**电视广告分户推送与营运规划**

**摘要**

本文利用相似度求解、线性回归模型、朴素贝叶斯和逻辑回归等方法建立视频广告和用户特征二者匹配模型、各时段底价估算模型和匹配动态推送模型，并且研究解决了提升收视率和买方产品销售量的问题。

针对问题一，我们用欧式距离和cos相似度方法建立了广告和用户分类匹配推送静态模型，分析收集的相关资料利用matlab软件绘制每个类型的观众人群对每种类型广告的观看记录条形图，再用欧式距离和cos相似度算法求解，距离小或者相似度大表明此类型用户与此类型广告近似匹配，从而得到电视频道用户和广告的特征匹配结果，结果详见表1.2。

针对问题二，我们收集了往季各时段卖方交易数据，然后借助线性回归和最小二乘法拟合出各时段相应的一元一次方程，拟合结果见1.1.1，在分析了往季月份和交易价格的关系后，结合官方数据验证了所建模型的准确性，最终采用线性回归预测的结果做为下一季度各时段竞买底价，预测结果详见表2.2。

针对问题三，为了能够实现动态更新视频广告与频道用户的分类匹配推送，我们先提取了广告的各类关键字，接着采用改进的朴素贝叶斯算法对用户观看广告的历史数据进行训练，结合在播视频广告的产品销售情况，建立了动态推送模型，其中为了使预测结果更具科学性，在数据统计时，为了防止某个分量在某个分类中未出现过从而导致结果为0的情况，们使用拉普拉斯平滑进行了处理，使得结果更科学性，最终实现了实时更新数据并向向观看频道用户进行推送的目的，结合表1.2，通过pycharm建立朴素贝叶斯多分类模型预测了结果，具体结果见表3.1。

针对问题四，想要建立一个合理的竞价交易模型，必须先确定一个用于评价的指标体系，我们根据题意选取了电视台收益，广告收视率以及产品销售量三个标准来评判各类广告是否投放，通过逻辑回归算法构造预测函数以及模型参数估计建立了逻辑斯谛回归模型，结合收集的相关数据以及利用matlab建立百分比饼状图了解到每个时段主要观看广告人群类型，饼状图见4.1，结合问题一的匹配结果，确定了时段推送的广告类型，通过逻辑斯谛回归模型来判断广告是否可以投放，接着根据投放/产出来确定广告是否在该时段主要投放来达到此问的要求。

针对问题五，我们通过对前四问题意分析以及获取的数据，决定第一问使用相似度算法，第二问使用线性回归算法，第三问使用朴素贝叶斯多分类算法，第四问使用逻辑回归算法。基本优化算法则使用梯度下降算法。相应算法见附录3，4，5，6。

**关键字：**贝叶斯 相似度 逻辑斯谛回归模型 线性回归 拉普拉斯平滑

1. **问题重述**

电视广告是地方电视台商业营运的主要业务之一。广告视频由产品销售或制造商负责制作并向电视台竞买播放时段，电视台对每个广告播放时段分周期组织竞卖，通过建立数学模型对各时段广告底价及观看用户类型做好规划，可以更好的完成下一周期的竞价交易，保证在提高电视台的收视率的同时增加产品销售量。我们还需要解决以下几个问题：

1、通过选择/提取视频广告和电视频道用户的分类特征建立二者分类匹配推送静态模型；

2、设计卖方分时段竞卖合理底价估算模型；

3、基于已知电视频道用户收视历史情况和在播视频广告的产品销售情况，建立视频广告与频道用户的分类匹配推送更新模型；

4、基于更新的分户推送方式建立竞价交易模型，在极大化卖方收益的同时提升收视率和买方产品销售量；

5、设计并实现你建立的模型的求解算法，通过收集或编撰数据给出算例。

1. **问题的分析**

电视广告分户推送与营运规划问题我们的理解就是根据用户的偏好推荐符合用户口味的广告，以此来提高收视率以及销售量。

**问题一的分析：**

问题一要求通过选择/提取视频广告和电视频道用户的分类特征建立二者分类匹配推送静态模型。由于此题没有数据，我们通过上网参考相关资料根据自己编撰的数据，对用户特征和广告特征进行欧式距离以及cos相似度的求解，距离小或者相似度大则表明此类用户对这种广告的偏好值大。以此来解决这个问题。

**问题二的分析：**

问题二要求设计卖方分时段竞卖合理底价估算模型。我们通过对往季广告交易数据的分析，采用线性回归模型对下一季度广告底价进行预测，以此来解决此问题。

**问题三的分析：**

问题三基于已知电视频道用户收视历史情况和在播视频广告的产品销售情况，建立视频广告与频道用户的分类匹配推送更新模型。我们通过中国广告网以及CSM获取了部分用户观看广告的历史情况以及电视台近来在播的广告。分析出此问是多分类问题，因此我们采用改进的朴素贝叶斯算法对用户观看广告的历史数据进行训练，根据训练结果对在播的广告进行分类，每当广告到达下个周期，都会扩大训练数据重新进行训练，以此建立动态推送模型。这就是我们这一问的主要思路。

**问题四的分析：**

问题四基于更新的分户推送方式建立竞价交易模型，在极大化卖方收益的同时提升收视率和买方产品销售量。我们通过对问题三结果以及此问的分析，收集了若干条广告在不同时段依据电视台收益，收视率和广告产品销售情况三个指标的分析，手动给予了是否投放广告的标签，然后根据投放/产出来确定广告是否在该时段投放来达到此问的要求。

**问题五的分析：**

我们通过对前四问的分析以及获取的数据，决定第一问使用相似度算法，第二问使用线性回归算法，第三问使用朴素贝叶斯多分类算法，第四问使用逻辑回归算法。基本优化算法则使用梯度下降算法。

1. **模型假设**
2. 假设收集的数据和实际数据相差不大。
3. 假设所有广告播放时长都是5秒。
4. 假设每天的时间段都是确定的。
5. 假设每个地区的用户对广告的偏好一致。
6. **符号及变量说明**

x 广告特征

y 用户特征

d(x,y) x和y的欧式距离

d cos相似度

Yi1 第i个时间段的成交价

Xi1 第i个时间段的月份

Yi2 第i个时间段的成交价

Xi2 第i个时间段的月份

T 训练数据

ajl 第j个特征的第l个可能的取值

arg maxP 最大概率

g Sigmoid函数

h 预测函数

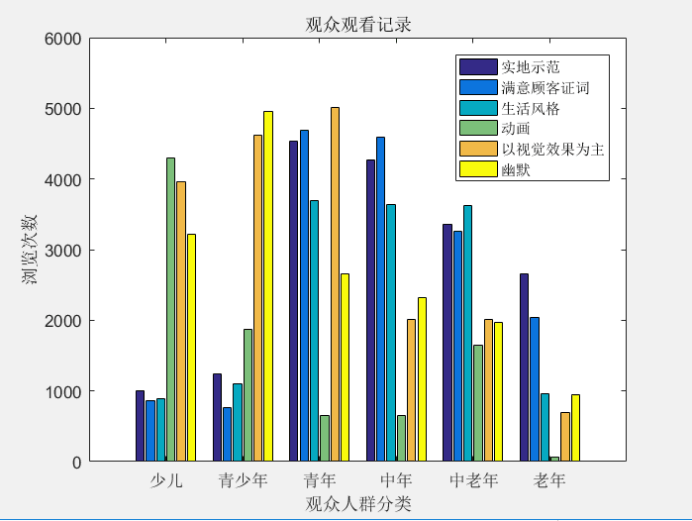
L 对数似然函数

c 广告的投入产出比

1. **模型的建立与求解**

**5.1 通过选择/提取视频广告和电视频道用户的分类特征建立二者分类匹配推送静态模型**

我们对用户在一天内对不同广告的浏览量做了如下统计：



由上图可知，对于不同观看广告的人群，其对广告的偏好也不尽相同，少儿侧重于动画类的广告，青少年侧重于幽默类型的广告，青年比较喜欢以视觉效果为主的广告类型，中年喜欢满意顾客证词类型的广告，而中老年则比较喜欢生活风格的广告类型，老年则喜欢实地示范的广告类型。综上所述，不同类型的人群都有其喜欢的广告类型，进行广告推送时，可以根据在什么时段什么类型的广告观看人群多，推送相应喜欢的广告类型，可以保证电视台收视率以及商品销售商实现共赢。

为了建立广告和用户分类匹配推送静态模型，我们将广告大致分为6类，根据用户年龄将用户分为6类。如图所示：

|  |  |
| --- | --- |
| 用户类型 | 用户偏好特征 |
| 少儿 | 易懂 形象 幼稚 童趣 |
| 青少年 | 搞笑 风趣 形象 |
| 青年 | 酷炫 感染力 想象 搞笑 |
| 中年 | 可信度高 反馈 顾客证词 功能 消费 |
| 中老年 | 生活习惯 风度 气质 可信度高 |
| 老年 | 演示 消费 可信度高 功能 |

|  |  |
| --- | --- |
| 广告类型 | 广告特征 |
| 实地示范 | 演示 消费 可信度高 功能 |
| 满意顾客证词 | 可信度高 反馈 顾客证词 功能 |
| 生活风格 | 生活习惯 风度 气质 |
| 动画 | 易懂 形象 幼稚 童趣 |
| 以视觉效果为主 | 炫酷 感染力 想象 |
| 幽默 | 搞笑 风趣 形象 |

我们根据提取的广告特征和用户的偏好特征通过计算它们之间的欧式距离和cos相似度构建静态模型。

我们将所有广告特征用set集合进行合并构建不重复词表，用数字1表示此表中存在此特征，0则表示不存在。部分数据如图所示：

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 种类\特征 | 风趣 | 反馈 | 形象 | 风度 | 可信度高 | 搞笑 | 功能 | 童趣 | 炫酷 |
| 实地示范 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |
| 满意顾客证词 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 |
| 生活风格 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 少儿 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 青少年 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 青年 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |

表1.1

**5.1.1欧式距离**

根据表1.1我们设用户偏好特征在词表中的存在情况为,广告特征在词表中的存在情况为y = (),它们之间的欧式距离为：

因此构建出的欧式距离矩阵为：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 实地示范 | 满意顾客证词 | 生活风格 | 动画 | 以视觉效果为主 | 幽默 |
| 少儿 | 2.828427 | 2.82842712 | 2.6457513 | 0 | 2.64575131 | 2.23606 |
| 青少年 | 2.645751 | 2.64575131 | 2.4494897 | 2.236 | 2.44948974 | 0 |
| 青年 | 2.828427 | 2.82842712 | 2.6457513 | 2.828 | 1 | 2.23606 |
| 中年 | 1.732050 | 1 | 2.8284271 | 3 | 2.82842712 | 2.82842 |
| 中老年 | 2.449489 | 2.44948974 | 1 | 2.828 | 2.64575131 | 2.64575 |
| 老年 | 0 | 2 | 2.6457513 | 2.828 | 2.64575131 | 2.64575 |

欧式距离越小则表明用户对此类广告越是偏爱。

**5.1.2cos相似度**

cos相似度对于x和y，余弦距离为：

cos相似度矩阵为：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 实地示范 | 满意顾客证词 | 生活风格 | 动画 | 以视觉效果为主 | 幽默 |
| 少儿 | 0 | 0 | 0 | 1 | 0 | 0.2886 |
| 青少年 | 0 | 0 | 0 | 0.2886 | 0 | 1 |
| 青年 | 0 | 0 | 0 | 0 | 0.8660254 | 0.2886 |
| 中年 | 0.67082039 | 0.89442719 | 0 | 0 | 0 | 0 |
| 中老年 | 0.25 | 0.25 | 0.866025 | 0 | 0 | 0 |
| 老年 | 1 | 0.5 | 0 | 0 | 0 | 0 |

cos相似度越大则表明用户对此类广告越偏爱。

**5.1.3模型求解**

详细代码见附录3

根据统计结果以及欧式距离和cos相似度的求解，我们得出结论：

|  |  |
| --- | --- |
| 用户种类 | 广告种类 |
| 少儿 | 动画 |
| 青少年 | 幽默 |
| 青年 | 以视觉效果为主 |
| 中年 | 满意顾客证词 |
| 中老年 | 生活风格 |
| 老年 | 实地示范 |

表1.2

**5.2 设计卖方分时段竞卖合理底价估算模型**

为了了解广告在往期不同时段成交价和底价的变化情况，本文参考山西电视台往期广告成交记录。记录情况如下表所示：

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 月份 |  | 06：00-17：54 | 17：54-18：26 | 18：26-18：57 | 18：57-20：25 | 20：25-22：00 | 22：00-22：20 | 22：20-23：14 | 23：14-01：20 | 01：20-06：00 |
| 一月 | 成交价 | 26000 | 28000 | 32000 | 36000 | 32000 | 22000 | 18000 | 9600 | 7600 |
| 底价 | 17000 | 20000 | 21000 | 22000 | 22000 | 15000 | 15000 | 9000 | 7200 |
| 二月 | 成交价 | 26000 | 27000 | 32000 | 35000 | 32000 | 21000 | 17000 | 9600 | 7600 |
| 底价 | 16800 | 19700 | 20600 | 21000 | 21000 | 15000 | 14000 | 8800 | 7100 |
| 三月 | 成交价 | 25000 | 27000 | 32000 | 34000 | 31500 | 21000 | 16000 | 9500 | 7550 |
| 底价 | 16500 | 19500 | 20500 | 20500 | 20500 | 14500 | 13800 | 8700 | 7050 |
| 四月 | 成交价 | 25000 | 26000 | 31000 | 32000 | 31000 | 20500 | 16000 | 9400 | 7500 |
| 底价 | 16000 | 19200 | 20500 | 20300 | 20500 | 14000 | 13500 | 8600 | 7010 |
| 五月 | 成交价 | 22000 | 23000 | 27000 | 30000 | 30000 | 19000 | 14000 | 9300 | 7200 |
| 底价 | 16000 | 19000 | 20200 | 20000 | 20000 | 13500 | 13000 | 8500 | 7000 |
| 六月 | 成交价 | 23000 | 24000 | 27000 | 31000 | 30000 | 19500 | 14500 | 9000 | 7200 |
| 底价 | 14000 | 19000 | 20000 | 19800 | 20000 | 13200 | 13000 | 8200 | 6800 |

表2.1

根据表2.1，我们用matlab做出以下三维图形：

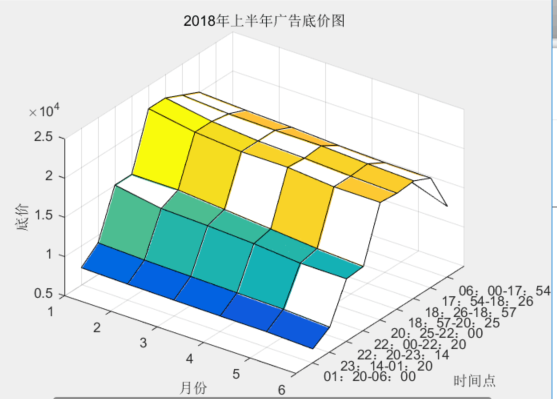
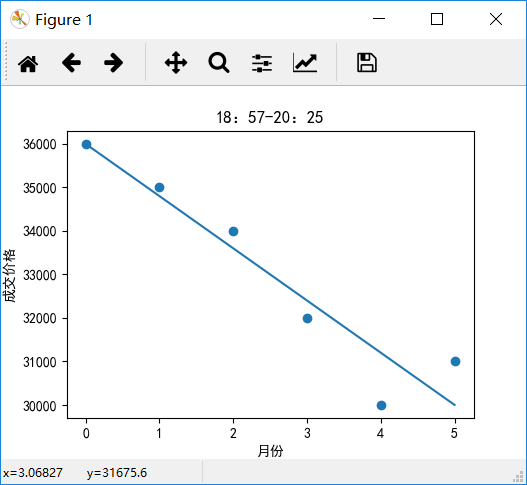
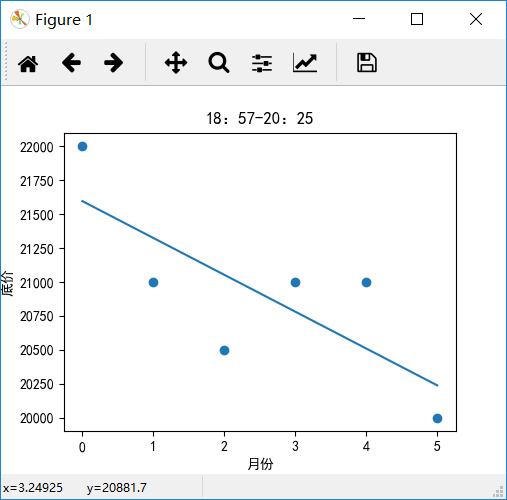


图2.1

如图2.1，我们可以很直观的发现无论是哪个时间段，广告底价和成交价随着月份的增大整体成下降趋势，因此我们根据线性回归和最小二乘法模拟不同月份各个时间段的成交价与底价变化情况，通过对其拟合方程的合理性进行分析，可以预测出未来三个月的底价和成交价的情况。

利用表2.1给出不同时段的成交价和底价的数据，拟合成一个一元一次方程。其中一个时间段的底价和成交价拟合图像分别为：





设：  
；

拟合结果为：

06：00-17：54：



17：54-18：26：



18：26-18：57：



18：57-20：25：



20：25-22：00：



22：00-22：20：



22：20-23：14：



23：14-01：20：



01：20-06：00：



1.1.1

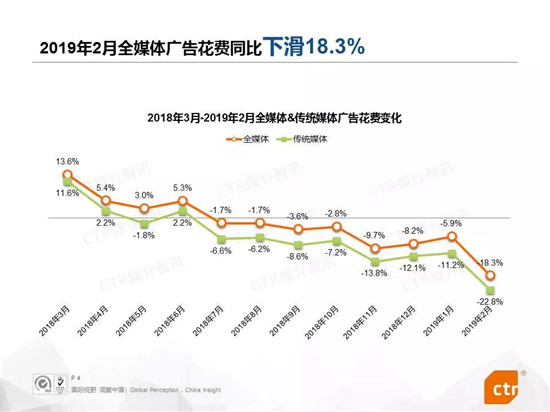
详细代码见附录4

通过以上拟合结果预测的接下来三个月的交易价格情况如下表所示：

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 月份 |  | 06：00-17：54 | 17：54-18：26 | 18：26-18：57 | 18：57-20：25 | 20：25-22：00 | 22：00-22：20 | 22：20-23：14 | 23：14-01：20 | 01：20-06：00 |
| 7月 | 成交价 | 21800 | 22533 | 26066 | 28800 | 29433 | 18600 | 13266 | 8860 | 7116 |
| 底价 | 14733 | 20466 | 21600 | 19966 | 19600 | 14066 | 13000 | 8806 | 7023 |
| 8月 | 成交价 | 21028 | 21590 | 24895 | 27600 | 28961 | 18057 | 12509 | 8720 | 7023 |
| 底价 | 14276 | 20552 | 21700 | 19695 | 19271 | 13895 | 12714 | 8775 | 6999 |
| 9月 | 成交价 | 20257 | 20647 | 23723 | 26400 | 28490 | 17514 | 11752 | 8580 | 6930 |
| 底价 | 13819 | 20638 | 21800 | 19423 | 18942 | 13723 | 12428 | 8743 | 6974 |

表2.2

根据表2.1中预测数据，再结合之前收集的1月-6月的数据，可大致看出该季度每月广告成交价和底价整体呈下降趋势，依据在中国广告网找到的官方数据显示（由下图所示），可以看出广告花费整体上也呈现下降趋势，与模型所预测的趋势基本吻合，可知预测的底价结果具有较高的准确度。



**5.3 基于已知电视频道用户收视历史情况和在播视频广告的产品销售情况，建立视频广告与频道用户的分类匹配推送更新模型**

通过用户的收视历史广告和热播视频广告的对照分析，在综合指标的对比下可以预测出当前热播广告的分类情况，再由表1.2可以匹配出相对应的用户种类。为了较为科学的对热播广告进行匹配，本文引用多分类贝叶斯的方法来进行推送更新模型的建立。

**5.3.1 分类结构的建立**

为了使用朴素贝叶斯算法对历史广告进行训练，我们先提取了各个广告的关键词作为指标，接着用提取的指标构建不重复词表。不重复词表如下：

['形象', '风趣', '炫酷', '想象', '生活习惯', '功能', '演示', '可信度高', '顾客证词', '搞笑', '幼稚', '易懂', '反馈', '消费', '感染力', '气质', '风度', '童趣', '生活']

**5.3.2 朴素贝叶斯算法**

根据相关资料得到朴素贝叶斯算法，步骤如下：

输入：训练数据,其中,是第i个样本的第j个特征，,是第j个特征可能取的第l个值,,,；实例x；

输出：实例x的分类。

1. 计算先验概率及条件概率
2. 对于给定的实例,计算
3. 确定实例x的类

详细代码见附录5

**5.3.3 数据统计**

为使预测分类具有科学性，本文选取了数十条历史数据并且对正在热播的广告进行了关键字的提取。数据如下所示：

|  |  |  |  |
| --- | --- | --- | --- |
| 历史观看广告 |  | 特征 | 类型匹配 |
| 霸王洗发水 | 生活习惯 消费 功能 | 实地示范 |
| 金噪子喉片 | 功能 生活 消费 | 实地示范 |
| 蓝翔技校 | 搞笑 可信度高 感染力 | 满意顾客证词 |
| 妇炎洁 | 功能 生活 形象 消费 | 实地示范 |
| 金坷垃 | 搞笑 形象 功能 | 实地示范 |
|  | | |
| 滴滴打车 | 搞笑 生活 风趣 形象 易懂 | 幽默 |
| 大众汽车 | 生活 搞笑 风趣 消费 | 幽默 |
| 公益广告 | 易懂 形象 感染力 形象 | 动画 |
| Prospecs | 炫酷 易懂 消费 感染力 | 以视觉效果为主 |
| 中国移动 | 生活 可信度高 反馈 功能 | 满意顾客证词 |
| 康师傅 | 风趣 功能 消费 演示 | 实地示范 |
| 当前热播广告 | BesTV | 独立创意 生活习惯 功能 消费 | 待预测 |
| 畅意 | 可信度高 消费 形象 生活 消费 搞笑 | 待预测 |
| 喜马拉雅app | 搞笑 风趣 形象 形象 演示 独立创意 | 待预测 |
| 肠炎宁片 | 反馈 消费 广告歌 生活 功能 中年 | 待预测 |
| 丸美眼霜 | 演示 生活 功能 消费 炫酷 想象 气质 女性 | 待预测 |
|  | | |
| 百事可乐 | 广告歌 搞笑 气质 风趣 消费 易懂 | 待预测 |
| 雅培奶粉 | 国家品牌 功能 可信度高 婴幼儿 消费 | 待预测 |
| 好爸爸洗衣露 | 功能 可信度高 消费 风趣 | 待预测 |
| 安慕希 | 功能 消费 可信度高 气质 生活 | 待预测 |
| 补肺丸 | 中年 功能 消费 形象 易懂 顾客证词 | 待预测 |

为了解决某个分量在某个分类中未出现过从而导致结果为0，我们使用拉普拉斯平滑进行了处理，为了防止因为小数连乘而造成向下溢出，我们对计算过程进行了取对数操作。部分结果如图所示：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 广告\种类 | 实地示范 | 满意顾客证词 | 生活风格 | 动画 | 以视觉效果为主 | 幽默 |
| BesTV | -4.2669 | -5.926 | -5.92692 | -6.23 | -5.2574 | -5.9269 |
| 畅意 | -7.4251 | -7.066 | -6.6608 | -8.45 | -8.03008 | -5.56228 |
| 肠炎宁片 | -5.5662 | -5.233 | -6.43775 | -8.31 | -6.64378 | -6.43775 |
| 拼多多 | -8.61905 | -7.354 | -8.96348 | -11.7 | -9.41637 | -9.65662 |
| 多芬 | -4.87308 | -4.763 | -5.74460 | -8.31 | -6.64378 | -6.43775 |
| 清扬 | -5.74855 | -5.233 | -7.53636 | -9.41 | -7.33693 | -8.22951 |
| 阿迪达斯 | -5.51970 | -5.233 | -5.23377 | -6.23 | -4.15888 | -5.92692 |

**5.3.4 模型求解**

通过朴素贝叶斯算法建立的多分类动态推送模型，结合历史数据，预测了当前热播广告的类型，结果如下表所示，再结合表1.2的匹配结果，将向少儿推送可爱多雪糕等广告，向青少年推送畅意、百事可乐等广告，向青年推送丸美眼霜、巴黎欧莱雅等广告，向中年推送补肺丸等广告，向中老年推送安慕希等广告，向老年推送好爸爸洗衣露等广告。通过实时更新动态推送模型，实现对不同类型的观看用户推送相应的匹配广告类型。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 广告名称 | 广告类型 | 用户种类 | 广告名称 | 广告类型 | 用户种类 |
| BesTV | 实地示范 | 老年 | **太平饼干** | 实地示范 | 老年 |
| 畅意 | 幽默 | 青少年 | **巴黎欧莱雅** | 以视觉效果为主 | 青年 |
| 喜马拉雅app | 幽默 | 青少年 | **雪碧** | 满意顾客证词 | 中年 |
| 肠炎宁片 | 满意顾客证词 | 中年 | **炫迈** | 幽默 | 青少年 |
| 丸美眼霜 | 以视觉效果为主 | 青年 | **vivox27** | 以视觉效果为主 | 青年 |
| 拼多多 | 满意顾客证词 | 中年 | **KFC** | 幽默 | 青少年 |
| 多芬 | 满意顾客证词 | 中年 | **奥妙** | 实地示范 | 老年 |
| 清扬 | 满意顾客证词 | 中年 | **安慕希** | 生活风格 | 中老年 |
| 益安宁丸 | 满意顾客证词 | 中年 | **君乐宝涨芝士啦** | 幽默 | 青少年 |
| 君乐宝奶粉 | 满意顾客证词 | 中年 | **百事可乐** | 幽默 | 青少年 |
| 可爱多雪糕 | 动画 | 少儿 | **雅培奶粉** | 实地示范 | 老年 |
| 金水宝胶囊 | 满意顾客证词 | 中年 | **好爸爸洗衣露** | 实地示范 | 老年 |
| 阿迪达斯 | 以视觉效果为主 | 青年 | **康师傅绿茶** | 幽默 | 青少年 |
| 养血补肾丸 | 满意顾客证词 | 中年 | **补肺丸** | 动画 | 少儿 |
| 金霸王电池 | 以视觉效果为主 | 青年 |  |  |  |

表3.1

可以看出，正在热播的广告主要受众人群为中年和青少年，与当前电视广的发展趋势也较为吻合。

**5.4 基于更新的分户推送方式建立竞价交易模型，在极大化卖方收益的同时提升收视率和买方产品销售量**

将近期热播广告与以往各时段广告的投放情况进行对照分析，在综合指标的对比下可以较为准确地估算出热播广告在不同时段的投放情况。为了较为科学的对广告投放情况进行评价，本题引用逻辑回归算法来进行模型的建立。

**5.4.1 评价指标体系的建立**

为了对广告投放情况进行综合客观的评价，首先需要建立一个用于评价的指标体系。评价指标体系如下图所示：

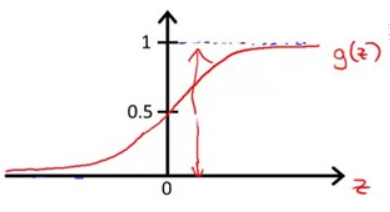
|  |  |  |
| --- | --- | --- |
| 广告投放评价指标体系 | | |
| 指标1 | 指标2 | 指标3 |
| 电视台即将收益 | 电视广告上一季度收视率（%） | 制造商上一季度产品销售量 |

**5.4.2 逻辑回归算法**

1. 构造预测函数

逻辑回归算法实际上是一种分类算法，主要用于二分类问题，设预测函数（h），显然，该函数的输出必须是两个值，所以利用Sigmoid函数，函数形式为：

对应的函数图像是一个取值在0和1之间的s型曲线，如图：



对于线性边界情况，构造的预测函数为：

1. 模型参数估计

逻辑斯蒂回归模型学习时，对于给定的训练数据集，其中，，,可以应用极大似然估计法估计模型参数，从而得到逻辑斯谛回归模型。

设：

似然函数为

对数似然函数为

对L(w)求极大值，得到w的估计值。

这样，问题就变成了以对数似然函数为目标函数的最优化问题，逻辑斯谛回归学习通常采用的方法是梯度下降及拟牛顿法。

假设w的极大似然估计值是，那么学到的逻辑斯谛回归模型为：

详细代码见附录6

**5.4.3 数据统计**

为了有针对性的投放广告，我们在中国广告网上搜集了不同时段不同用户观看电视的人数（详情可见附件1），利用matlab将附录表一用饼状图表示，如图所示：



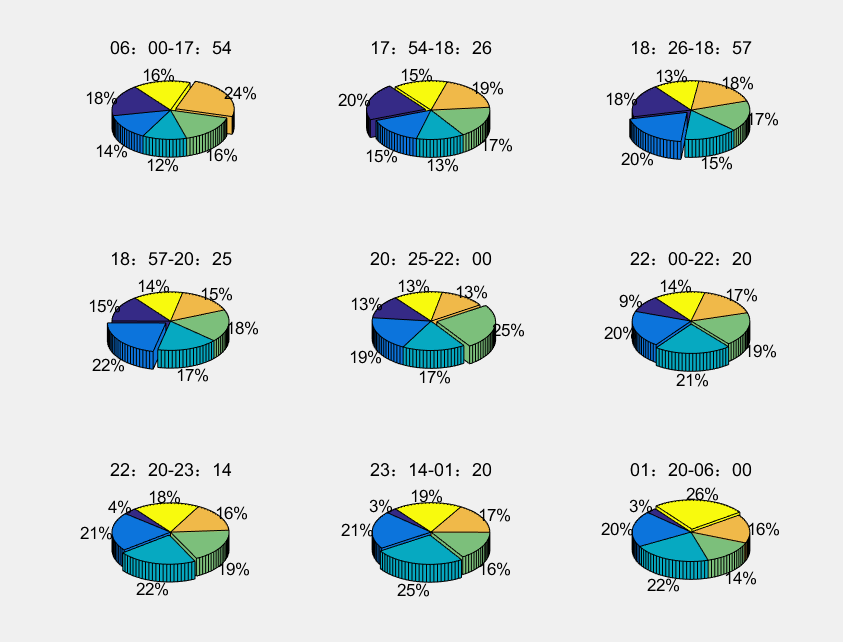


图4.1

由图4.1可以看出，每个时间段对应一种用户类型，6：00-17：54这个时间段中老年占主要比重，17：54-18：26少儿居多，18：26-18：57和18：57-20：25这两个时间段中，青少年人数居多，20：25-22：00这个时间段内中年观众大于其他类型观众，22：00-22：20，22：20-23.14以及23：14-01：20这三个时间段内，青年观众人数较多，01：20-06：00老年观众居多。对电视台来说，可以根据每个时间段对应的用户特征以便于确定投放广告的类型，从而提高电视台的收视率和产品的销售量。

经计算求解得逻辑斯谛回归模型为：

**5.4.4 模型求解**

由于三个指标的数据维度差异较大，因此我们采用将数据进行处理，使所有数据都在0-1范围内，这样可以加快模型优化的速度。

根据问题三的结果和收集的相关数据建立的竞价交易模型确定了广告是否投放，详情见附录，1表示投放，0表示不投放。同时考虑到投放的每个广告带来的利润不同，我们计算了每种广告的c值大小，在极大化卖方收益情况下选取少部分投入产出比较小的广告作为此时段主要投放广告，从而实现提升收视率的同时增加买方产品销售量。

6：00-17：54在确立的已投放广告中，选取的主要投放广告是安慕希、可爱多雪糕、康师傅绿茶、畅意和KFC，满足此时段较多观看人群对应的广告类型，c值也较小；17：54-18：26这个时间段主要投放的广告是可爱多雪糕、vivox27、康师傅绿茶和肠炎宁片；18：26-18：57主要投放的广告是喜马拉雅app、炫迈和雅培奶粉；18：57-20：25主要投放的广告是肠炎宁片、炫迈、安慕希和康师傅绿茶；20：25-22：00主要投放的广告是畅意、vivox27和安慕希；22：00-22：20主要投放的广告是补肺丸、丸美眼霜；22：20-23:14主要投放的广告是安慕希、多芬和百事可乐；23：14-01：20主要投放的广告是康师傅绿茶、金水宝胶囊和KFC；01：20-06:00主要投放的广告是阿迪达斯 、巴黎欧莱雅和太平饼干。

1. **模型的检验、灵敏度分析**

**6.1 模型检验：**

问题一中我们分别采用欧式距离和cos相似度的方法对问题及进行求解，求解结果与统计数据得到的结果完全一致，因此符合实际情况。

问题二中我们在中国广告网找了2018年广告价格的波动趋势，我们预测的结果与之基本一致，因此也符合实际情况。

问题三中我们通过已有数据进行训练学习和比较来检验模型。经检验证实建立的模型均基本符合模型建立初衷。

问题四中我们根据历史数据并且结合问题三的结果对收集的热播广告进行预测与分析，得出的结论较为合理，也基本符合实际情况。

**6.2 灵敏度分析：**

由于本题采用的数据基本都是离散数据，而非连续数据，因此改变某个数据对模型影响较低。

1. **模型的优缺点分析，模型的推广**

**7.1 模型优点：**

本文从多个角度考虑广告的特征和用户特征之间的关系，以及广告在不同时间段对于不同用户的销量和卖方收益之间的关系。本文利用excel，MATLAB等多种工具并绘制了较多图表，使模型叙述清晰流畅。在进行预测时采用了欧式距离，cos相似度，朴素贝叶斯算法，线性回归算法，逻辑斯谛回归等多种模型，并在解题过程中根据不同情况分别优化选取了不同算法，使最后结果较为满意。

**7.2 模型缺点：**

由于时间紧迫以及数据量的不足，使部分模型较为粗糙，同时由于时间和设备以及自身水平的局限，计算结果可能存在误差。

1. **参考文献**

[1] 中国广告网

[2]百度百科，投入产出比<https://baike.baidu.com/item/%E6%8A%95%E5%85%A5%E4%BA%A7%E5%87%BA%E6%AF%94>

[3] 李航 《统计方法》

[4] 姜启源 《数学方法》

[5] CSM数据网 <http://www.csm-huan.com/>

[6] 腾讯广告算法大赛 <https://algo.qq.com/>

[7] 数据归一化 <https://blog.csdn.net/pipisorry/article/details/52247379>

[8] 拉普拉斯平滑 <https://blog.csdn.net/Wprofessor/article/details/86931062>

1. **附录**

附件1

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 时间段 | 观看总数（万） | 少儿（万） | 青少年（万） | 青年（万） | 中年（万） | 中老年（万） | 老年（万） | 特征人群 |
| 06:00-17:54 | 3894.6 | 687.4 | 551.1 | 475.9 | 633.3 | 928.8 | 618.1 | 中老年 |
| 17.65% | 14.15% | 12.22% | 16.26% | 23.85% | 15.87% |
| 17：54-18：26 | 3797.2 | 773.1 | 578.3 | 500.5 | 658.8 | 721.1 | 565.4 | 少儿 |
| 20.36% | 15.23% | 13.18% | 17.35% | 18.99% | 14.89% |
| 18：26-18：57 | 4191.3 | 764.5 | 824.7 | 615.7 | 713.4 | 746.2 | 526.8 | 青少年 |
| 18.24% | 19.68% | 14.69% | 17.02% | 17.80% | 12.57% |
| 18：57-20：25 | 4456.8 | 647.1 | 960.9 | 750.9 | 799.9 | 684.6 | 613.3 | 青少年 |
| 14.52% | 21.56% | 16.85% | 17.95% | 15.36% | 13.76% |
| 20：25-22：00 | 4302.3 | 549.4 | 815.7 | 744.3 | 1069.9 | 542.95 | 579.9 | 中年 |
| 12.77% | 18.96% | 17.30% | 24.87% | 12.62% | 13.48% |
| 22：00-22：20 | 3019.8 | 279.3 | 610.9 | 636.6 | 561.9 | 501.9 | 429.1 | 青年 |
| 9.25% | 20.23% | 21.08% | 18.61% | 16.62% | 14.21% |
| 22：20-23：14 | 2612 | 95.3 | 557.9 | 570.9 | 490.3 | 417.1 | 480.3 | 青年 |
| 3.65% | 21.36% | 21.86% | 18.77% | 15.97% | 18.39% |
| 23：14-01：20 | 1259.7 | 38.4 | 260.9 | 313.5 | 199.8 | 208.7 | 238.3 | 青年 |
| 3.05% | 20.71% | 24.89% | 15.86% | 16.57% | 18.92% |
| 01：20-06：00 | 752.9 | 21.8 | 149.2 | 162.4 | 107.3 | 119.5 | 192.7 | 老年 |
| 2.90% | 19.82% | 21.57% | 14.25% | 15.87% | 25.59% |

附件2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 时间段 | 广告 | 电视台即将收益 | 电视广告上一季度收视率（%） | 制造商上一季度产品销售量 | 是否投放 | c=投入/产出 |
| 06：00-17：54 | 金霸王电池 | 19000 | 0.458 | 12358 | 0 | 1.537465609 |
| 肠炎宁片 | 22000 | 0.3078 | 15468 | 0 | 1.422291182 |
| 巴黎欧莱雅 | 25000 | 0.6942 | 31586 | 1 | 0.791489901 |
| BesTV | 17000 | 0.0976 | 1562 | 0 | 10.88348271 |
| 丸美眼霜 | 25000 | 0.4183 | 19865 | 0 | 1.25849484 |
| vivox27 | 23000 | 0.7353 | 31762 | 1 | 0.72413576 |
| 奥妙 | 24000 | 0.6801 | 28731 | 1 | 0.835334656 |
| 喜马拉雅app | 20000 | 0.2586 | 2347 | 0 | 8.52151683 |
| 百事可乐 | 26000 | 1.0729 | 38752 | 1 | 0.670933113 |
| 阿迪达斯 | 24500 | 0.5904 | 26485 | 0 | 0.925051916 |
| 君乐宝涨芝士啦 | 24000 | 0.6612 | 31354 | 1 | 0.765452574 |
| 养血补肾丸 | 18500 | 0.2949 | 5623 | 0 | 3.290058688 |
| 康师傅绿茶 | 21000 | 0.6622 | 36527 | 1 | 0.574917185 |
| 可爱多雪糕 | 25500 | 0.976 | 45328 | 1 | 0.562566184 |
| 雅培奶粉 | 23000 | 0.513 | 2685 | 0 | 8.566108007 |
| 炫迈 | 23000 | 0.5442 | 19879 | 0 | 1.156999849 |
| 金水宝胶囊 | 22500 | 0.0981 | 335 | 0 | 67.1641791 |
| 畅意 | 26000 | 1.9011 | 42582 | 1 | 0.610586633 |
| 太平饼干 | 25500 | 0.9378 | 36478 | 1 | 0.699051483 |
| 补肺丸 | 21000 | 0.0671 | 269 | 0 | 78.0669145 |
| 安慕希 | 24500 | 0.9251 | 38545 | 1 | 0.635620703 |
| 多芬 | 19000 | 0.5448 | 16741 | 0 | 1.134938176 |
| KFC | 20500 | 0.7083 | 32985 | 1 | 0.621494619 |
| 17:54-18:26 | 金霸王电池 | 28000 | 1.8761 | 48358 | 1 | 0.579014848 |
| 肠炎宁片 | 22500 | 0.9451 | 38642 | 1 | 0.582267999 |
| 巴黎欧莱雅 | 26000 | 0.8751 | 43258 | 1 | 0.601044893 |
| BesTV | 19000 | 0.0253 | 253 | 0 | 75.09881423 |
| 丸美眼霜 | 22000 | 1.0257 | 39746 | 1 | 0.553514819 |
| vivox27 | 27500 | 1.5384 | 50028 | 1 | 0.549692172 |
| 奥妙 | 21000 | 0.6294 | 26485 | 1 | 0.792901642 |
| 喜马拉雅app | 19500 | 0.2764 | 7840 | 0 | 2.487244898 |
| 百事可乐 | 25500 | 0.8936 | 41788 | 1 | 0.610223031 |
| 阿迪达斯 | 28000 | 0.7456 | 17394 | 0 | 1.609750489 |
| 君乐宝涨芝士啦 | 20000 | 0.2617 | 13089 | 0 | 1.528000611 |
| 养血补肾丸 | 20000 | 0.0894 | 743 | 0 | 26.9179004 |
| 康师傅绿茶 | 25500 | 0.9577 | 45821 | 1 | 0.556513389 |
| 可爱多雪糕 | 24500 | 0.7465 | 57821 | 1 | 0.423721485 |
| 雅培奶粉 | 26500 | 0.5792 | 29431 | 0 | 0.900411131 |
| 炫迈 | 19200 | 0.3654 | 19832 | 0 | 0.968132311 |
| 金水宝胶囊 | 19000 | 0.2834 | 2747 | 0 | 6.916636331 |
| 畅意 | 26000 | 0.8364 | 50811 | 1 | 0.511700222 |
| 太平饼干 | 20500 | 0.4872 | 26731 | 0 | 0.766899854 |
| 补肺丸 | 19500 | 0.0326 | 1739 | 0 | 11.213341 |
| 安慕希 | 28000 | 1.2748 | 39689 | 1 | 0.705485147 |
| 多芬 | 22000 | 0.7837 | 27494 | 1 | 0.800174584 |
| KFC | 25500 | 0.8943 | 39751 | 1 | 0.641493296 |
| 18:26-18:57 | 金霸王电池 | 32000 | 0.9876 | 45327 | 1 | 0.705980983 |
| 肠炎宁片 | 28500 | 0.1093 | 7495 | 0 | 3.802535023 |
| 巴黎欧莱雅 | 23000 | 0.4672 | 25473 | 0 | 0.902916814 |
| BesTV | 30000 | 1.0743 | 38939 | 1 | 0.77043581 |
| 丸美眼霜 | 22500 | 0.2697 | 8691 | 0 | 2.588885054 |
| vivox27 | 28000 | 1.4652 | 47359 | 1 | 0.5912287 |
| 奥妙 | 32000 | 1.685 | 52394 | 1 | 0.610756957 |
| 喜马拉雅app | 30000 | 1.235 | 48625 | 1 | 0.616966581 |
| 百事可乐 | 28500 | 0.6523 | 27529 | 0 | 1.035271895 |
| 阿迪达斯 | 31500 | 1.2321 | 41398 | 1 | 0.760906324 |
| 君乐宝涨芝士啦 | 28500 | 0.6354 | 33857 | 0 | 0.841775704 |
| 养血补肾丸 | 29000 | 0.5968 | 32898 | 0 | 0.881512554 |
| 康师傅绿茶 | 29500 | 1.002 | 41938 | 1 | 0.703419333 |
| 可爱多雪糕 | 26000 | 0.4667 | 27499 | 0 | 0.945488927 |
| 雅培奶粉 | 22500 | 0.8439 | 39765 | 1 | 0.565824217 |
| 炫迈 | 22000 | 1.1474 | 47932 | 1 | 0.45898356 |
| 金水宝胶囊 | 22500 | 0.0629 | 2846 | 0 | 7.905832748 |
| 畅意 | 30500 | 1.4826 | 52882 | 1 | 0.576755796 |
| 太平饼干 | 27500 | 0.4242 | 26484 | 0 | 1.038362785 |
| 补肺丸 | 24000 | 0.1943 | 6583 | 0 | 3.645754215 |
| 安慕希 | 32000 | 1.0364 | 47692 | 1 | 0.670972071 |
| 多芬 | 30000 | 0.5472 | 34788 | 0 | 0.862366333 |
| KFC | 31500 | 0.9263 | 42737 | 1 | 0.737066242 |
| 18:57-20:25 | 金霸王电池 | 26000 | 0.1254 | 1132 | 0 | 22.96819788 |
| 肠炎宁片 | 33500 | 1.0102 | 53296 | 1 | 0.628564995 |
| 巴黎欧莱雅 | 36000 | 1.4678 | 52913 | 1 | 0.680362104 |
| BesTV | 24000 | 0.0514 | 689 | 0 | 34.83309144 |
| 丸美眼霜 | 35000 | 1.2569 | 49867 | 1 | 0.701866966 |
| vivox27 | 36000 | 1.8467 | 46329 | 1 | 0.777051091 |
| 奥妙 | 32400 | 0.9783 | 42463 | 1 | 0.763017215 |
| 喜马拉雅app | 27500 | 0.5738 | 26482 | 0 | 1.038441205 |
| 百事可乐 | 34500 | 1.2537 | 44261 | 1 | 0.779467251 |
| 阿迪达斯 | 35500 | 1.6532 | 51730 | 1 | 0.686255558 |
| 君乐宝涨芝士啦 | 28500 | 0.7839 | 37652 | 1 | 0.756931903 |
| 养血补肾丸 | 26000 | 0.3673 | 14773 | 0 | 1.759967508 |
| 康师傅绿茶 | 30000 | 0.7362 | 44626 | 1 | 0.672253843 |
| 可爱多雪糕 | 33000 | 0.6823 | 38783 | 1 | 0.850888276 |
| 雅培奶粉 | 32500 | 0.6372 | 32643 | 1 | 0.995619275 |
| 炫迈 | 32000 | 1.5318 | 54839 | 1 | 0.583526323 |
| 金水宝胶囊 | 27500 | 0.1253 | 7499 | 0 | 3.667155621 |
| 畅意 | 35500 | 1.4673 | 47298 | 1 | 0.750560277 |
| 太平饼干 | 34500 | 0.9842 | 40327 | 1 | 0.855506237 |
| 补肺丸 | 26500 | 0.2413 | 27343 | 0 | 0.96916944 |
| 安慕希 | 36000 | 1.9363 | 56393 | 1 | 0.638377104 |
| 多芬 | 34500 | 0.8837 | 47598 | 1 | 0.724820371 |
| KFC | 35000 | 1.0372 | 45287 | 1 | 0.77284872 |
| 20:25-22:00 | 金霸王电池 | 32000 | 1.3542 | 40125 | 1 | 0.797507788 |
| 肠炎宁片 | 25500 | 0.6352 | 28764 | 0 | 0.886524823 |
| 巴黎欧莱雅 | 29500 | 0.4671 | 16349 | 0 | 1.804391706 |
| BesTV | 23000 | 0.2749 | 10734 | 0 | 2.142724054 |
| 丸美眼霜 | 30500 | 1.2643 | 40723 | 1 | 0.748962503 |
| vivox27 | 32000 | 1.5473 | 47623 | 1 | 0.671944229 |
| 奥妙 | 30500 | 1.2376 | 42126 | 1 | 0.724018421 |
| 喜马拉雅app | 24500 | 0.4321 | 18467 | 0 | 1.326690854 |
| 百事可乐 | 31000 | 0.9372 | 35765 | 1 | 0.866769188 |
| 阿迪达斯 | 31500 | 1.0252 | 42627 | 1 | 0.73896826 |
| 君乐宝涨芝士啦 | 22000 | 0.5213 | 28462 | 0 | 0.772960438 |
| 养血补肾丸 | 22000 | 0.1732 | 8362 | 0 | 2.630949534 |
| 康师傅绿茶 | 31500 | 1.5698 | 45231 | 1 | 0.696425018 |
| 可爱多雪糕 | 20500 | 0.1694 | 3982 | 0 | 5.14816675 |
| 雅培奶粉 | 20000 | 0.08681 | 694 | 0 | 28.8184438 |
| 炫迈 | 30500 | 1.5237 | 43685 | 1 | 0.698180153 |
| 金水宝胶囊 | 26500 | 0.6526 | 26585 | 0 | 0.996802708 |
| 畅意 | 31500 | 1.2763 | 48276 | 1 | 0.652498136 |
| 太平饼干 | 26500 | 0.4839 | 26347 | 0 | 1.005807113 |
| 补肺丸 | 23500 | 0.3645 | 50321 | 0 | 0.467001848 |
| 安慕希 | 32000 | 1.6382 | 47728 | 1 | 0.670465974 |
| 多芬 | 30000 | 1.3723 | 43820 | 1 | 0.684618895 |
| KFC | 30500 | 1.2842 | 37218 | 1 | 0.819495943 |
| 22:00-22:20 | 金霸王电池 | 16000 | 0.2157 | 13654 | 0 | 1.171817782 |
| 肠炎宁片 | 20000 | 0.7523 | 29873 | 1 | 0.669500887 |
| 巴黎欧莱雅 | 21500 | 0.9658 | 30112 | 1 | 0.714001063 |
| BesTV | 18000 | 0.1137 | 5621 | 0 | 3.202277175 |
| 丸美眼霜 | 22000 | 1.2637 | 32764 | 1 | 0.671468685 |
| vivox27 | 22000 | 0.9732 | 29573 | 1 | 0.743921821 |
| 奥妙 | 21500 | 0.7628 | 25348 | 1 | 0.848193151 |
| 喜马拉雅app | 16500 | 0.1738 | 6432 | 0 | 2.565298507 |
| 百事可乐 | 20500 | 0.6721 | 27580 | 1 | 0.743292241 |
| 阿迪达斯 | 22000 | 0.8842 | 30263 | 1 | 0.726960315 |
| 君乐宝涨芝士啦 | 22000 | 0.7237 | 29647 | 1 | 0.742064964 |
| 养血补肾丸 | 21500 | 0.6324 | 27984 | 1 | 0.768296169 |
| 康师傅绿茶 | 21000 | 0.5892 | 27544 | 1 | 0.762416497 |
| 可爱多雪糕 | 21500 | 0.6298 | 23573 | 1 | 0.912060408 |
| 雅培奶粉 | 21000 | 0.6523 | 25433 | 1 | 0.825698895 |
| 炫迈 | 15500 | 0.2654 | 6893 | 0 | 2.248658059 |
| 金水宝胶囊 | 16500 | 0.3234 | 10788 | 0 | 1.529477197 |
| 畅意 | 16500 | 0.1263 | 2132 | 0 | 7.739212008 |
| 太平饼干 | 18500 | 0.2582 | 5817 | 0 | 3.180333505 |
| 补肺丸 | 19000 | 0.4893 | 26304 | 1 | 0.722323601 |
| 安慕希 | 20000 | 0.5623 | 24899 | 1 | 0.80324511 |
| 多芬 | 19500 | 0.5438 | 27421 | 1 | 0.711133803 |
| KFC | 22000 | 0.8473 | 29578 | 1 | 0.743796065 |
| 22:20-23:14 | 金霸王电池 | 17000 | 0.8975 | 23541 | 1 | 0.722144344 |
| 肠炎宁片 | 18000 | 0.8531 | 22983 | 1 | 0.783187573 |
| 巴黎欧莱雅 | 16000 | 0.2568 | 6814 | 0 | 2.348106839 |
| BesTV | 16500 | 0.9764 | 30432 | 1 | 0.542192429 |
| 丸美眼霜 | 17500 | 0.6923 | 19482 | 1 | 0.898265065 |
| vivox27 | 16000 | 0.5324 | 16320 | 1 | 0.980392157 |
| 奥妙 | 13500 | 0.2467 | 4246 | 0 | 3.179463024 |
| 喜马拉雅app | 15000 | 0.4768 | 16342 | 0 | 0.917880308 |
| 百事可乐 | 17500 | 1.0782 | 32198 | 1 | 0.543512019 |
| 阿迪达斯 | 16000 | 0.6213 | 23834 | 1 | 0.671309893 |
| 君乐宝涨芝士啦 | 16500 | 0.5423 | 19732 | 1 | 0.836205149 |
| 养血补肾丸 | 13000 | 0.1732 | 3672 | 0 | 3.540305011 |
| 康师傅绿茶 | 14500 | 0.2543 | 7823 | 0 | 1.853508884 |
| 可爱多雪糕 | 14000 | 0.2318 | 5123 | 0 | 2.732773765 |
| 雅培奶粉 | 17000 | 0.7923 | 25312 | 1 | 0.671618205 |
| 炫迈 | 18000 | 0.7438 | 23442 | 1 | 0.767852572 |
| 金水宝胶囊 | 16500 | 0.5836 | 16372 | 1 | 1.007818226 |
| 畅意 | 14500 | 0.4128 | 6425 | 0 | 2.256809339 |
| 太平饼干 | 14000 | 0.4231 | 8912 | 0 | 1.570915619 |
| 补肺丸 | 14500 | 0.3831 | 3341 | 0 | 4.340017959 |
| 安慕希 | 17500 | 0.9263 | 32932 | 1 | 0.531398032 |
| 多芬 | 16000 | 0.763 | 28732 | 1 | 0.556870388 |
| KFC | 16500 | 0.6732 | 24365 | 1 | 0.677200903 |
| 23:14-01:20 | 金霸王电池 | 9100 | 0.1287 | 1029 | 0 | 8.843537415 |
| 肠炎宁片 | 9400 | 0.6537 | 7642 | 0 | 1.230044491 |
| 巴黎欧莱雅 | 9550 | 1.0003 | 16854 | 1 | 0.566631067 |
| BesTV | 8800 | 0.0372 | 783 | 0 | 11.23882503 |
| 丸美眼霜 | 8850 | 0.1734 | 1231 | 0 | 7.189277011 |
| vivox27 | 9450 | 0.8723 | 19347 | 1 | 0.488447821 |
| 奥妙 | 9500 | 0.7534 | 17438 | 1 | 0.544787246 |
| 喜马拉雅app | 9500 | 0.9473 | 18934 | 1 | 0.501742896 |
| 百事可乐 | 9600 | 1.1738 | 21932 | 1 | 0.437716579 |
| 阿迪达斯 | 8950 | 0.3123 | 7823 | 0 | 1.14406238 |
| 君乐宝涨芝士啦 | 8900 | 0.2989 | 832 | 0 | 10.69711538 |
| 养血补肾丸 | 9050 | 0.3238 | 932 | 0 | 9.710300429 |
| 康师傅绿茶 | 9050 | 1.0276 | 21903 | 1 | 0.413185408 |
| 可爱多雪糕 | 9100 | 0.7632 | 17233 | 1 | 0.528056636 |
| 雅培奶粉 | 9300 | 0.8236 | 18932 | 1 | 0.491231777 |
| 炫迈 | 9350 | 0.7324 | 19348 | 1 | 0.483254083 |
| 金水宝胶囊 | 9550 | 0.9734 | 20128 | 1 | 0.474463434 |
| 畅意 | 9750 | 1.1288 | 21303 | 1 | 0.457682017 |
| 太平饼干 | 8850 | 0.1253 | 1928 | 0 | 4.590248963 |
| 补肺丸 | 8900 | 0.0126 | 673 | 0 | 13.2243685 |
| 安慕希 | 9050 | 0.4821 | 7834 | 0 | 1.155220832 |
| 多芬 | 9150 | 0.5781 | 12293 | 0 | 0.744326039 |
| KFC | 9650 | 0.8923 | 19372 | 1 | 0.498141648 |
| 01:20-06:00 | 金霸王电池 | 7700 | 0.1394 | 568 | 0 | 13.55633803 |
| 肠炎宁片 | 7700 | 0.2698 | 438 | 0 | 17.57990868 |
| 巴黎欧莱雅 | 7800 | 0.6858 | 2658 | 1 | 2.934537246 |
| BesTV | 7750 | 0.3651 | 1195 | 0 | 6.485355649 |
| 丸美眼霜 | 7800 | 0.5139 | 2258 | 1 | 3.454384411 |
| vivox27 | 7650 | 0.5623 | 2187 | 1 | 3.497942387 |
| 奥妙 | 7300 | 0.4023 | 1243 | 1 | 5.872888174 |
| 喜马拉雅app | 7450 | 0.4892 | 1334 | 1 | 5.584707646 |
| 百事可乐 | 7700 | 0.5623 | 1947 | 1 | 3.95480226 |
| 阿迪达斯 | 7550 | 0.5067 | 1527 | 1 | 4.944335298 |
| 君乐宝涨芝士啦 | 7500 | 0.4976 | 1289 | 1 | 5.818463926 |
| 养血补肾丸 | 7350 | 0.2536 | 372 | 0 | 19.75806452 |
| 康师傅绿茶 | 7300 | 0.2174 | 257 | 0 | 28.40466926 |
| 可爱多雪糕 | 7200 | 0.1278 | 89 | 0 | 80.8988764 |
| 雅培奶粉 | 7250 | 0.0653 | 42 | 0 | 172.6190476 |
| 炫迈 | 7600 | 0.5193 | 2152 | 1 | 3.531598513 |
| 金水宝胶囊 | 7600 | 0.4967 | 1893 | 1 | 4.014791337 |
| 畅意 | 7300 | 0.3871 | 1289 | 1 | 5.663304888 |
| 太平饼干 | 7250 | 0.3138 | 1324 | 1 | 5.475830816 |
| 补肺丸 | 7650 | 0.4798 | 1543 | 1 | 4.957874271 |
| 安慕希 | 7600 | 0.5238 | 2046 | 1 | 3.714565005 |
| 多芬 | 7800 | 0.5729 | 2641 | 1 | 2.953426732 |
| KFC | 7200 | 0.2671 | 387 | 0 | 18.60465116 |

附件3

核心代码：

# 欧式距离  
def Euldist(listnewVocaset, label):  
 fr = open('C:\\Users\\王教授\\Desktop\\aaa.txt')  
 kinds = []  
 kindData = []  
 for line in fr.readlines():  
 line = line.strip().split()  
 kindData.append(line[1:])  
 kinds.append(line[0])  
 kindListnewVocaset = []  
 for listvocaset in kindData:  
 kindListnewVocaset.append(setword(vocaset, listvocaset))  
 listnewVocaset = np.array(listnewVocaset)  
 kindListnewVocaset = np.array(kindListnewVocaset)  
 print(kindListnewVocaset)  
 distMat = []  
 for i in range(len(label)):  
 temp = []  
 for j in range(len(kinds)):  
 diffMat = (listnewVocaset[j] - kindListnewVocaset[i]) \*\* 2  
 temp.append(np.sum(diffMat) \*\* 0.5)  
 distMat.append(temp)  
 return distMat  
  
  
def cosSimilarity(listnewVocaset, label):  
 fr = open('C:\\Users\\王教授\\Desktop\\aaa.txt')  
 kinds = []  
 kindData = []  
 for line in fr.readlines():  
 line = line.strip().split()  
 kindData.append(line[1:])  
 kinds.append(line[0])  
 kindListnewVocaset = []  
 for listvocaset in kindData:  
 kindListnewVocaset.append(setword(vocaset, listvocaset))  
 listnewVocaset = np.array(listnewVocaset)  
 kindListnewVocaset = np.array(kindListnewVocaset)  
 distMat = []  
 for i in range(len(label)):  
 temp = []  
 for j in range(len(kinds)):  
 formatM = np.sum(listnewVocaset[j] \* kindListnewVocaset[i])  
 formatZ = (np.sum(listnewVocaset[j] \*\* 2) \*\* 0.5) \* (np.sum(kindListnewVocaset[i] \*\* 2) \*\* 0.5)  
 temp .append(formatM / formatZ)  
 distMat.append(temp)  
 return distMat

附件4

# 线性回归（标准方程法）  
import numpy as np  
import pandas as pd  
import matplotlib.pyplot as plt  
  
def loadDataSet():  
 fr = pd.read\_excel('C:\\Users\\王教授\\Desktop\\校赛D题\\deal.xlsx')  
 dataSet = []  
 labels1 = []  
 labels2 = []  
 # fr.loc[x][y]:x表示列索引，y表示行索引  
 # fr.loc[x] :读取x这一列  
 # 同理 fr.iloc[][] :按行读取  
 for j in range(9):  
 temp1 = []  
 temp2 = []  
 for i in range(12):  
 if i % 2 == 0:  
 temp1.append(fr.iloc[i][j])  
 else:  
 temp2.append(fr.iloc[i][j])  
 labels1.append(temp1)  
 labels2.append(temp2)  
 for i in range(6):  
 dataSet.append([1, i])  
 return dataSet, labels2[3]  
  
  
def standRegres(xArr, yArr):  
 xMat = np.mat(xArr)  
 yMat = np.mat(yArr).T # 先转换成n行一列  
 temp = (xMat.T \* xMat)  
 if np.linalg.det(temp) == 0.0: # 判断行列式是否为0  
 print("行列式为0，无法计算")  
 return  
 else:  
 w = temp.I \* xMat.T \* yMat  
 return w  
  
  
def predit(w, input):  
 return input \* w

附件5

# 贝叶斯处理多分类问题  
import numpy as np  
  
# 数据样本  
def loadDataSet():  
 fr = open('C:\\Users\\王教授\\Desktop\\校赛D题\\test2.txt')  
 dataset = []  
 label = []  
 for line in fr.readlines():  
 line = line.strip().split()  
 dataset.append(line[1:-1])  
 label.append(int(line[-1]))  
 return dataset, label  
  
  
# 获取文档中出现的不重复词表  
def createVocabList(dataset):  
 vocaset = set([]) # 用集合结构得到不重复词表  
 for document in dataset:  
 vocaset = vocaset | set(document) # 两个集合的并集  
 return list(vocaset)  
  
  
def setword(listvocaset, inputSet):  
 newVocaset = [0] \* len(listvocaset)  
 for data in inputSet:  
 if data in listvocaset:  
 newVocaset[listvocaset.index(data)] = 1 # 如果文档中的单词在列表中，则列表对应索引元素变为1  
 return newVocaset  
  
  
def train(listnewVocaset, label):  
 label = np.array(label)  
 numDocument = len(listnewVocaset) # 样本总数  
 numWord = len(listnewVocaset[0]) # 词表的大小  
 pInsult = 1.0 / 6.0  
 p1num = np.ones(numWord)  
 p2num = np.ones(numWord)  
 p3num = np.ones(numWord)  
 p4num = np.ones(numWord)  
 p5num = np.ones(numWord)  
 p6num = np.ones(numWord)  
 p1Denom = 2.0 # 拉普拉斯平滑  
 p2Denom = 2.0  
 p3Denom = 2.0  
 p4Denom = 2.0  
 p5Denom = 2.0  
 p6Denom = 2.0  
 for i in range(numDocument):  
 if label[i] == 1:  
 p1num += listnewVocaset[i]  
 p1Denom += 1  
 elif label[i] == 2:  
 p2num += listnewVocaset[i]  
 p2Denom += 1  
 elif label[i] == 3:  
 p3num += listnewVocaset[i]  
 p3Denom += 1  
 elif label[i] == 4:  
 p4num += listnewVocaset[i]  
 p4Denom += 1  
 elif label[i] == 5:  
 p5num += listnewVocaset[i]  
 p5Denom += 1  
 elif label[i] == 6:  
 p6num += listnewVocaset[i]  
 p6Denom += 1  
 # 取对数是为了防止因为小数连乘而造成向下溢出  
 p1 = np.log(p1num / p1Denom)  
 p2 = np.log(p2num / p2Denom)  
 p3 = np.log(p3num / p3Denom)  
 p4 = np.log(p4num / p4Denom)  
 p5 = np.log(p5num / p5Denom)  
 p6 = np.log(p6num / p6Denom)  
 return p1, p2, p3, p4, p5, p6, pInsult  
  
  
# 分类函数  
def classiyyNB(Inputdata, p1, p2, p3, p4, p5, p6, pInsult):  
 # 因为取对数，因此连乘操作就变成了连续相加  
 p1vec = np.sum(Inputdata \* p1) + np.log(pInsult)  
 p2vec = np.sum(Inputdata \* p2) + np.log(pInsult)  
 p3vec = np.sum(Inputdata \* p3) + np.log(pInsult)  
 p4vec = np.sum(Inputdata \* p4) + np.log(pInsult)  
 p5vec = np.sum(Inputdata \* p5) + np.log(pInsult)  
 p6vec = np.sum(Inputdata \* p6) + np.log(pInsult)  
 save = []  
 save.append(p1vec)  
 save.append(p2vec)  
 save.append(p3vec)  
 save.append(p4vec)  
 save.append(p5vec)  
 save.append(p6vec)  
 print(save)  
 t = save.index(max(save))  
 return t + 1

附件6

def loadDataSet():

fr = pd.read\_excel('C:\\Users\\王教授\\Desktop\\校赛D题\\ha.xlsx')  
 fr\_ = pd.read\_excel('C:\\Users\\王教授\\Desktop\\第四问.xlsx')  
 data = []  
 for i in range(53, 58):  
 data.append(  
 [fr.iloc[i]['电视台即将收益'], fr.iloc[i]['电视广告上一季度收视率（%）'], fr.iloc[i]['制造商上一季度产品销售量'], fr.iloc[i]['是否投放']])  
 dataMat = []  
 LabelMat = []  
 for line in data:  
 dataMat.append([line[0], line[1], line[2]])  
 if len(line[3]) == 1:  
 LabelMat.append(1)  
 else:  
 LabelMat.append(0)  
 for k in range(184, 207):  
 dataMat.append([fr\_.iloc[k]['电视台即将收益'], fr\_.iloc[k]['电视广告上一季度收视率（%）'], fr\_.iloc[k]['制造商上一季度产品销售量']])  
 # 数据归一化处理  
 dataMat = np.array(dataMat)  
 mindata = dataMat.min(0)  
 maxdata = dataMat.max(0)  
 m = dataMat.shape[0]  
 resultData = dataMat - np.tile(mindata, (m, 1))  
 resultData = resultData / np.tile(maxdata - mindata, (m, 1))  
 dataMat = []  
 for line in resultData:  
 dataMat.append([1, line[0], line[1], line[2]])  
 inputData = dataMat[5:]  
 dataMat = dataMat[0:5]  
 return dataMat, LabelMat, data, inputData  
  
  
# 辅助函数  
def signed(x):  
 return 1.0 / (1 + np.exp(-x))  
  
  
# 批处理  
def gradAscent(dataMat, LabelMat):  
 data = np.mat(dataMat)  
 labels = np.mat(LabelMat).T  
 m, n = np.shape(data)  
 weights = np.mat(np.zeros((n, 1)))  
 for i in range(500):  
 h = signed(data \* weights)  
 error = labels - h  
 weights = weights + data.T \* error  
 return weights  
  
  
# 随机梯度(每次仅用一个样本点更新回归系数)  
def stocGradAscent(dataMat, LabelMat):  
 dataMat = np.array(dataMat)  
 LabelMat = np.array(LabelMat)  
 m, n = np.shape(dataMat)  
 weights = np.ones(n)  
 for i in range(3000):  
 randIndex = np.random.randint(0, m)  
 h = signed(np.sum(dataMat[randIndex] \* weights))  
 error = LabelMat[randIndex] - h  
 weights = weights + error \* dataMat[randIndex]  
 return weights