frequency (English)"*](http://en.algoritmy.net/article/40379/Letter-frequency-English). Algoritmy.net. [*Archived*](https://web.archive.org/web/20210304152631/http:/en.algoritmy.net/article/40379/Letter-frequency-English) from the original on 4 March 2021*. Retrieved 14 June 2022*. Source is Leland, Robert. Cryptological mathematics. [s.l.] : The Mathematical Association of America, 2000. 199 p. ISBN 0-88385-719-7



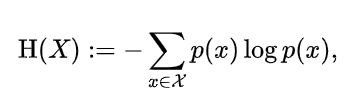
综合考虑到字母频率，组合频率，相关系数的因素，我们选择如下作为组合

w y g j z繁化字母组

aehiosrtn 简化字母组

首先引入猜测信息熵的定义

在[信息论](https://zh.wikipedia.org/wiki/%E4%BF%A1%E6%81%AF%E8%AE%BA" \o "信息论)中，**熵**（英语：entropy）是接收的每条消息中包含的信息的平均量。这里，“消息”代表来自分布或数据流中的事件、样本或特征。来自信源的另一个特征是样本的概率分布。这里的想法是，比较不可能发生的事情，当它发生了，会提供更多的[信息](https://zh.wikipedia.org/wiki/%E4%BF%A1%E6%81%AF" \o "信息)。事件的概率分布和每个事件的信息量构成了一个随机变量，这个随机变量的均值（即[期望](https://zh.wikipedia.org/wiki/%E6%95%B0%E5%AD%A6%E6%9C%9F%E6%9C%9B" \o "数学期望)）就是这个分布产生的信息量的平均值（即熵）。于是可以得到公式



熵的单位通常为比特。需要强调的是，熵具有可加性。

考虑到对于给定单词作为答案，我们以词频作为每次猜测的可能性，猜测信息熵被定义为每次猜测可获得的信息熵的加权平均数，其中权值为词频f\_ word \_x。

我们每次可获得的信息熵为原始存在的信息熵，与猜测后在游戏提示内容限定下的信息熵的差。记限定后所有可能的单词答案构成词库可以得到

我们继续假定此时作为玩家，我们的答案在所有可能的情况中**均匀分布**，以此得到

由于此时数据处理存在困难，我们进行一次近似操作：将游戏对整个单词的情况提示限定到字母。即

对于单个字母，我们计算

（i=1,2,3,4,5）

于是我们使用c++代码统计，伪代码如下：

名称：统计单词猜测信息熵

for every x in wordlist

for every word\_guess in vocabulary

根据规则生成题目提升

for every letter L in word

+=

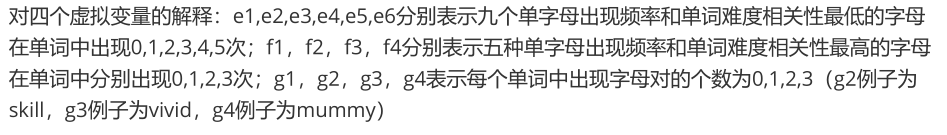
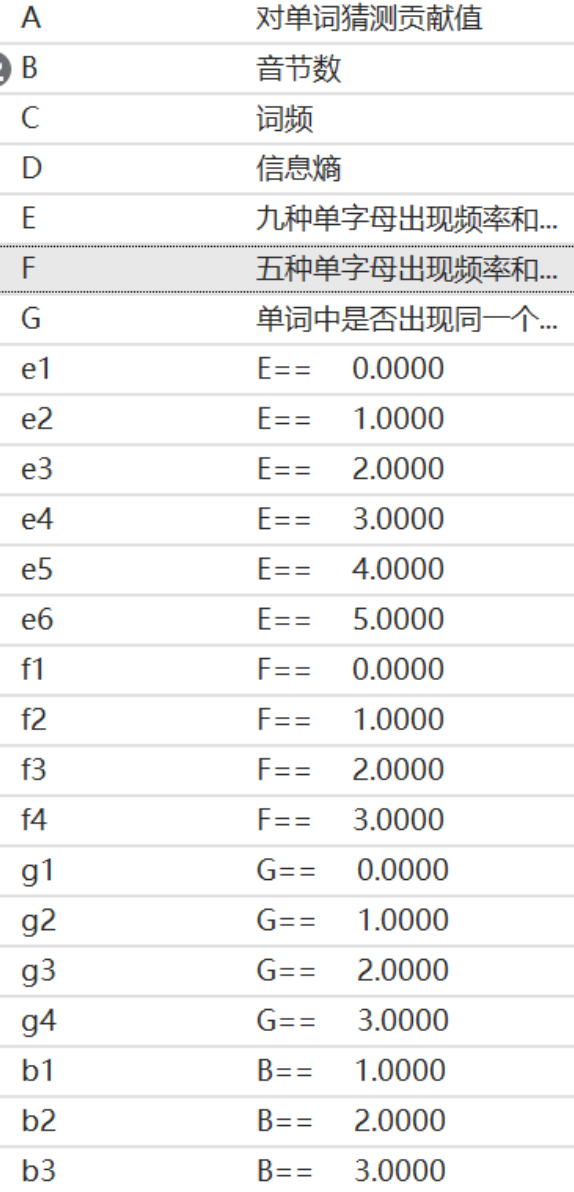
最后，每个字母的信息熵如图

【问cy要图】

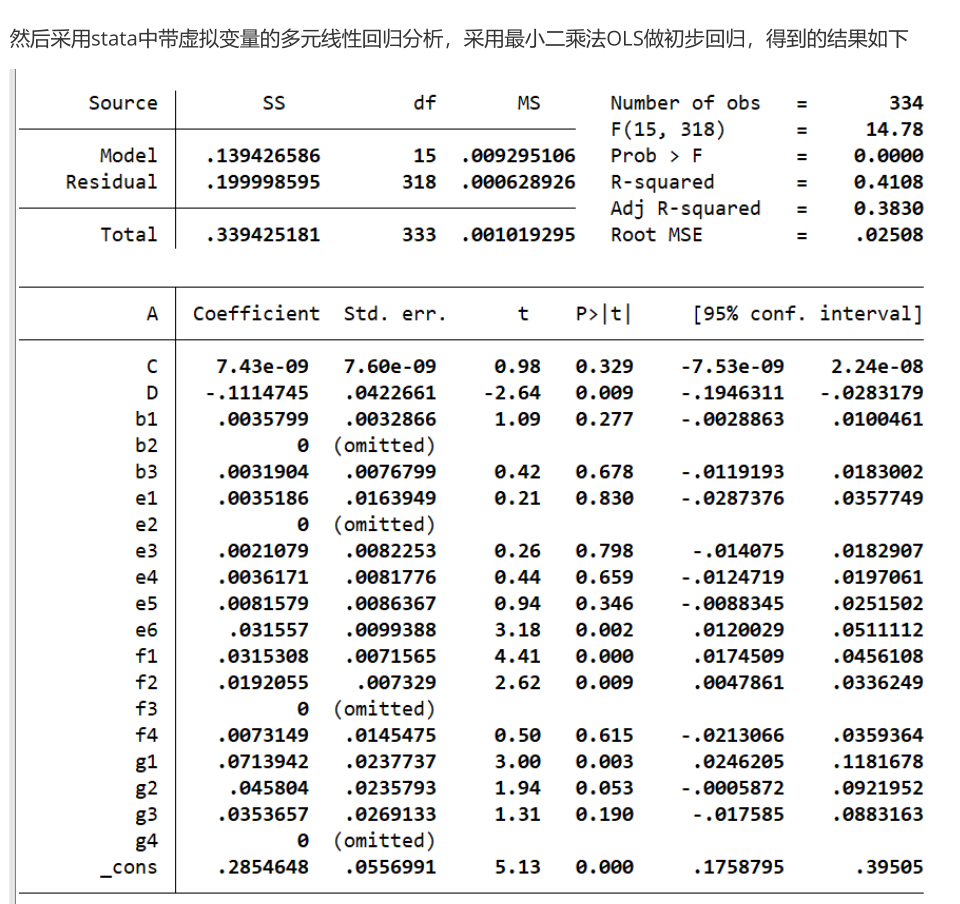
* 第二部分 多元回归分析

以单次猜测贡献值con\_x作为自变量，以词频f\_word\_x，音节数syl\_x，猜测信息熵ent\_x，简化字母组出现频率f\_eletter\_x，繁化字母组出现频率f\_dletter\_x，重复字母对出现频率f\_repletter\_x作为因变量，进行多元回归分析。对音节数、简化字母组出现频率、繁化字母组出现频率、重复字母对出现频率，这四个可以认为是离散的 定性变量，做虚拟变量的处理；考虑到验证模型的准确性这一问题，拟留出最后25个单词作为测试集，使用前334个作为训练集

系数符号：(A不用)



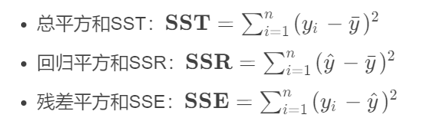
假设因变量和自变量之间存在线性关系——变量命名还没确定

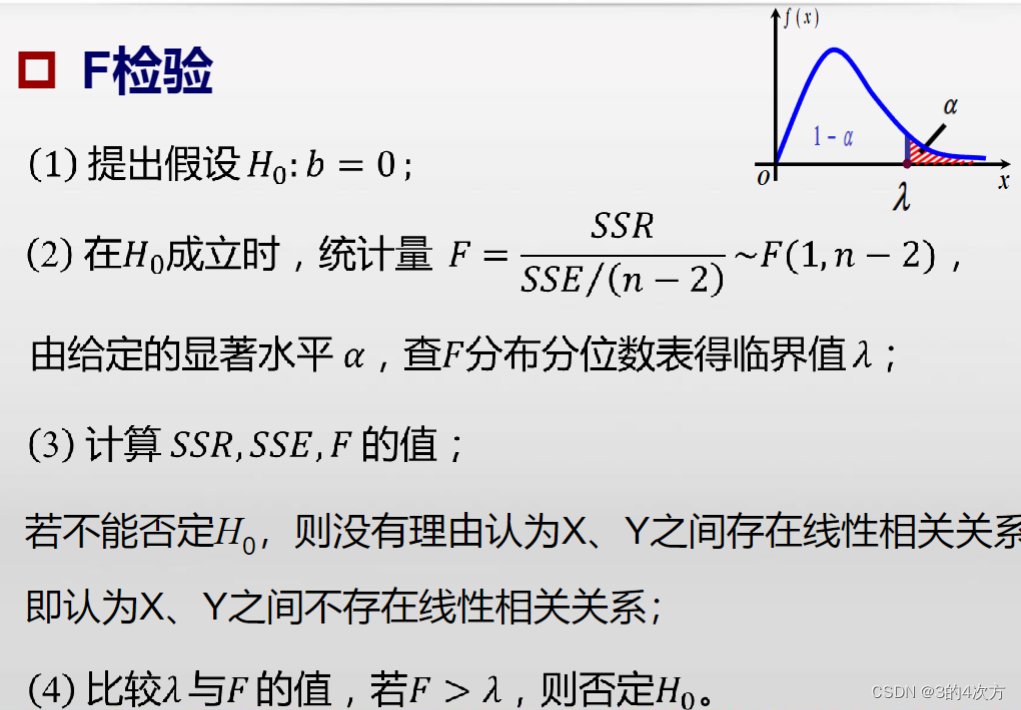


Number of obs 指样本观测值数量，共359条数据观测值；

SS列对应误差平方和，第一行回归平方和SSR，第二行剩余平方和SST；

df列指自由度；MS指均方差，第一行MSR回归均方差（其值是[回归](https://baike.baidu.com/item/%E5%9B%9E%E5%BD%92/0?fromModule=lemma_inlink)平方和除以回归自由度再开根。），第二行剩余均方差MSE（其值是剩余平方和除以回归自由度再开根。）



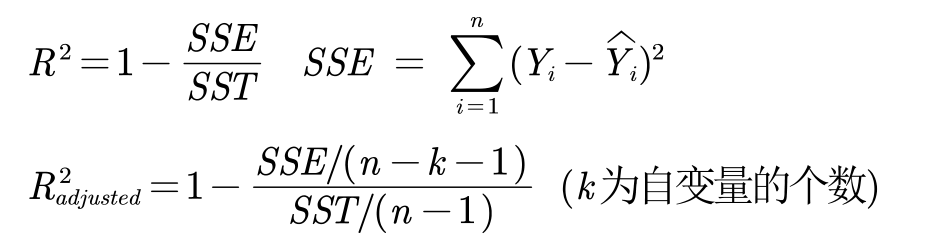


F(15,318)=MSR/MSE，显示的是F检验-方差检验结果，为模型的全局检验，来表明拟合方程是否有意义，其中，15是回归自由度，即回归模型中没有误差的个数；318是残差自由度，即回归模型中有误差的个数；

下面一条表示F检验的显著性，即F(15,318)在对应X值为14.78时的概率密度函数值，显然小于5%，因此具有显著的线性关系；

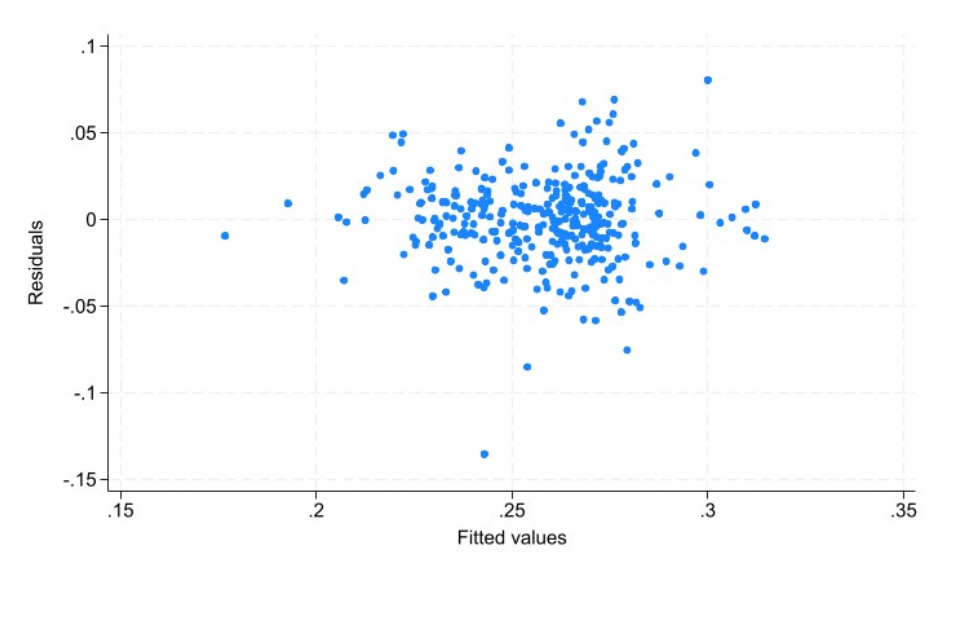
R^2=SSR/SST 为相关系数R的平方，值在0-1之间，表示模型的拟合优度，越大说明模型预测越准；

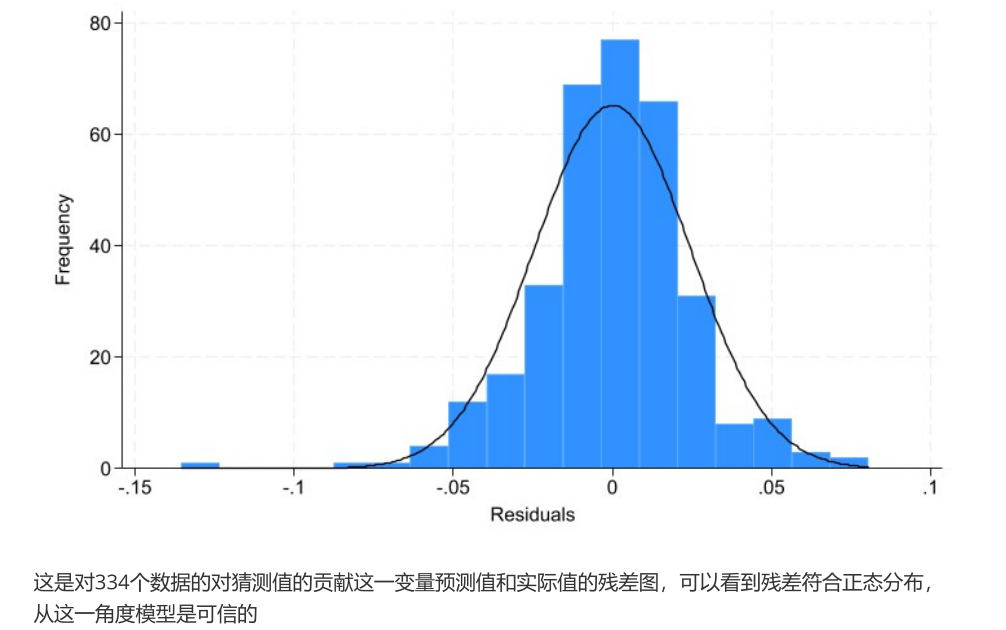
Adj R^2则表示的为调整后的拟合优度，同样越大表示模型预测越准确；表现这一多元线性回归分析有一定准确度；



P>|t|的值小于0.05表示在95%置信水平下回归系数显著异于零，Coefficient为该项在回归方程中对应的系数，Std.error表示该项的标准差

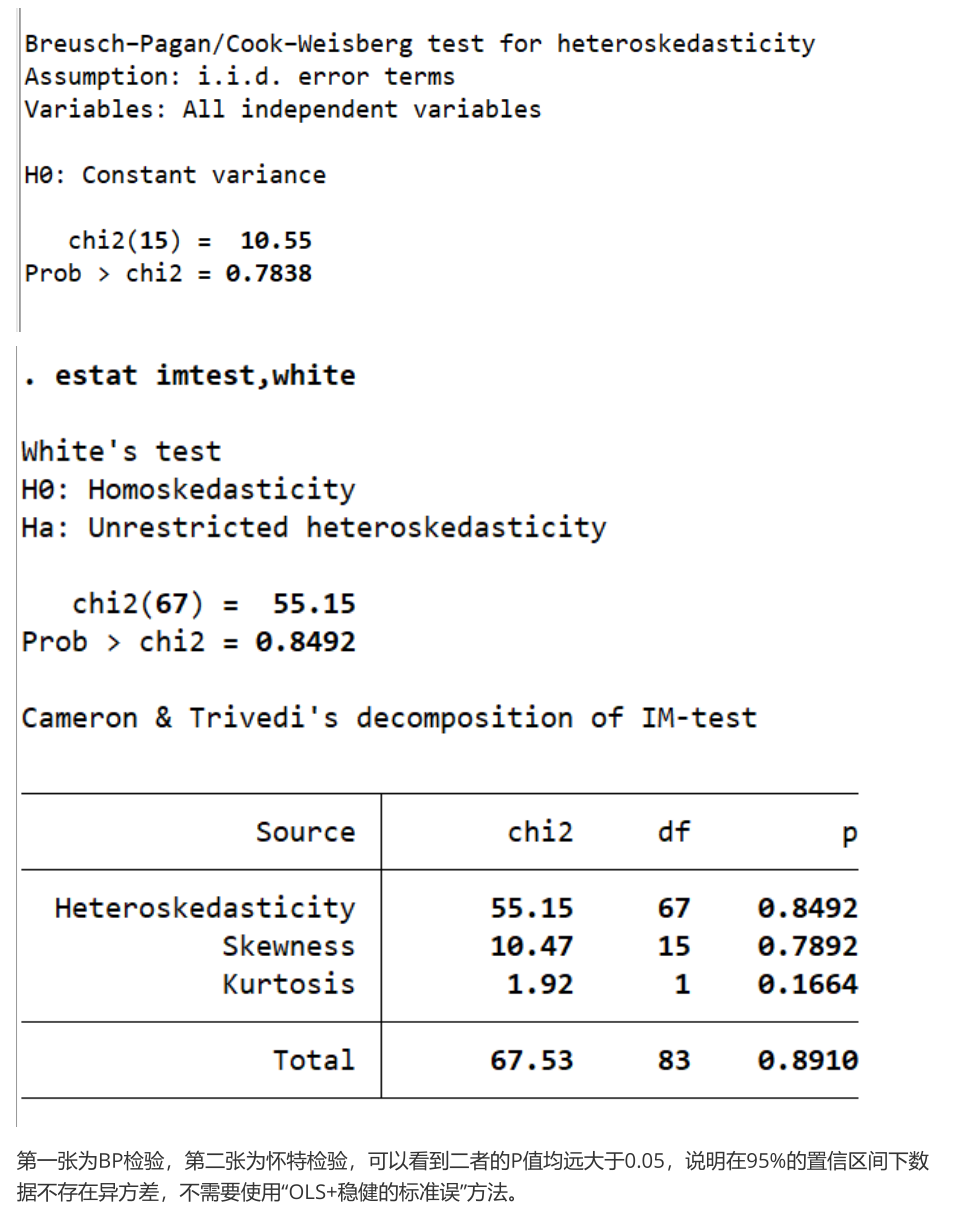
残差与拟合值的散点图，可以看到大多数数据的残差在可控范围之内，也符合正态分布

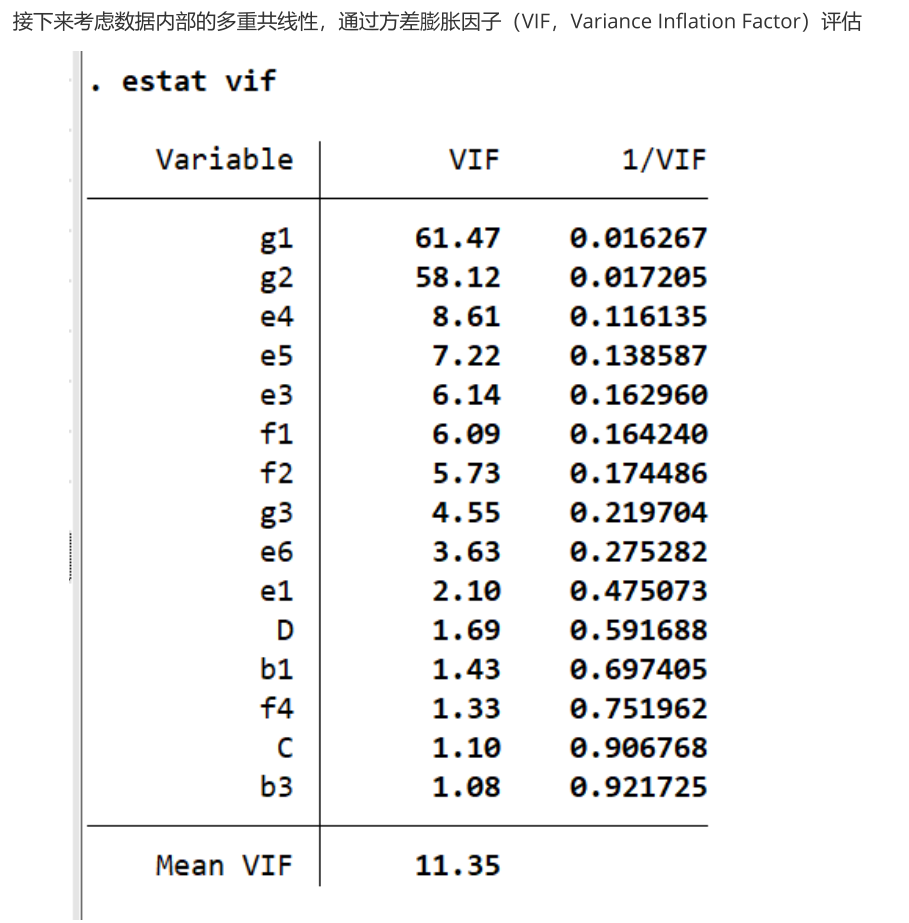




总之这一回归在数学上的误差“看起来”不算很大，似乎有可信度就是

接下来是异方差检验，采用BP检验和怀特检验，原假设为数据不存在异方差，BP检验和怀特检验的结果分别如下：





可以注意到除了g1和g2（一个单词中出现一个字母对和两个字母对）以外，其它元素都没有明显的多重

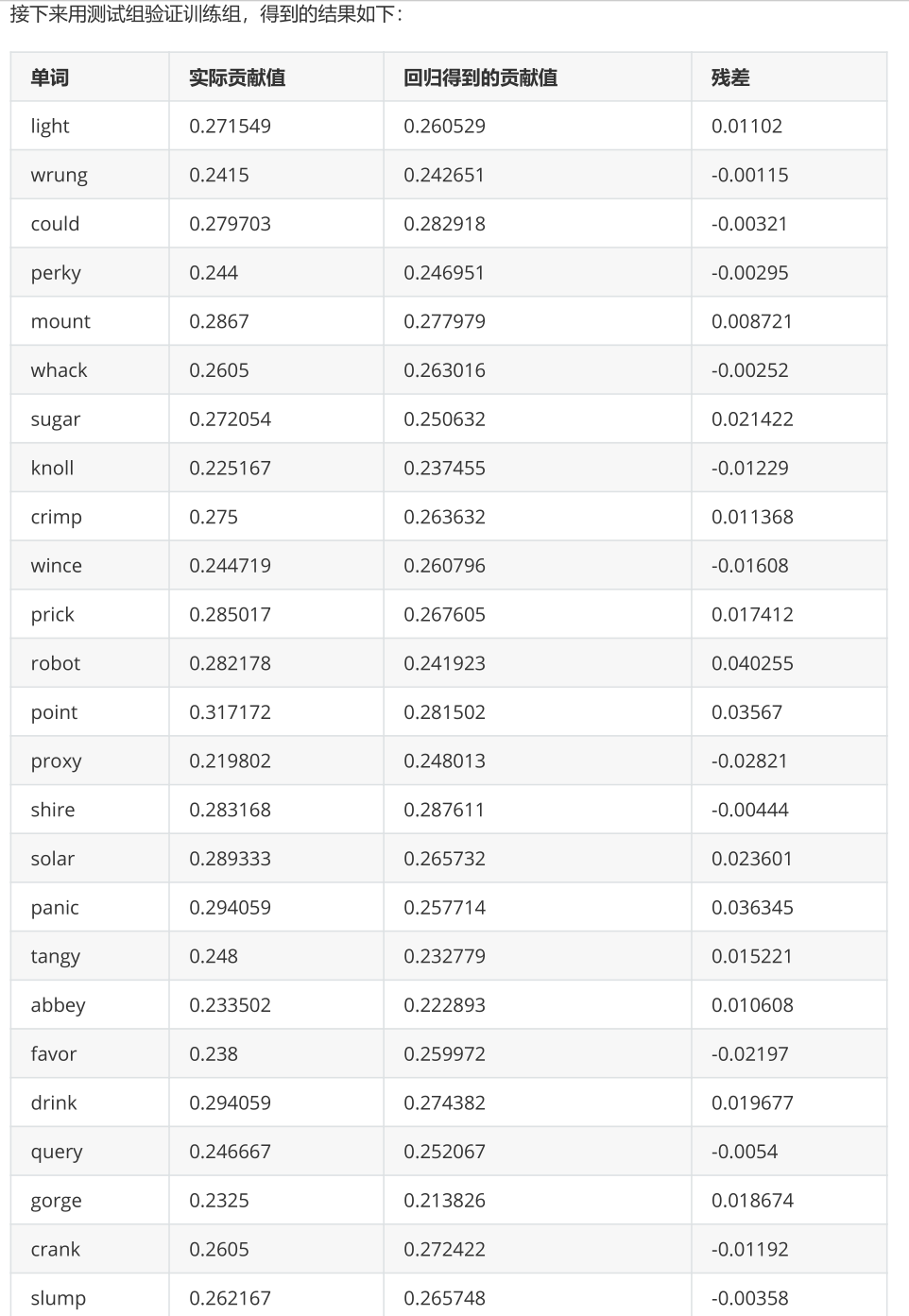
共线性，而且多重共线性对多个因素下模型本身的预测能力影响也不大；因此可以忽略多重共线性

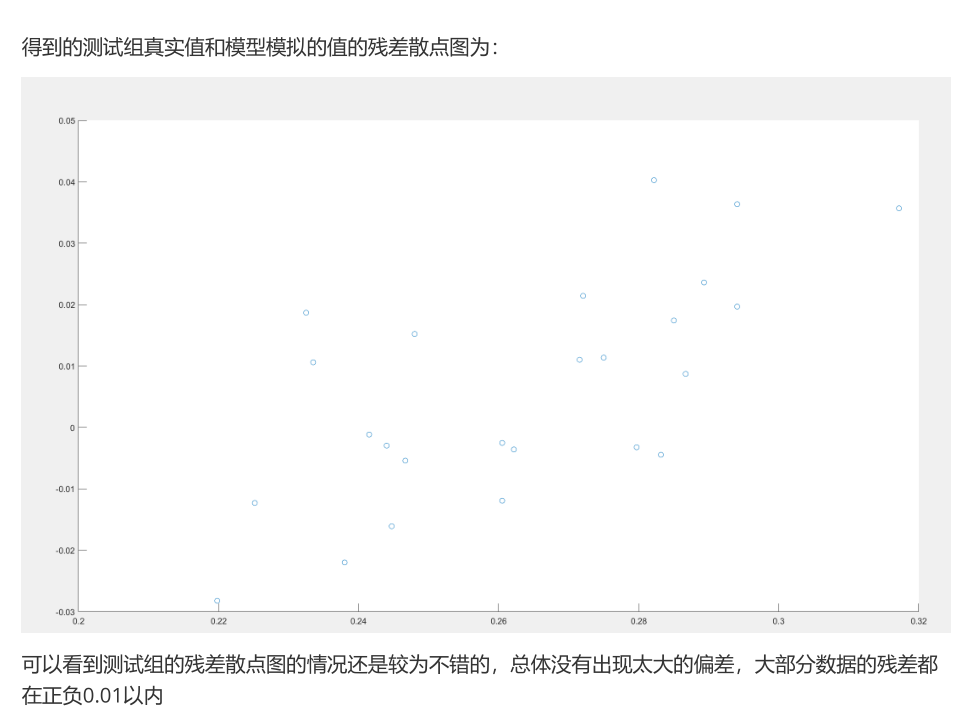
因此我们得到的多元线性回归方程为：

A（单词猜测的贡献值） = C×7.43e-09 + D×-0.1114745 + b1×0.0035799 + b3×0.0031904+

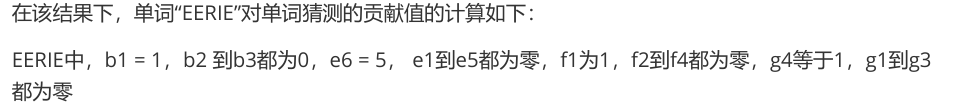
e1×0.0035186 + e3×0.0021709+ e4×0.0036171 + e5×0.0081579 + e6×0.031557 + f1×0.0315308 +

f2×0.0192055 + f4×0.0073149 + g1×.0713942+ g2×0.045804 + g3×0.0353657 + 0.2854648





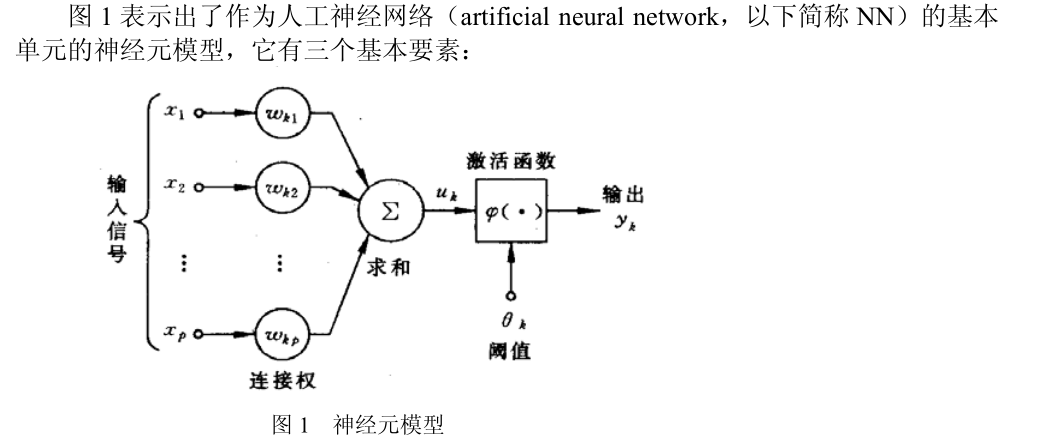
应用

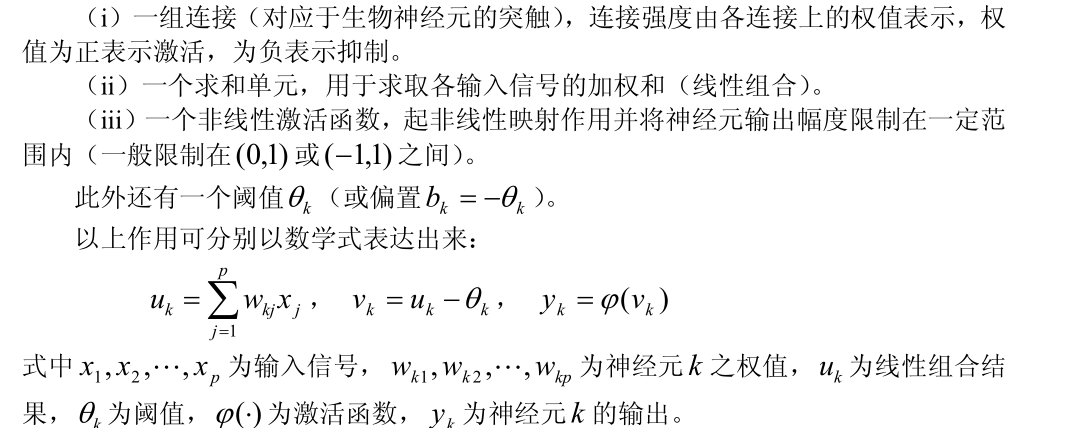


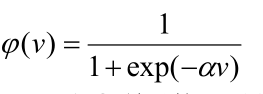
？值呢

* 第三部分 BP神经网络

由于第二问中要求我们预测的是EERIE一词的不同尝试次数的比例，因此需要预测的是一个有多个值的向量。因此我们考虑BP神经网络



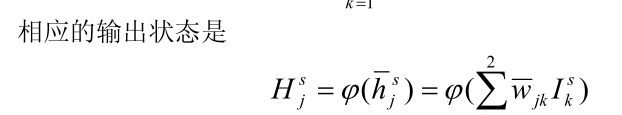


激活函数ϕ使用

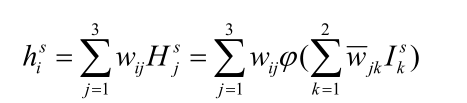
所有结点都是计算单元，同时也可接受输入，并向外界输出。

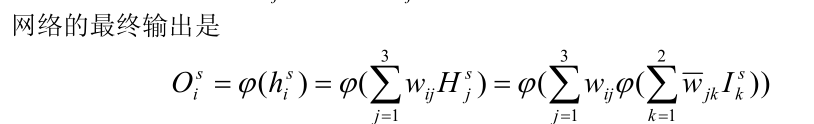
NN 的工作过程主要分为两个阶段：第一个阶段是学习期，此时各计算单元状态不变，各连线上的权值可通过学习来修改；第二阶段是工作期，此时各连接权固定，计算单元状态变化，以达到某种稳定状态。



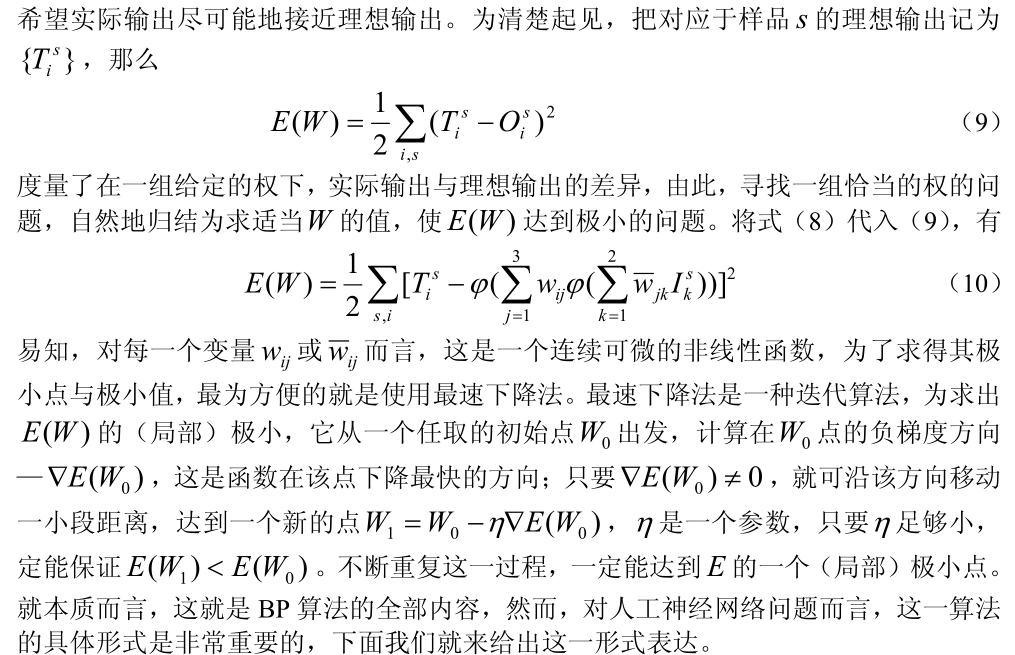


由此，输出单元 i 所接收到的迭加信号是



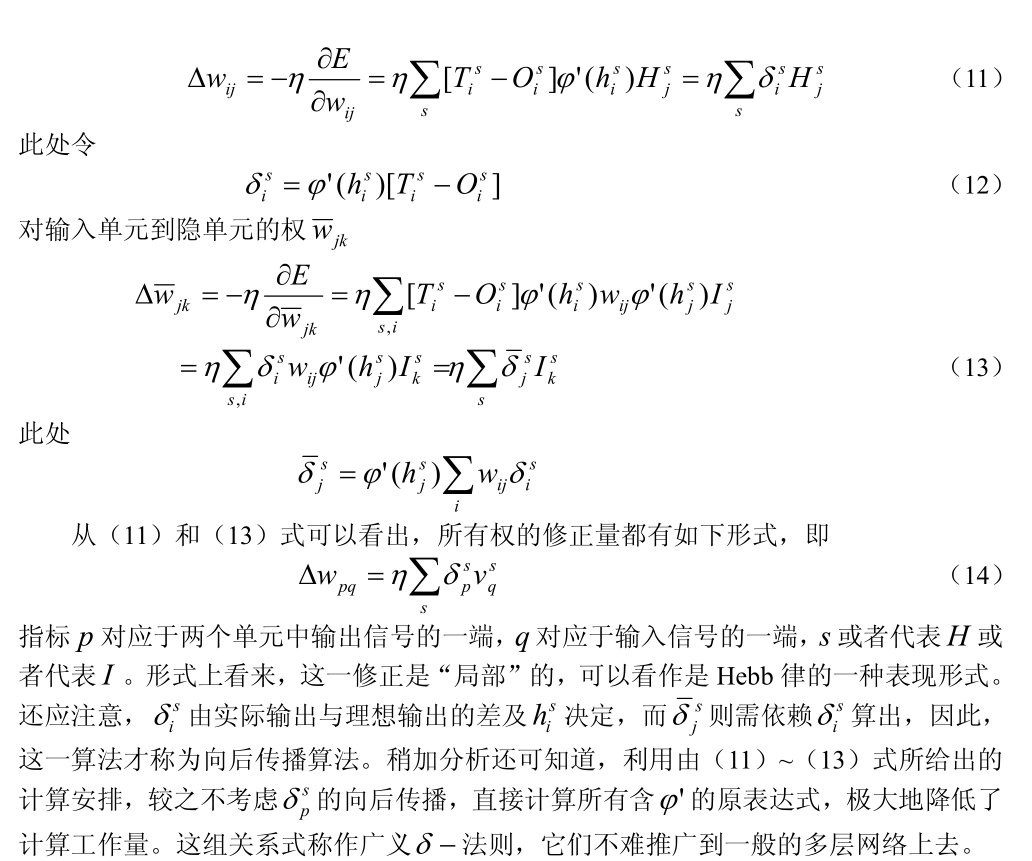


1985 年，美国加州大学的一个研究小组提出了所谓向后传播算法（Back-Propagation）



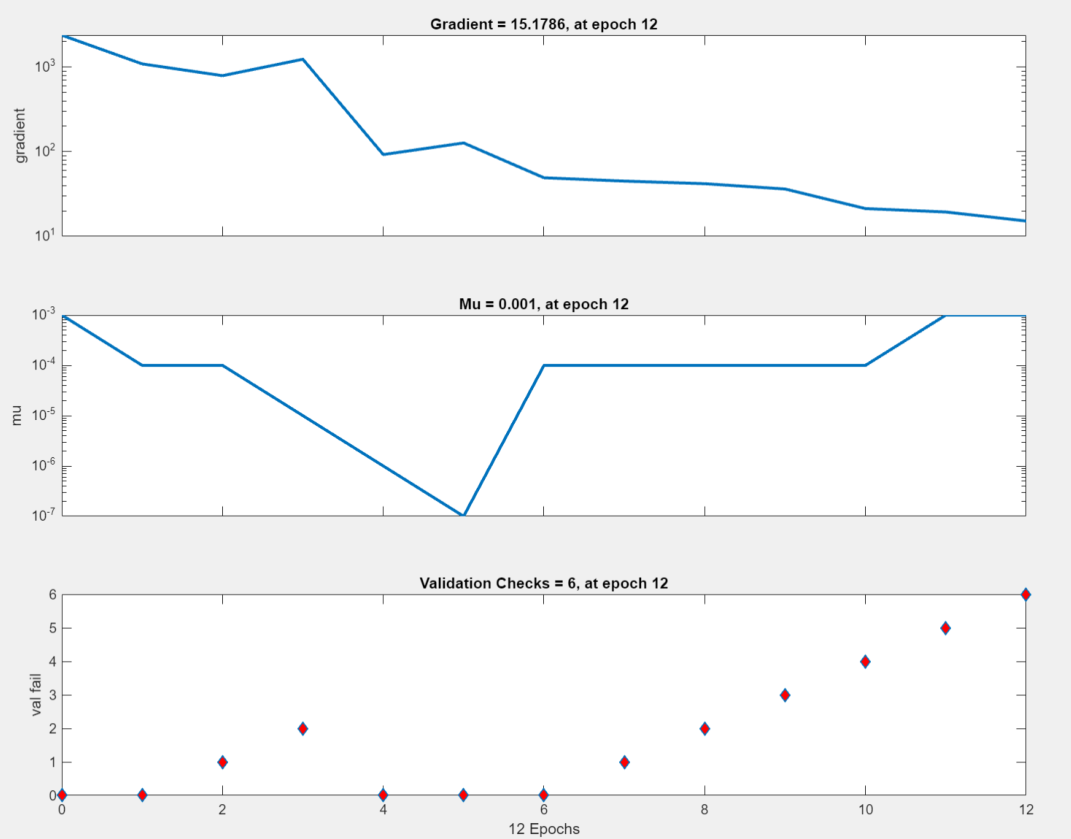
给出这一形式表达。





取1~6次成功的次数而舍去六次猜不出的数据

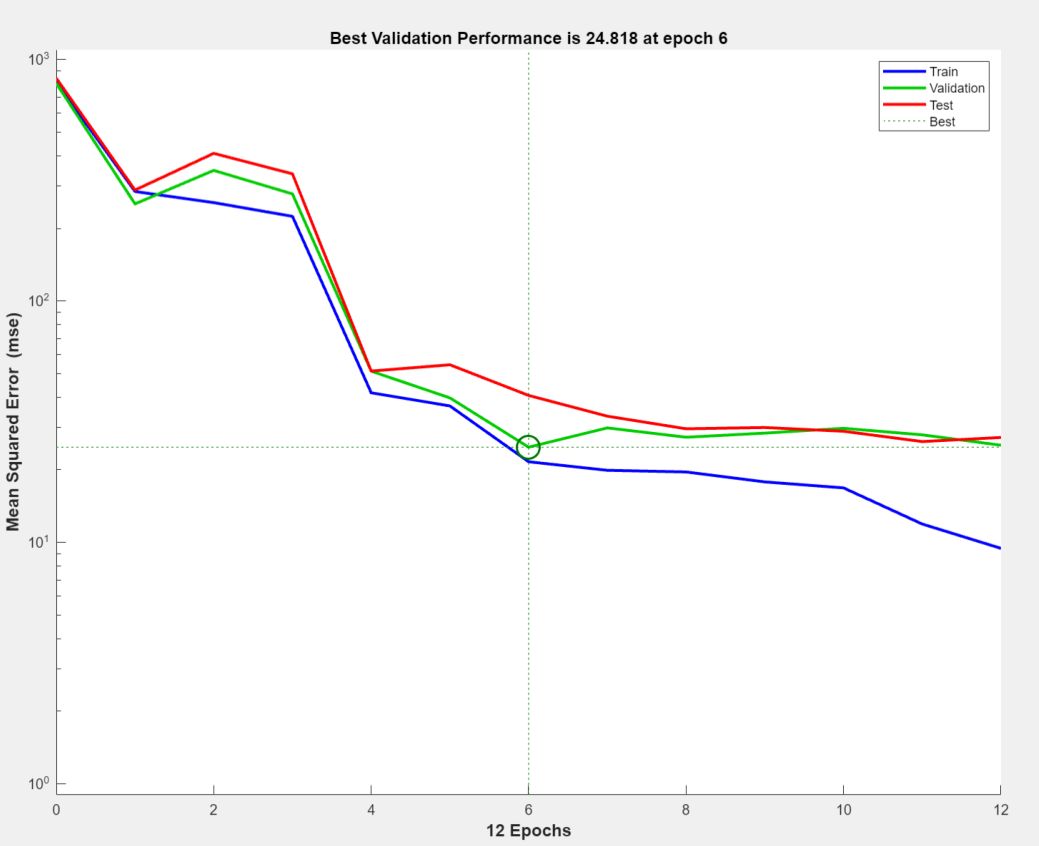
采用莱文贝格-马夸特算法，通过均方误差评价性能，选择70%数据作为训练集，15%数据作为验证集，15%数据作为测试集，共为15层



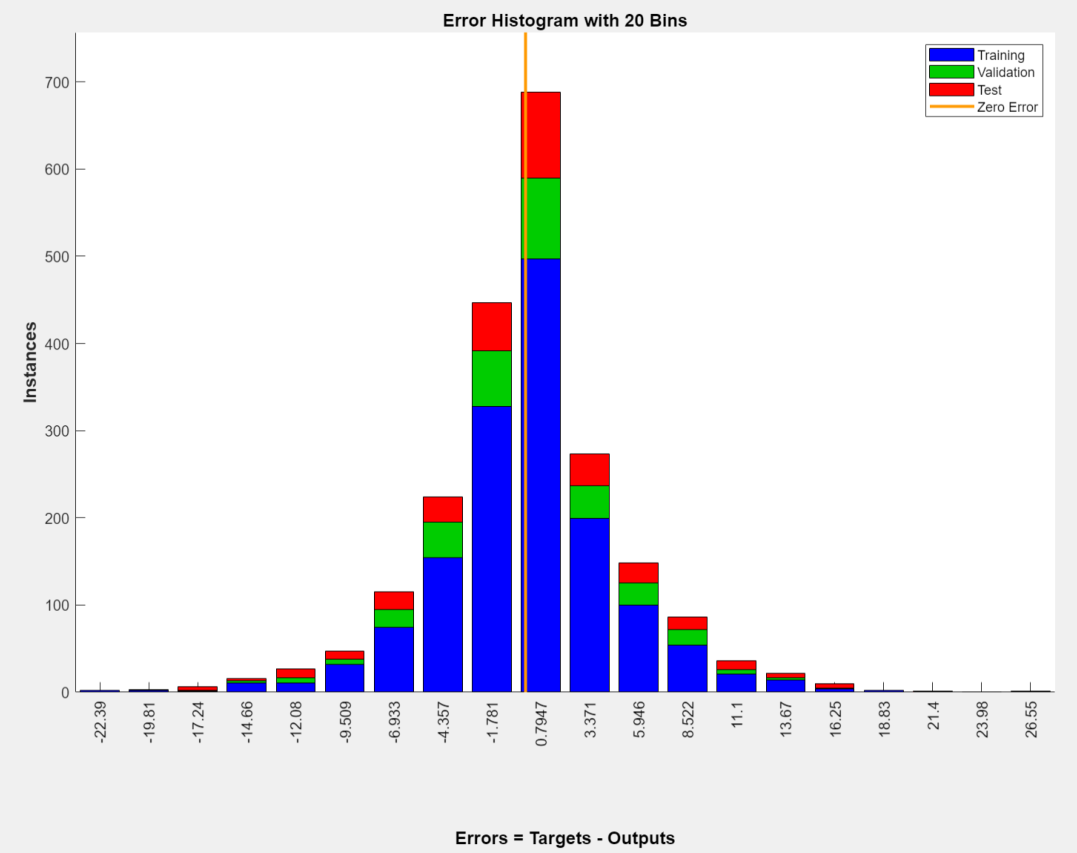
在第13轮达到了最小梯度？

？（中间图的解释）

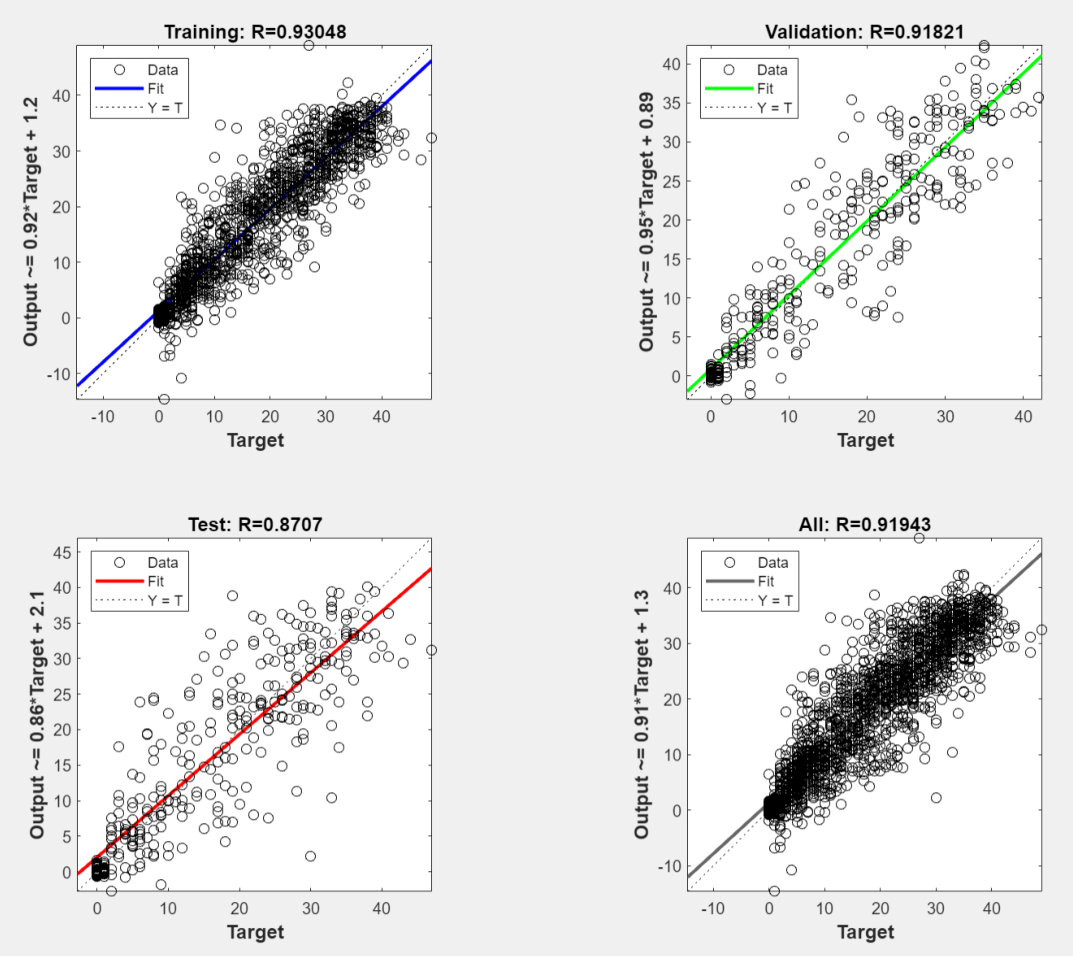
在训练6次时达到误差最小的训练效果



误差最好的训练表现出现在第6层，此时验证组的MSE（Mean Squared Error）最小



这是训练集、验证集和测试集的误差分布直方图



这是各个样本集和总体的相关性分析图

EERIE得到的结果如下：

-0.494446644820410 -2.84867271797616 12.5885952843330 38.4242111088678 33.7581894468274 15.1374753006053

近似得[0,0,12.5,38,33,15,2]

单次猜测贡献值0.226，残差在可接受范围

由此我们对BP神经网络模型与多元回归分析模型都有信心。

* 第四部分 敏感性分析

所谓高斯噪声是指它的概率密度函数服从高斯分布（即正态分布）的一类噪声。如果一个噪声，它的幅度分布服从高斯分布，而它的功率谱密度又是均匀分布的，则称它为高斯白噪声。高斯白噪声的二阶矩不相关，一阶矩为常数，是指先后信号在时间上的相关性。

对变量词频、信息熵分别乘以均值为1，标准差为0.004,0.006,0.008,0.01的高斯噪音，重新进行多元线性回归，得到其F检验值，R方与“eerie”预测难度结果，并汇出二维直方图

由此可见高斯噪音对模型的影响较为轻微，也即模型在一定限度下是稳定的。受到词频统计的轻微波动。

~~（单次猜测没波动）~~

/

# model 3

开发并总结一个模型，通过难度**分类解决方案单词**。**确定**与每个分类**相关**联的给定单词的**属性**。*使用您的模型，单词EERIE有多难？讨论您的分类模型的准确性。列出并描述该数据集的其他有趣特征。*

在模型2特征方程后的基础上聚类分析即可

需要提前降维（）

关键：K-Medoids聚类方法

针对本文中提到的单次猜测贡献，进行K-Medoids聚类。

介绍K-Medoids聚类：

k-medoid（也称为围绕medoid的划分）算法是由Kaufman和Rousseeuw于1987年提出的。k-medoids聚类是一种无监督的聚类算法，它对未标记数据中的对象进行聚类。k-medoids算法步骤:

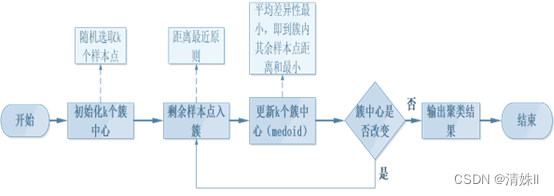
步骤1：随机选择k个样本作为初始的簇中心；

步骤2：对剩余的每个样本，将其划分入距离它最近的簇中心所在的簇中，从而形成k个簇；

步骤3：重新计算簇中心，即对于每个簇，找到该簇中的一个样本点，称为medoid，使得该簇中所有其他点到该样本点的距离之和最小;

k-medoids聚类中心一定是**原始样本点**，

步骤4：如果簇中心没有发生任何改变，算法停止，否则回到步骤2。

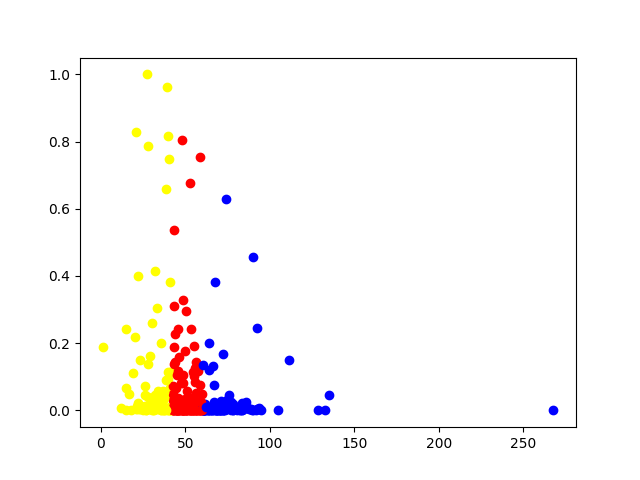


在这里，样本的距离被定义为：

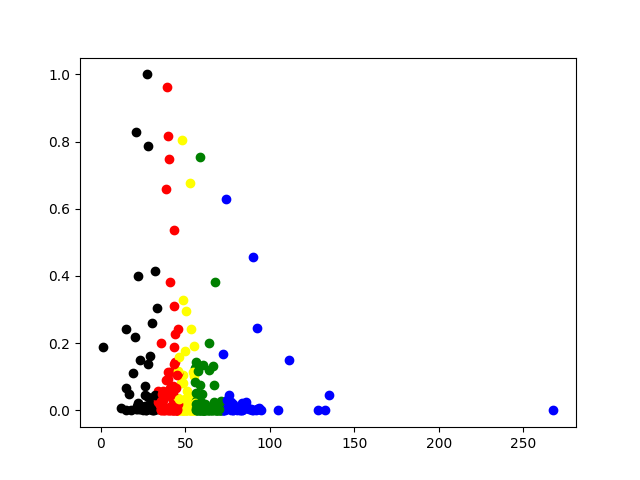
（公式）

由于K-medoid需要固定k，于是我们尝试了常用的k值进行K-medoid聚类，下面是结果

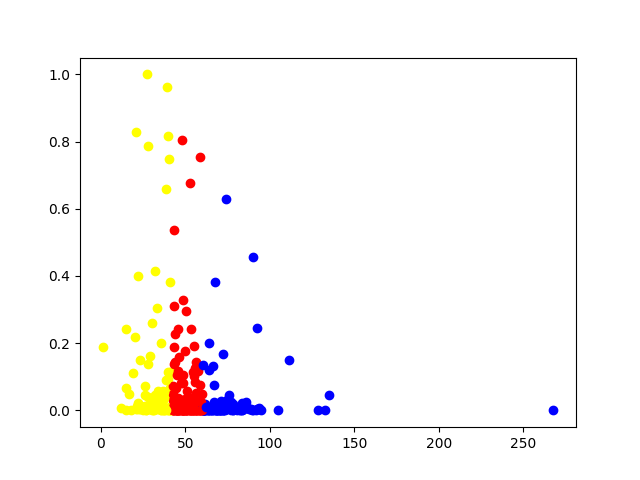
k=3



k=5



我们根据聚类结果，效果最好的是类别数为3时，聚类效果图如下



其中中心点为（等cy数据）

其中难度与属性如下

|  |  |  |  |
| --- | --- | --- | --- |
| 类别 | 预测难度 | 信息熵特征 | 罕见、重复对特征 |
| Easy case | 25 | 信息熵<1.08 | 无 |
| Normal case | 45 | 信息熵>=1.08 | 和为0或1 |
| Hard case | 70 | 信息熵>=1.08 | 和大于等于2 |

以此分类模型，对eerie进行分类，则知eerie应属于hard case，验证eerie的难度xx，的确属于Hard case。

*讨论您的分类模型的准确性。*

需要数据支撑（cy）

*列出并描述该数据集的其他有趣特征*

1 与第三位置的元音莫名其妙的相关

2孤立点

judge