参赛编号：	CDA02016
赛题题号：	A

医药电商销售数据的分析报告

摘　要

在疫情冲击了线下交易的 2020-2021 年，人们通过电商品牌购买药品的需求激增，再伴随着国家政策的放开，越来越多的药企参与并且重视起电商领域的布局。本次对这两年内天猫维生素类药品的销售数据的分析，从包括删除无用数据，文字信息提取，销售额计算等一系列数据预处理开始，依次分析和挖掘了不同店铺、药品和品牌的销售情况，并预测了天猫维生素类药品未来三个月的销售总额。

首先，在对店铺和药品的分析中，我们主要使用了分组描述性统计和数据可视化，并配以必要的假设检验和文字挖掘。在参与销售的 24 个店铺中，销售额占比最高的 4 各店铺都是与淘宝官方相关的本土或者海外店铺，其中占比最高的阿里健康大药房售卖较好的药品大多是医学上作证的必要补剂，例如孕妇需要补充的维生素 D。

在过万种药品中，销量前十的药品在总销量中占比从 0.8%到 2.4%不等，他们来自 7 各不同的品牌和 4 个不同的的店铺。从不同药品的销售额随时间变化的曲线中，我们可以推断保持日常销售额稳定与在大促期间取得销售额的快速增长存在着一定相互促进的作用。

然后，在对品牌的分析中，我们找到了 516 个品牌中总销量前十的品牌，其中澳洲品牌 swisse 在这两年间一直处于销售额的第一名。为了探究销售额占比最高的十个店铺销售较好的原因，我们还对比了在总数据集和只含高销量品牌的数据集中各变量与销售额的互信息(MI)，我们发现与总体情况不同，高销售额的品牌的药品销量比单价对销售额的影响更大。

最后，我们算出了每月的总销售额的时间序列并对其使用 ARIMA 模型建模和预测。在对数列经行局部加权回归(SLT)分解之后，我们发现下该序列展现出了正趋势和季节性；为了得到平稳非白噪音序列，我们对原序列的一阶差分序列进行了平稳性检验和白噪音检验；接着，我们发现一阶差分序列自相关一阶截尾，偏自相关拖尾，所以使用了 ARIMA(0,1,1)和有着最小 AIC 的 ARIMA(2,1,3)分别建模，并使用 RMSE 较小的 ARIMA(0,1,1)预测出 2022 年前三个月天猫上维生素药品的总销量分别约为 8.45 千万、8.94 千万和 9.45 千万。

关键词：数据可视化　假设检验　文本挖掘　互信息　时间序列　ARIMA 模型

目 录

1. 背景与挖掘目标

在商品电子化和疫情冲击了线下交易的大背景下，人们越来越多的选择在电商平台购买药品。随着需求的激增和政策的放宽，各大药企为了获得更多的盈利也开始更加重视线上交易的运营。本次数据挖掘将多角度探索那些可能影响销售额的因素，以此助力他们找到顺应时代潮流，立足市场，扩大份额的可行方案。

2. 数据预处理

在导入了基本的模块和进行全局设置之后，我们读取了保存到本地的维生素销售数据文件并查看了其基本信息。

```
df = pd.read_excel("data.xlsx") #读取信息

df.dtypes #数据类型

df.shape #数据大小

df.isnull().sum(axis=0) #空值情况

df.duplicated().sum(axis=0) #重复情况
```

本数据集的大小为 75110*10，其中没有完全重复的数据。每条记录依次包含了月份(date_time)、id、店铺名(shop_name)、商品标题(title)、货存单位(sku_name)、商品单价(price)、商品销量(sold)、商品折扣(discount)、商品品牌(brand)和商品参数(parameter)，其中除了月份为时间性数据，id 和销量为整数型，单间为连续性，其余列都是字符串形式储存的数据。空值的分布情况可以被可视化为下图，其中白色代表空值：

```
import missingno as msno

msno.matrix(df)
```

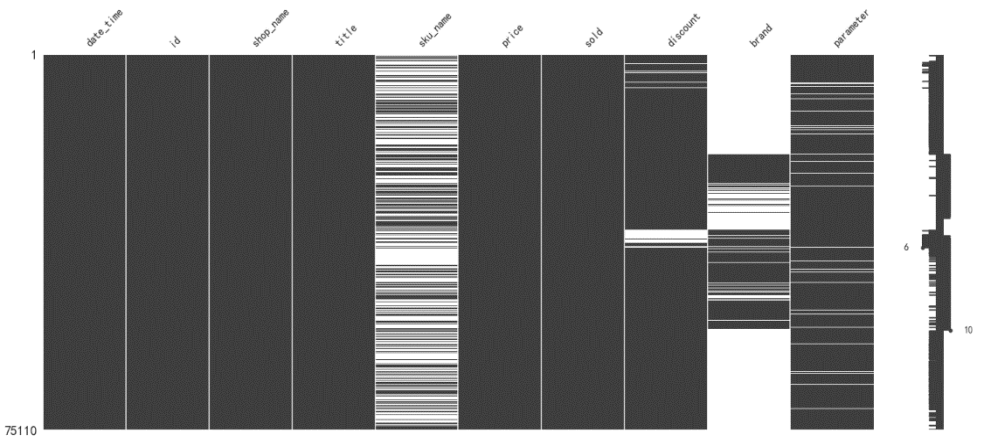

图 1：初始数据空值情况

结合空值情况和各列数据格式，我们依次做出了以下处理:

- 删除 id 列
- 从 discount 列中提取打折的具体数值，例如，从“9.5 折中提取“9”。如果是空值，就当作不打折处理。然后再将单价乘上销量折扣，算出单条数据的销售额
- 将 parameter 列的各行以“||”为分割标记分开，并分别提取被开的各列中的有用信息，其中 parameter 中缺失的品牌信息会用 brand 列来填补
- 品牌中存在不少使用不同名字的品牌（例如，’汤臣倍健’和’BY－HEALTH/汤臣倍健’），需要找到全部并且统一为一个品牌名
- 从 title 和 sku_name 中找到有用的文字信息，并从中填补缺失信息和提取适用性别，是复合多种维生素，和有无名人推荐/同款等信息
- 删除一条单价为 4266.25 的异常数据，因其余单价都在 2000 以下
- 删除处理完毕并且无更多用处的原特征列

所使用到全部代码详见附录，其中在处理 parameter 列时，我们首先通过以下代码找到了可能含有的信息：

```
content = []
for i in range(len(df.parameter)):
    splitted = str(df.parameter[i]).split('||') #一条记录所含的 content
    content.extend( [splitted[j].strip().split(':')[0] for j in range(len(splitted))] )
content = list(set(content))#去重
print(content)
```

然后我们再将可能信息（content）总结为以下四个信息类别：

- 品类：’药品名称’，’品名’，’药品商品名’，’产品名称’，’药品通用名’，’系列’
- 品牌：’品牌’（加上本来就有的 brand column）
- 厂商：’厂名’，’生产企业’
- 规格：’药品规格’，’规格’

从而，我们定义了 find_information 函数来找到各行的以上四个信息类别并存储到四个不同的列中。

在从 title 和 sku_name 中找到有用的信息时，我们先把他们合并为一列，然后使用 lambda x: re.sub('[^\u4E00-\u9FD5,a-z,A-Z./*0-9]+','',x) 删除无用信息，最后通过定义 pattern 和 lambda x: re.findall(pattern,x) 找到所需的信息。

经过以上的预处理步骤，我们得到了最终的数据集并打印前五条数据。其中，‘性别’列可以取值为通用、男性或者女性，‘复合’为 1 代表是复合维生素，‘推荐/同款’为 1 表示标题中含有名人或者直播推荐的信息。虽然 SKU、品类、品牌、厂商和规格列均含有空值，但是我们不将空值行或者列删除已备查看具体某商品某月的销售情况。

	日期	店铺	标题	SKU	单价	销量	折扣	销售额	品类	品牌	厂商	规格	性别	复合	推荐/同款
0	2020-11-01	宜度海外专营店	特价处理！超低价19.9元起！含维生素ABCDE B族B12 维生素锌硒片	维生素e 300粒--买3送1	60.900000	19	9.0	1041.390	综合系列	Puritan's Pride/普丽普莱	Puritan's Pride	None	通用	1	0
1	2020-11-01	天猫国际进口超市	FANCL男40岁综合营养包30包/袋*2复合多种维生素日本芳珂fancl	(【直营】日本FANCL/芳珂 40岁男性综合维生素营养 30袋（30天量）)*2	500.000000	71	9.0	31950.000	综合营养包	FANCL	FANCL株式会社	30包/袋	男性	1	0
2	2021-12-01	阿里健康大药房	民生21金维他多维元素片30片儿童男女成人补充维生素矿物质缺乏	国民大牌| 补充21种维矿	22.416667	366	9.5	7794.275	21金维他 多维元素片（21）30片*1瓶/盒	21金维他	杭州赛诺菲民生健康药业有限公司	30片*1瓶/盒	通用	1	0
3	2021-12-01	天猫国际进口超市	【直营】美国进口安利简装倍立健多种复合维生素矿物质营养186片	NaN	452.250000	28	9.5	12029.850	倍立健	AMWAY/安利	Amway	None	通用	1	0
4	2021-03-01	阿里健康大药房	买2送礼】elevit/爱乐维复合维生素片140粒维生素缺乏症贫血官方	爱乐维玛咖锌淫羊藿胶囊140粒+爱乐维 复合维生素片30片/盒	371.666667	32	9.0	10704.000	复合维生素片	elevit/爱乐维	拜耳医药保健有限公司	30片/盒	通用	1	0

图 2：预处理结束后的数据前五条

2. 店铺分析

2. 1 总体情况

在数据处理完成之后，考虑到药企在将商品上架到电商时需要选择店铺，我们首先对数据中所含的店铺进行了分析。通过 len(df.shop_name.unique()) 得到一共有 26 个店铺，我们接着对这 26 个店铺的销售额进行了分组求和和降序排序得到所有店铺在 2020 和 2021 年的两年总销售额。

```
#计算每个店铺的销售额和销售占比

shops=pd.DataFrame(df.groupby(by='shop_name')['sale_total'].sum()).sort_values('sale_total',ascending=False).reset_index() #降序排序

shops['proportion'] = shops['sale_total']/sum(shops['sale_total']) #销售占比

shops.head(10)
```

	shop_name	sale_total	proportion
0	阿里健康大药房	6.751027e+08	0.449943
1	天猫国际进口超市	3.561012e+08	0.237335
2	天猫超市	1.148217e+08	0.076527
3	阿里健康大药房海外店	1.065089e+08	0.070986
4	康恩贝官方旗舰店	9.559872e+07	0.063715
5	ChemistWarehouse海外旗舰店	5.539680e+07	0.036921
6	天猫国际妙颜社	3.569804e+07	0.023792
7	苏宁易购官方旗舰店	1.065959e+07	0.007104
8	thejamy保健海外专营店	9.798825e+06	0.006531
9	skyshop海外专营店	6.506159e+06	0.004336

图 3：总销售额最高的 10 个店铺

将上表化成饼图：

```
data = list(shops.sale_total[0:5])

data.append(shops.sale_total.sum()-shops.sale_total[0:5].sum())

labels = list(shops.shop_name[0:5]) .append('其他')

colors = sns.color_palette('pastel')[0:6]

#创建饼图

plt.pie(data, labels = labels, colors = colors, autopct='%.0f%%')
```

```
plt.rcParams['figure.figsize'] = (20.0, 8.0)
plt.legend(loc ='upper left',bbox_to_anchor=(0.9,0.9))
plt.title('各店铺总销售额占比')
plt.show()
```

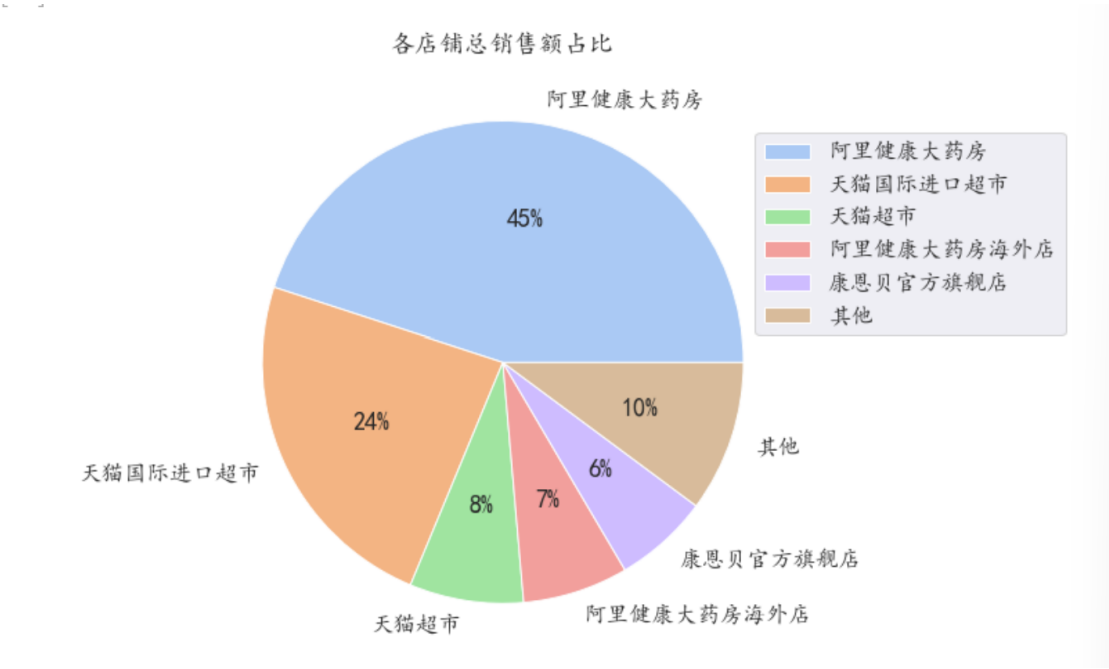

图 4：各店铺销售额占比

我们可以看到这两年总销量前四中的店铺：阿里健康大药房的官方店铺和海外官方店铺，天猫超市以及其国际版，都是与淘宝官方（阿里巴巴集团）相关的店铺，并且他们的总销售额都超过了 1 亿。

造成这个现象的原因一方面在于人们对于官方认证和售卖的商品有更高的信任，在相似商品相似价格的情况下，更倾向于选择在阿里官方店铺购买商品，并且我们可以猜测人们在购买国内药品时会首先考虑阿里健康大药房，在购买进口药品时会首选天猫国际进口超市。另一方面，因为官方店被信任和销量大，他们在吸引商户和定价商讨时有天然的优势，所以这些官方店售卖的药品种类全，价格实惠。此外，还有一个电商特有因素—用户搜索关键词后推荐的顺序和首页广告的投放，因为官方店的盈利与整体集团相关度更大，所以淘宝或天猫 APP 的推荐系统对官方店的商品很有可能存在优先推荐的情况，这进一步扩大了潜在客户，因为人们只有知道商品的存在才可能购买，所以这也是官方店不可忽视的优势所在。

剩下的 22 个店铺总共只占有了 16%的销售额，其中康恩贝旗舰店占有了 6%的份额与天猫超市和阿里健康大药房海外店销售额极为接近。该店铺只售卖该品牌的商品却能拥有如此高的销量不仅仅是因为其在中国售卖多年，人们对其认知度和认可度较高，更因为在商品有保障的前提下，该店仍有很大的价格优势。与官方店店比，旗舰店的商品不需要给店铺佣金并且拥有更自由地定价权，特别是在大

促 1 时，可以通过更大的活动力度来吸引更多的购买者。与其他品牌店比，康恩贝作为大企业，不仅研发生产线完备，营销和广告也更成熟，低廉的成本和高效的生产让其可以走薄利多销的路线。

站在药企的角度，如果是并未发展成熟的小企业，那么入驻各个官方店铺是享受平台巨大用户群体的便利最快速的方式之一，如果有条件能够达成广告投放的协议，那也是非常值得投入的前期成本。如果是像康恩贝一样被人们熟知的大企业，那在投放商品在官方店的基础上，也应该着手运营自己的官方旗舰店并通过降低定价等方式吸引消费者。

2.2 销量额占比最高的店铺

我们接下来对销量最多的店铺（阿里健康大药房）进行更详细的分析。首先，提取所有店铺名为“阿里健康大药房”的数据并命名为 data_1：`data_1 = df[df.shop_name == '阿里健康大药房'].drop(['shop_name'],axis=1).reset_index(drop=True)` 。然后，用 `data_1.describe()`对数值列计算了描述性统计值计算：

	price	sold	discount_number	sale_total	复合	推荐/同款
count	16958.000000	16958.000000	16958.000000	1.695800e+04	16958.000000	16958.000000
mean	92.668085	551.286296	9.339663	3.981028e+04	0.165880	0.000354
std	91.857052	2169.318065	0.301560	2.045990e+05	0.371984	0.018807
min	2.900000	0.000000	8.500000	0.000000e+00	0.000000	0.000000
25%	28.900000	15.000000	9.000000	8.660475e+02	0.000000	0.000000
50%	69.000000	72.000000	9.500000	3.648879e+03	0.000000	0.000000
75%	124.000000	339.000000	9.500000	1.763485e+04	0.000000	0.000000
max	1123.060000	103940.000000	10.000000	9.978240e+06	1.000000	1.000000

图 5：阿里健康大药房单价、销量、折扣、销售额、复合和推荐/同款的描述性统计

从表格中，我们可以发现单价虽然从 2.9 到 1123.06 不等，但中间 50%的单价都集中在[28.9，124]的区间并且均值比中间值要大超过 20，所以单价数据的分布是正偏态/右偏态的。相似地，销量和总销售额也是右偏态分布的，如果我们想更清楚地可视化或者在模型中加入这三个变量中的一个，我们最好对其取对数。

阿里健康大药房的商品在这两年的平均折扣为 9.3 折并且变化波动的范围不大。有约 16.6%的商品在标题中标明为复合维生素，但是否复合对销售额的影响还需进一步探究。只有不到 0.01%的商品写明了某某推荐或同款，这很符合官方店铺并不需要名人或者直播带货的属性，因此我们不再继续研究该变量。

在 2020-2021 年，一共有 192 个品牌在阿里健康大药房销售维生素类药品，虽然其中星鲨（6.5%）、爱乐维（5.9%）、伊可新（4.0%）、善存（3.1%）和养生堂（2.4%）为销量前五的品牌，但并没有一个品牌在该店铺拥有垄断性地位。其中，星鲨主要售卖的品类为维生素 E 软胶囊和维生素 D 滴剂，爱乐维主营复合维生素片，而伊可新只销售维生素 AD 滴剂这一个品类。虽然他们并没有在维生素类垄断销售额，但在细分平类后，做到了店铺内的类目第一和总销量领先者，这对入驻官方

店的企业有着一定的启示意义：比起售卖各种不同类别的维生素，在这种有不同品牌入驻的店铺，集中精力扩大自身优势产品的销量或许为更好的选择。

为探究阿里健康大药房销售情况随时间的变化趋势，我们画出了随月份销售额和销量的变化曲线图：

```
#销售额和销售量随月份变化图
data_1_monthly = pd.DataFrame(data_1.groupby('date_time')[['sale_total','sold']].sum())
data_1_monthly.plot(subplots=True)
```

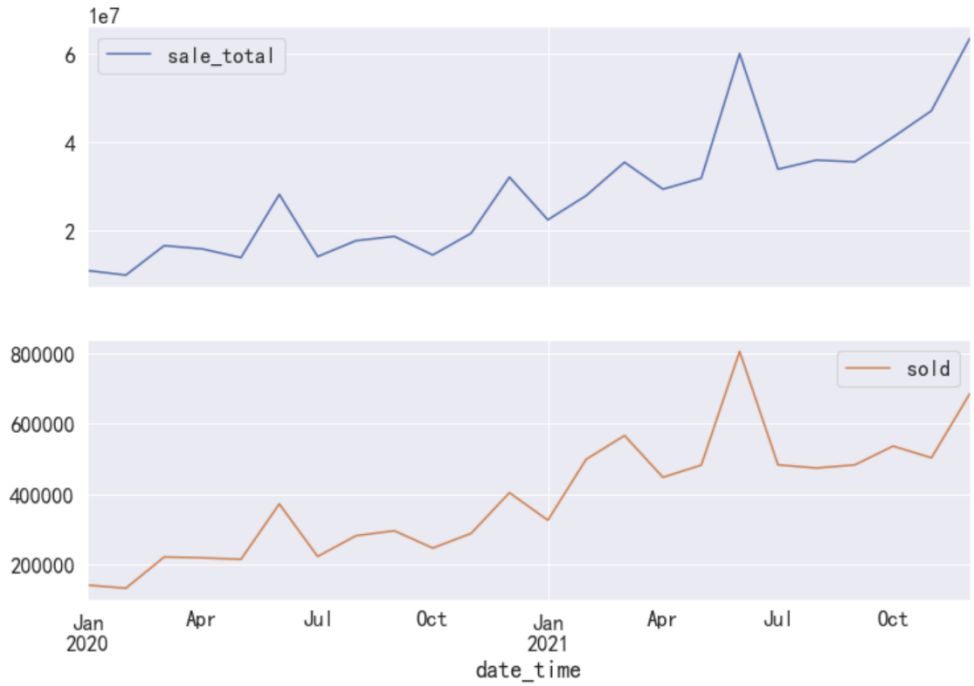

图 6：阿里健康大药房总销售额和销量随时间变化图

从图 6 中我们可以发现月总销量与总销售额的变化趋势基本一致，证明在这两年中，阿里健康大药房的平均定价与折扣并没有非常显著的波动。除此之外，我们可以看到在波动增长的过程中，月总销量和月总销售额在 2021 年六月都急速增长了一次并在七月回到与之前差不多的水平。我们可以猜测此现象与“618”购物节相关，人们会在购物节“囤”维生素，导致了后面接着两三个月销量与五月相比上升较少。相似的波动在 2020 年也发生了，但 2020 年 6 月的销量甚至都没有 2021 平均的高，这也证实了越来越多的人会选择在电商平台购买维生素，不论是隔离政策迫使或者出于自身意愿。

接着，我们对比了该店铺在 2021-06 销量前十和 2 年总销量前十的产品并发现差别不大。不论是在大促或者平时，针对孕妇或者婴幼儿补充维生素 D 的产品销量都很高，这是由该药品本身的不可缺失性导致的。孕妇是非常容易缺维生素 D 的人群，而孕期维生素 D 缺乏可影响胎儿的骨骼发育，并可导致新生儿的低钙血症及牙齿发育缺陷，所以许多医生都会建议服用[1]。维 A 酸乳膏的高销量也是相同的原理，该商品是治疗面部痤疮的所会使用的药物。由此我们可以发现，大量的

消费者选择在阿里健康大药房购买“不可或缺”的维生素类药品而不是“锦上添花”的保健类产品，这不仅反映了大部分人的需求也体现出人们对此官方店铺的信任。

在探究了总体的购买情况之后，我们尝试了分析不同受众性别，是否为复合和折扣水平对销量分布的影响。首先，我们分别画出不同性别和复合形况下取对数后总销量的分布。

```
data_1['log_sale'] = np.log(data_1['sale_total'])

fig,axes=plt.subplots(1,3,figsize = (16,6))

for i, sex in enumerate(data_1['性别'].unique()):

    sns.histplot(data_1[data_1['性别']==sex], x='log_sale', hue="复合", kde=True,ax = axes[i]).set(title=sex)

    axes[i].set(xlim=(2, 17))
```

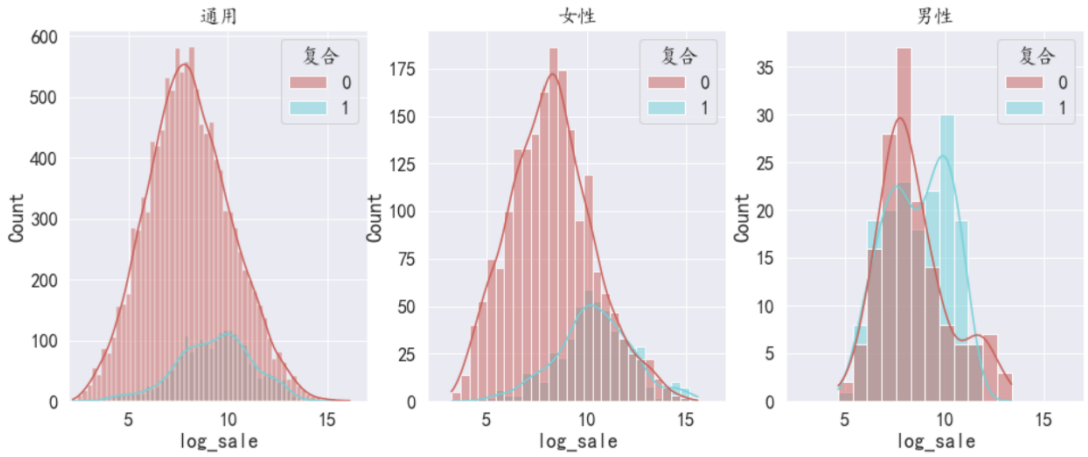

图 7: 不同性别和复合形况下取对数后总销量的分布

对比三张图的纵坐标，我们可以发现阿里健康大药房绝大多数的维生素药品都是男性女性都通用的，其次是针对女性的，最后是针对男性的。对比不同颜色的分布图高度，我们发现除了目标客户是男性的药品，非复合维生素商品在数量上都超出了复合维生素三倍。

如果我们观察分布的集中区间，就可以发现不论针对的性别群体是什么，复合维生素的集中分布的区间的销售额都比非符合产品的区间的要高。因为这是取对数后的销量，原销量数值在不同复合情况下聚集区间区别应更大。为了证实这个想法，我们对不同性别下复合和非复合维生素的分布进行了原假设为二者分布相同的 two-sample Kolmogorov-Smirnov 检验，所得到的 p 值都小于 0.05，因此我们有显著证据说复合产品的单月销售额会比非复合的高。

相似的，我们画出不同性别和折扣比例下取对数后总销量的分布。

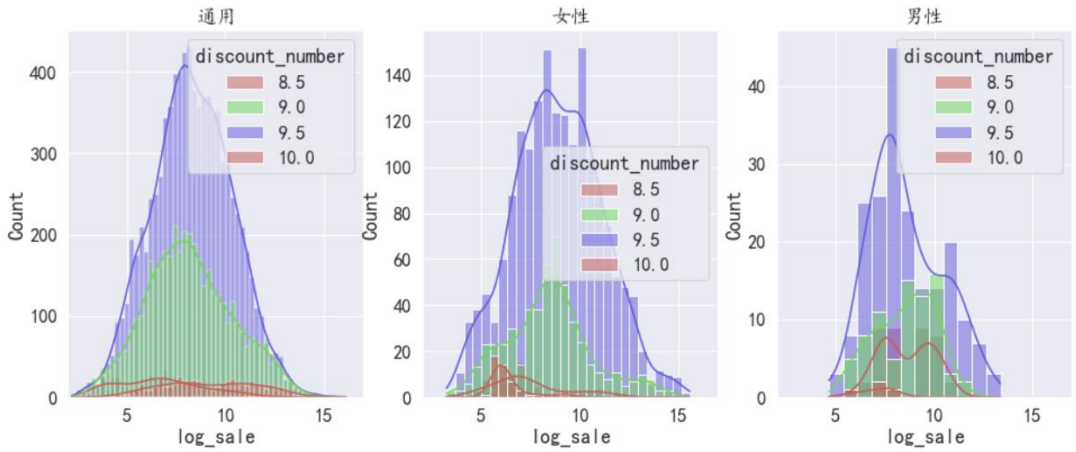

图 8: 不同性别和折扣比例下取对数后总销量的分布

不论是通用维生素，还是单性别维生素，阿里健康大药房最多使用的折扣都是 9.5 折，其次是 9 折。我们暂时很难推断出更多有效的信息，虽然男性和女性维生素在不同折扣下都存在双峰分布的情况，但由于统计数据量不足通用维生素，我们有理由猜测如果数据量足够大，这些双峰分布可能会趋于单峰。当药企选择入驻该店铺时产品的价格时，可以预先考虑到折扣情况以保障自身盈利。

3. 药品分析

3.1 总体情况

除了店铺选择对经营电商药品有很大的影响,经营何种药品也是不可忽视的重要决策之一。虽然 len(df.title.unique()) 算出有 11258 种药品，但我们可以发现其中的很多都高度相似，他们只是标题表述或规格稍有差异，却有着同样的厂商和品牌。实际药品的数量肯定是远小于 11258 个的，但我们还是以商品链接为单位定义药品数量并进行后续的研究。

接着，我们统计了药品数量随着时间的变化情况，我们发现药品从第一个月（2020-01）的 1882 种逐步增加到了最后一个月（2021-12）的 2940 种，其增长过程随存在小幅度的波动，但总体上是平缓而稳定的。

3.2 销售额占比最高的 10 个药品

与对店铺的分析相似，我们先算出每个药品在这两年内的总销量和销量占比，在降序排序找到销量前十的药品。

```
products=pd.DataFrame(df.groupby(by='title')['sale_total'].sum()).sort_values('sale_total'
,ascending=False).reset_index()

products['proportion'] = products['sale_total']/sum(products['sale_total'])

products.head(10)
```

	title	sale_total	proportion
0	领券减】爱乐维复合维生素100片孕妇叶酸补充备孕早期多维元素B	3.569218e+07	0.023788
1	3盒】星鲨维生素D滴剂(胶囊型)24粒补钙维生素D佝偻病官方婴儿钙	2.836576e+07	0.018905
2	Swisse 斯维诗澳洲进口钙片维生素D150片 孕妇VD钙 成人老年补钙	2.248593e+07	0.014987
3	3盒包邮】达因伊可新维生素AD滴剂30粒1岁以上AD预防佝偻病ad滴剂	1.622559e+07	0.010814
4	3盒】星鲨维生素D滴剂(胶囊型)24粒补钙维生素D3佝偻病官方婴儿钙	1.612723e+07	0.010749
5	21金维他多维元素片100片复合维生素B b6 b1 b2维生素c钙铁锌21	1.508954e+07	0.010057
6	达因伊可新维生素AD滴剂50粒0~1岁维生素儿童婴幼儿婴儿钙ad滴剂	1.443055e+07	0.009618
7	100片康恩贝维生素C咀嚼片vc片维他命c含片成人男女高含量维c搭ve	1.396282e+07	0.009306
8	【3盒装】星鲨维生素D滴剂(胶囊型)24粒补钙维生素D 佝偻病	1.279443e+07	0.008527
9	惠氏钙尔奇钙片60片碳酸钙D3孕妇成人补钙维生素D3骨质疏松药店	1.211564e+07	0.008075

图 9：总销量最高的 10 个药品

我们可以看到销量前十的维生素类药品都达到了千万级别的销量，在总销量中占比从 0.8%到 2.4%不等。我们从总数据集中找到与这是个商品相关的记录 data_2 = df[df.title.isin(list(products.head(10).title))]，然后通过找到店铺和品牌的独特值，发

现了这十个商品来自康恩贝官方旗舰店， 阿里健康大药房， 天猫国际进口超市和阿里健康大药房海外店。在这十个商品的七个品牌中，星鲨、爱乐维、伊可新是阿里健康大药房销量前三的品牌，康恩贝是本品牌旗舰店售卖，其余三个来自进口官方店。这十个商品只有 21 金维他的多种维生素复合片在其中，所以该商品是多种维生素复合片的销量第一名。相似的，康恩贝 VC 咀嚼片也是其品类的销售第一名。如果有销售相似商品的药企，那么该药企在借鉴这些成功商品的经验的同时，更找到相应品牌所做的不够的地方并在自身经营的过程中完善这些缺陷，才能在更好的获得抢占市场份额。

为了更清楚找到前十商品标题中的高频关键词，我们对所有的标题进行了分词 jieba.lcut ， 去除停用词 lambda x: [i for i in x if i not in stop] 和绘制图云 WordCloud.fit_words：

图 10：销量前十药品的标题词云

除了品牌名称之外，我们可看到“孕妇”、“婴幼儿”、”钙”等此类关键词出现的非常频繁，这意味着这类“刚需”药物在销量前十中占比较大，但是不及它们在阿里健康大药房的销售额占比，因为还存在许多别的商品的关键词。

接着，我们将每个药品每月的销售额相加(data_2_monthly)并绘制销售额占比前十的药品每月销售曲线图。

```
sns.lineplot(x="date_time", y="sale_total",hue='title',  data=data_2_monthly)
plt.legend(loc ='upper left',bbox_to_anchor=(1,0.9))
plt.show()
```

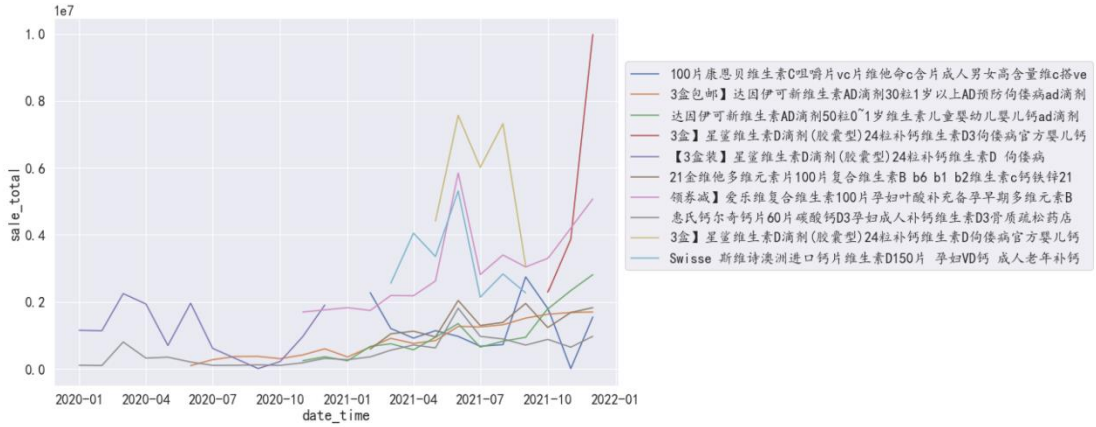

图 11：销量前十的药品随时间的销量变化

从上图我们可以看到，星鲨的三个维生素 D 滴剂分别出现在了 2020 年 1 月到 12 月，2021 年 5 月到 9 月和 2021 年的 10 月到 12 月，我们可以猜测这是商家下架了其中一个商品链接又重新再上架了一个，他们本质上是同一款产品。与之不同的是伊可新的 AD 滴剂，在同一时间段出现了两条不同的线，仔细观察标题我们可以发现这两款商品的规格不同。为了更好的展现真正的产品走势图，我们画出了该销量前十药品的品牌销量随时间的走势图 sns.lineplot(x="date_time", y="sale_total",hue=品牌', data=data_2_monthly)。

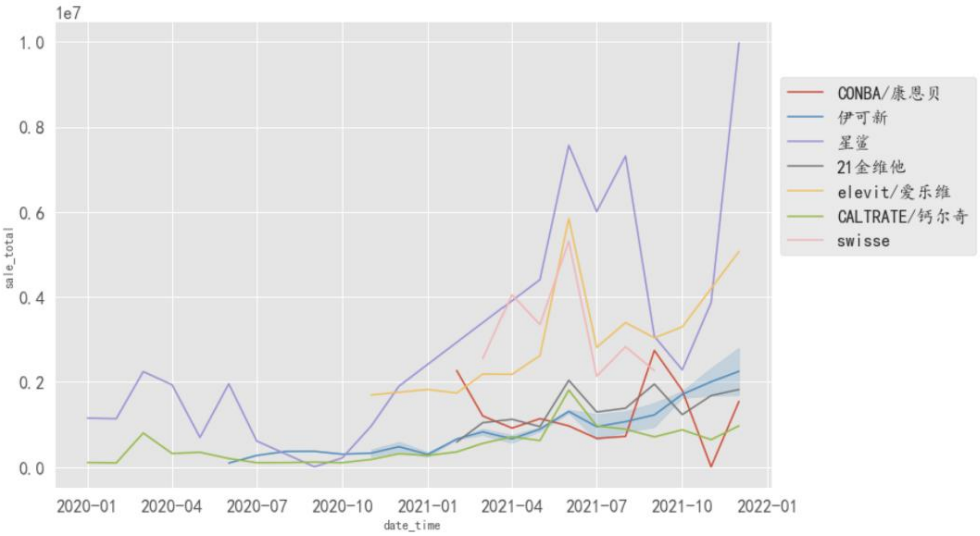

图 12：销量前十的商品的品牌销量随时间的变化

图 12 证实了我们在上面的猜想，其中，星鲨只有一条线，而伊可新的变化曲线附近有置信曲线，而区间内的线是这两款不同规格商品的平均销售额变化。

分别观察每条曲线的变化趋势，我们可以推测星鲨的维生素 D 滴剂在大部分时间里都是天猫上销售额最高的维生素类药品。康恩贝、swisse、爱乐维和 21 金维他的“销冠”产品都是从 2020 年底到 2021 年初才开始售卖的产品，从侧面体现出了消费者 21 年显著增加了在电商购买药品的量，否则只销售了一年的产品不

会在总销售额中占比前十。

与钙尔奇的 D3 片和伊可新的 AD 滴剂一直保持着较为稳定的销量所不同的是，卖的最好的三款商品在“618”的销售额增长幅度非常显著。虽然我们无法判断是平时的销售额导致购物节的高增长更多，还是反过来购物节对日常销售促进更大，但是我们可以推测这二者之间存在着相互促进的作用。如果在平时保持稳定的销售渠道和客户群体，那大促就可以花更多精力在保证库存充足和物流通畅上。如果在大促上取得客观的销售额，那不仅可能给品牌带来了许多新的消费者并且扩大知名度，还可能带动整个药企的运营和管理升级。

4. 品牌分析

4.1 总体情况

在分析了 2020-2021 天猫上销售维生素类药品的店铺和单个药品的销售情况之后，我们又对多有的药品品牌进行了分析。经过预处理中品牌名称的统一，我们用 len(df['品牌'].unique())算出在这两年的时间段中，天猫上一共有 516 个品牌参与了维生素类药品的销售。

接着，我们统计了品牌数量随着时间的变化情况，我们发现其数量从第一个月（2020-01）的 221 个逐步增加到了最后一个月（2021-12）的 340 个，其增长过程随存在小幅度的波动，但总体上是平缓而稳定的。

4.2 销售额占比最高的 10 个品牌

与前面的分析相似，我们先算出每个品牌在这两年内的总销量和销量占比，在降序排序找到销量前十的品牌。

```
brands=pd.DataFrame(df.groupby(by='品牌')['sale_total'].sum()).sort_values('sale_total',
ascending=False).reset_index()
brands['proportion'] = brands['sale_total']/sum(brands['sale_total'])
brands.head(10)
```

	品牌	sale_total	proportion
0	swisse	2.666612e+08	0.178731
1	CONBA/康恩贝	1.231655e+08	0.082552
2	CENTRUM/善存	1.131437e+08	0.075835
3	elevit/爱乐维	1.008569e+08	0.067600
4	星鲨	9.753611e+07	0.065374
5	汤臣倍健	6.560236e+07	0.043970
6	养生堂	6.017866e+07	0.040335
7	伊可新	5.991355e+07	0.040157
8	FANCL	5.092043e+07	0.034130
9	朗迪	3.281707e+07	0.021996

图 13：销售额前十的品牌和销售额占比

总销量占比最高的品牌是来自澳大利亚的 swisse，并且该品牌的占比(17.9%)比第二名(8.3%)的两倍都要多。从 swisse 的官网我们可以得知，该品牌建立于 1960 年代，并在 90 年代推出分别针对男性和女性的复合维生素，获得了广泛好评。从 2016 年上年开始，swisse 官方进驻了中国各大电商品牌，接着又在 2017 年陆续进驻到了线下[2]。单从其发展历史，我们可以发现很多在国内发展较好的外

来品牌的影子：他有着悠久且优秀的品牌故事，前沿的产品理念，并且抓住了国人对外来商品高度信任的年份快速地获得了知名度。他在 2020 和 2021 年的高销量如果没有之前铺下的基础，是不可能达到的。我们可以猜测销量第三名来自美国的善存和第九名来自日本的 FANCL 的成功都或多或少有相似的因素的推动。

销量最高的国产品牌是康恩贝，其成功的原因我们在分析为何其店铺销量是销量最高的品牌旗舰店是已经分析过了，在这里就不赘述了。

接着我们画出这十个品牌销售额随月份的变化趋势图，以便观察是否有品牌有着不同的销售额曲线。

图 14：销售额前十的品牌的销售额随时间变化

从图 14 中，我们可以看到这十个品牌都是从 2020 年 1 月到 2021 年 12 月都有在售的产品的，并且 swisse 一直都是销量第一的品牌。所有的品牌的销量都在波动中上涨，只是涨幅和波动的幅度不同。

为了探究为什么为何这十个品牌销售较好，我们首先定义了只含这十个品牌的数据集 `data_3 = df[df.品牌.isin(list(brands.head(10).品牌))]` 。然后，计算了只含这十个品牌的数据中的销售额与别的特征的互信息（mutual information，MI）。在概率论和信息论中，两个随机变量的互信息表示两个变量 X 与 Y 是否有关系，以及关系的强弱。它除了可以度量相关系数可以度量的线性关系，还可捕捉更复杂的关系，比如平方关系。因为厂商、SKU 和规格在 `data_3` 中的缺失值太多，我们没有考虑这 3 个变量。

```
from sklearn.feature_selection import mutual_info_regression

def make_mi_scores(X, y, discrete_features): #计算 MI

    mi_scores = mutual_info_regression(X, y, discrete_features=discrete_features)

    mi_scores = pd.Series(mi_scores, name="MI Scores", index=X.columns)

    mi_scores = mi_scores.sort_values(ascending=False)
```

```
    return mi_scores

X = df.copy().drop(['date_time'],axis=1)

y = X.pop('sale_total')

for colname in X.select_dtypes("object"):

    X[colname], _ = X[colname].factorize() #分类变量编码

discrete_features = X.dtypes == int

mi_scores = make_mi_scores(X, y, discrete_features)
```

表 1：只含前十品牌的数据和元数据的 MI 值

特征	只含前十品	原数据集
销量	1.499033	1.439596
单价	1.158628	1.714627
标题	1.010450	0.919847
品类	0.856038	0.784820
店铺	0.657173	0.595289

表一中包含了与销售额的互信息前五大的特征在 data_3 以及原始数据中的互信息大小。由于总销售额就是由销量乘以单价再乘以折扣算出来的，因此销量和单价与销售额的 MI 值较大是复合常理的。其中，在原数据集，单价比销量对销售额的影响更大；然而，在只含有销售额前十的品牌的数据集中，单价对销售额的影响显著下降了。事实上，原数据中，与销售额互信息第五大的是品牌（MI=0.640798），而并非店铺，所以我们可以猜测消费者对于这些本身就广为人知的牌子的商品的定价更为宽容，高价和低价对于销量的影响更少。

剩余的药品（标题）、药品种类（品类）和店铺都与这十个牌子药品的销量之间有着或强或弱的关系，并且这些关系都比在原数据中要强。药品和药品平类在这十个品牌内与销量关系更大除了企业科研和生产的规划以及平台推荐算法的设计，还有可能是处于人们对牌子和其代表商品之间的捆绑信任关系导致的，就像我们之前说过的 swisse 靠复合维生素起家，星鲨 c 长时间主要售卖维生素 D 滴剂等情况一样，人们认识到这些牌子的“代表作”后，在需要相应药品时，会主动搜索对应的品牌。

至于不同品牌在不同店铺的售卖情况，我们品牌和店铺作为对销量分组求和的二位标准 data_3.groupby(['品牌','shop_name'])['sale_total'].sum()，查看了不同品牌在不同店铺的销售情况。我们发现除了朗迪只在阿里健康大药房售卖之外，别的品牌都至少在 3 个店铺有销售自己的药品，其中 swisse 有 12 个天猫店铺在售卖且在天猫国际进口超市销售额最高，而善存有 14 个且在阿里健康大药房的销售额最高。这对想要入驻中国市场的外国药企/保健品企业有着一定启示作用：建立本品牌的官方旗舰店大多时候并不如扩大销路，在官方店上架和推广商品有效。

除了其他变量与销售额之间的关系之外，我们还探究了所有变量之间是否存在线性关系。使用分类变量编码后的数据，用 corr()算出来每个变量之间的相关系数并用 sns.heatmap()可视化相关系数矩阵。

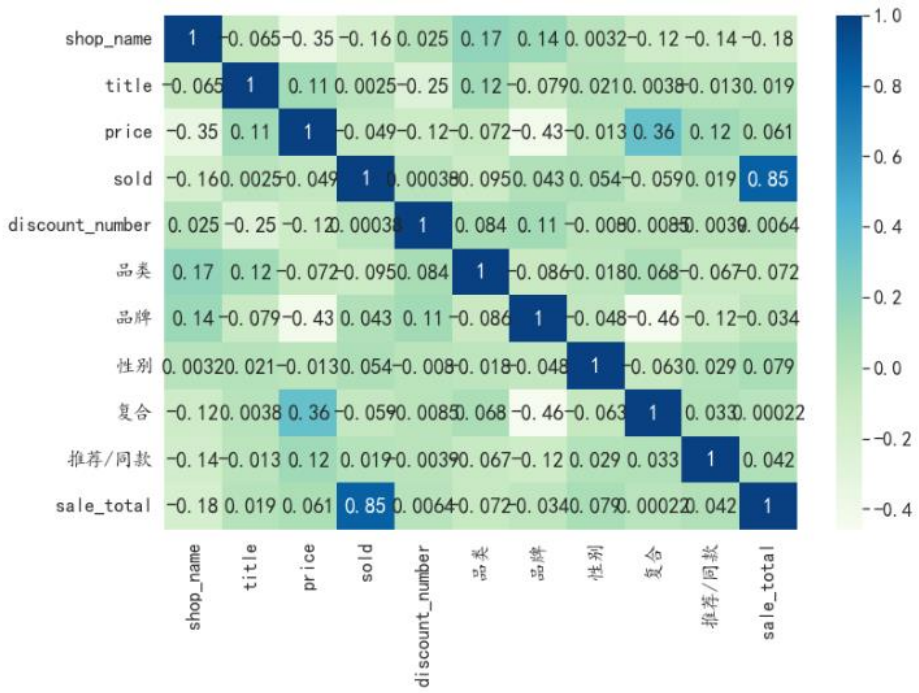

图 15：销量前十的品牌的各变量间的相关系数

从图中我们可以看到除了销量和销售额的线性相关度较高之外，不存在相关度很高的变量，所以我们暂时不轻易猜测各个变量间的交互作用。

最后，为了更加了解这些销总额高的品牌的定价和销量分布，我们画出了每个品牌定价和销量取对数后的箱型图（sns.catplot）。

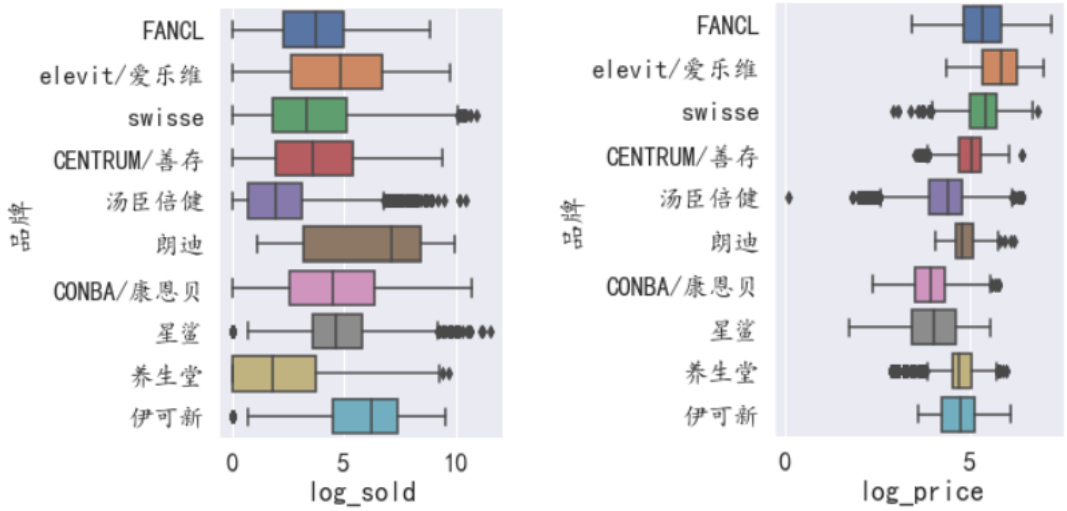

图 16：销售额最高的 10 个品牌取对数后的销量（左）和单价（右）

我们从取对数后药品单价的分布为出发点来分析。首先，我们可以看到 FANCL、

爱乐维和 swisse 的定价区间的集中区域要高于剩下的 7 个品牌，同时他们销量集中的区间是较为低的，因此我们可以猜到他们走的是“精品”或是“高端”路线。他们都是进口品牌，拥有良好的技术背书和企业形象，也会经常营销和广告自己的产品，所以成本会较高，在借着人们“要买就买最好的”和“买大品牌买安心”等心理，顺理成章地将定价提升。与之相反的是，像康恩贝和星鲨这样低价高量的国产品牌。

当然，品牌想要成功并不是只有这两种选项。比如，汤成倍健在这两年中一共售卖了 9014 种商品，排除重复的商品链接和规格修改的情况，其药品品种也也远远多于别的品牌，并且从上图中我们也可以发现该品牌的商品定价和销量的取值区间跨度非常的广，虽不排除“贴牌”销售的情况，但这更可是该品牌希望产品能覆盖到所有可能的客户群体所导致的。不论年龄阶段、销售水平、健康需求是什么，汤成倍健都为对应的消费群体准备了合适的选择，从而使品牌在每个细分市场都有占比。

5. 销售总额预测

因为一位理智而聪明的决策者在做出长远决策时不仅会仔细分析过往的案例，还会做出对未来的预期以增加长远决策的可行行和有效性，所以我们还需对未来销售额可能发生的变化进行预测和思考策略转变的时间和方式。

首先，我们创建所有商品月总销售额的时间序列 `ts=pd.DataFrame(df.groupby('date_time')['sale_total'].sum())` 并画出变化曲线。然后，我们画出连续三个月的滑动平均值与标准差，并对原始序列进行 Dicky-Fuller 平稳性检验[3]。

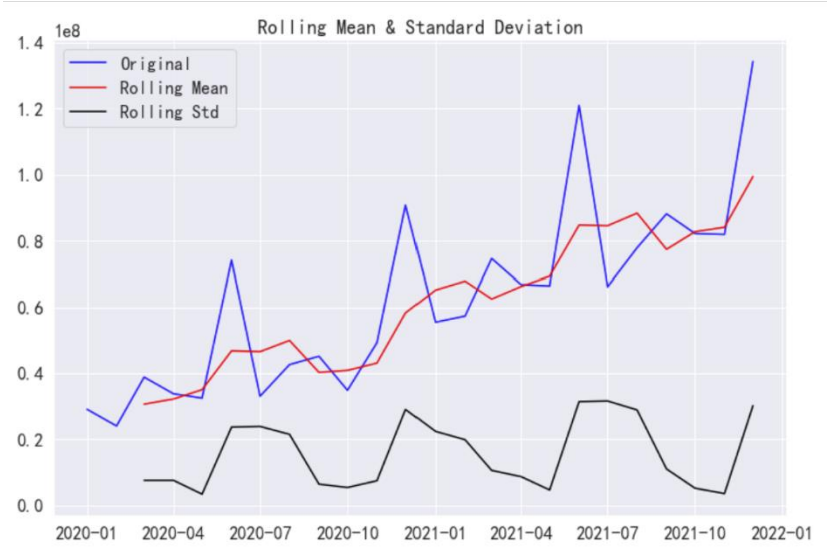

图 17：总销售额、三个月滑动平均和标准差随时间变化图

从上图我们可以发现，在这两年内，天猫上维生素类药品的月总销售额总体在波动中增加，最后的一个月销售额比初始的要多出一个亿。Dicky-Fuller 平稳性检验也给出了 0.93 的 p 值，所以该数据不是平稳的。除此之外，原序列有一定的季节性，在每年的六月和十二月都会显著增加，这一定程度上是因为天猫上最大的两个购物节分别在年初和年末举行，同时会有许多商家特意促使消费者一次购买半年的需求量以增强消费者对品牌的忠诚度。

5.1 ARIMA 模型

现在，为了更好的展现该序列的波动规律，我们使用对异常值具有鲁棒性的 SLT 分解法（局部加权回归）进行季节性和趋势性分。

```
from statsmodels.tsa.seasonal import STL

stl = STL(ts,robust=True)

res_robust = stl.fit()

fig = res_robust.plot()
```

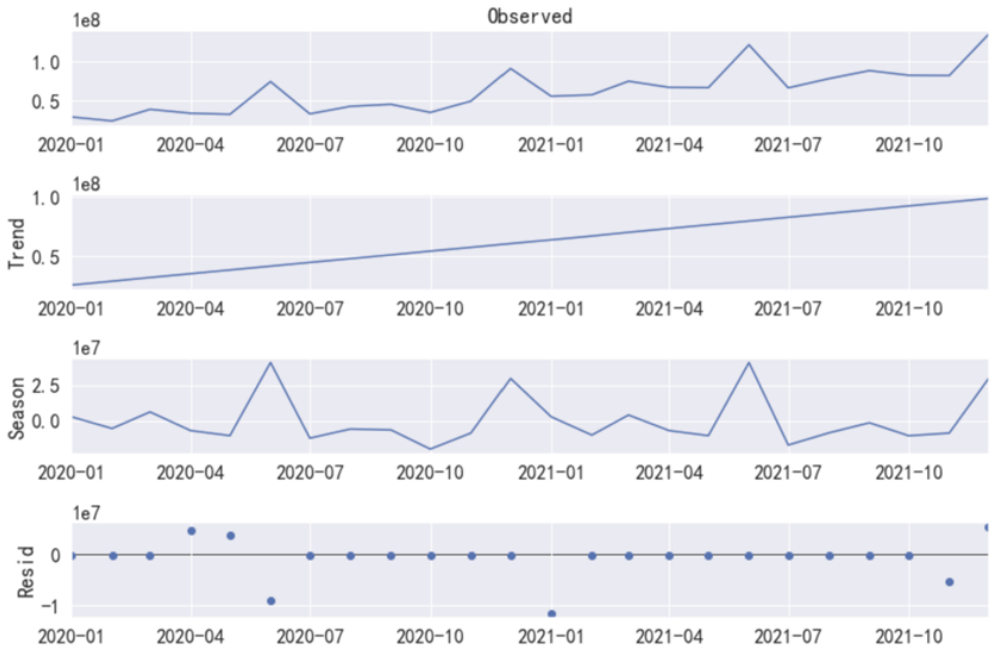

图 18：SLT 分解后的时间序列

上图证实了我们前面发现的正趋势和季节性，所以为了能够适用 ARMA 模型进行分析预测，我们尝试使时间序列满足平稳非白噪声的条件。为了惩罚数据的正趋势，我们对月总销售额取了对数 `ts_log=np.log(ts)` ；然后，为了减少季节性，我们对数据进行了差分处理 `ts_log_diff=ts_log-ts_log.shift(1).dropna(inplace=True)`。最后，我们画出差分后的数据并做了平稳性检验。

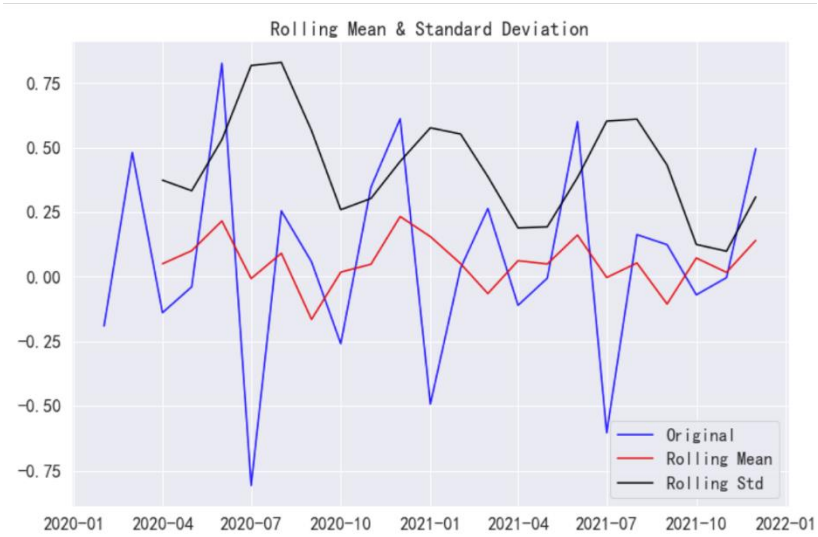

图 19：一阶差分后总销售额、三个月滑动平均和标准差随时间变化图

不论从图像，还是根据平稳性检验给出的 p 值，我们都能发现一阶差分对数序列是平稳的。为了保证该序列不是白噪音序列，我们做了以下检验：

```
from statsmodels.stats.diagnostic import acorr_ljungbox

acorr_ljungbox(ts_log_diff.sale_total, lags = [i for i in range(1,11)],boxpierce=True)
```

所有的 p 之都很小，我们拒绝原假设，故一阶差分对数序列是平稳的非白噪声序列。

我们已经得到一个平稳的时间序列，接下来就是选择合适的 ARIMA 模型，即 ARIMA 模型中合适的 p,q。第一步，我们通过 sm.graphics.tsaplots.plot_acf 和 sm.graphics.tsaplots.plot_pacf 创建自相关图和偏自相关图。

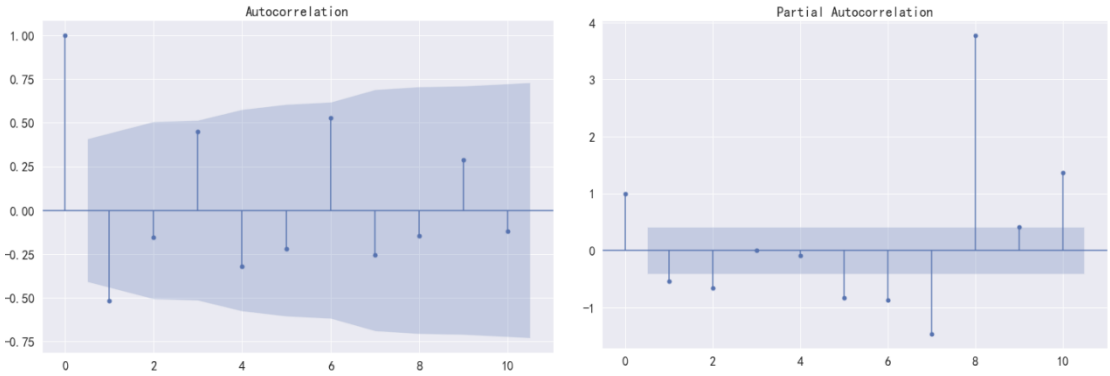

图 20：一阶差分对数序列的自相关（左）和偏自相关图（右）

从上面两张图中我们可以发现，自相关一阶截尾，偏自相关拖尾，所以应该选择 q = 1, p = 0。我们接着使用 sm.tsa.stattools.arma_order_select_ic(ts_log_diff,max_ar=4, max_ma=4,ic='aic')['aic_min_order'] 以 AIC 为准则选择最优的参数，得到 q = 3，p = 2。综合上面算法和人工选择的参数，我们将建立 ARIMA(0,1,1)和 ARIMA(2,1,3)模型，并选择在原数据集上以均方根误差（RMSE）为标准的表现较好的模型来预测未来三个月的总销量。

```
from statsmodels.tsa.arima_model import ARIMA

model_1 = ARIMA(ts_log,order=(0,1,1)) #建立 ARIMA 模型

results_ARIMA=model_1.fit()

pred=results_ARIMA.fittedvalues.append(results_ARIMA.predict(start='2022-01-01 ' , end='2022-03-01',dynamic=True)) #预测

prediction_log=pd.DataFrame(pd.Series(pred,copy=True).cumsum()+ts_log['sale_total'][0])

prediction_log.columns=['sale_total']

predictions_ARIMA_diff =pd.Series(results_ARIMA.fittedvalues,copy=True)

predictions_ARIMA_diff_cumsum= predictions_ARIMA_diff.cumsum()

predictions_ARIMA_log=pd.Series(ts_log['sale_total'][0],index=ts_log.index)
```

```
predictions_ARIMA_log=predictions_ARIMA_log.add(predictions_ARIMA_diff_cumsum,fill_value=0)

predictions_ARIMA=np.exp(prediction_log) #最终预测数据

differences = predictions_ARIMA.reset_index().drop([23,24,25],axis=0)['sale_total']-ts.reset_index().drop(0,axis=0).reset_index()['sale_total'] #差值

RMSE_1 = np.sqrt(sum((differences)**2)/len(ts)) #均方根误差
```

我们最终发现 ARIMA(0,1,1)比 ARIMA(2,1,3)的 RSME,所以我们使用 ARIMA(0,1,1)对 2022 年前三个月总销售额的预测值，他们分别约为 8.45 千万、8.94 千万和 9.45 千万，接着我们将原数据和与预测的数据画在同一张图中。

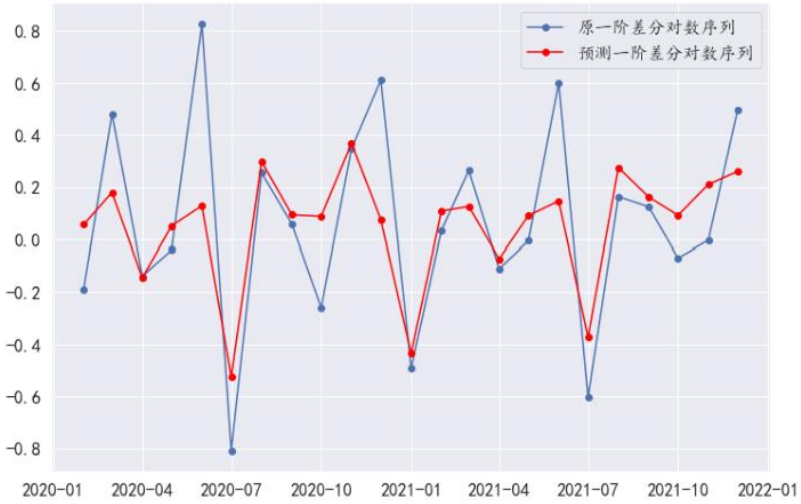

图 21：原一阶差分对数序列和 ARIMA(0,1,1)预测的一阶差分对数序列

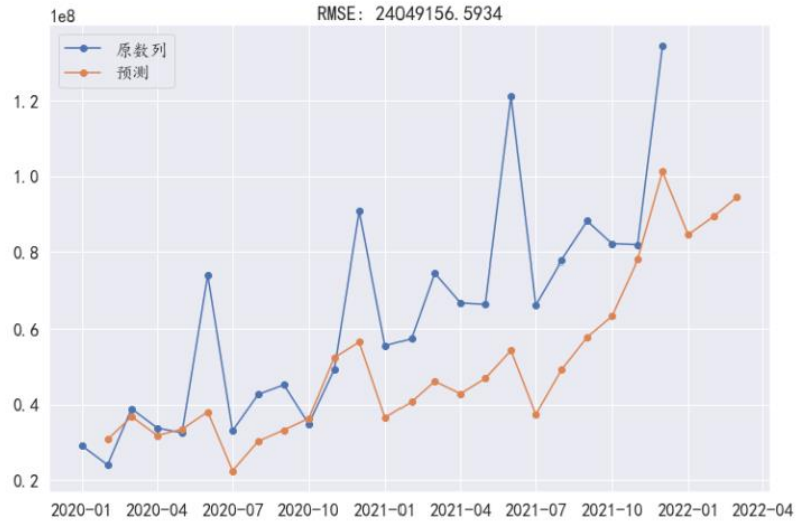

图 22：原时间序列和 ARIMA(0,1,1)预测的时间序列

我们可以看到 ARIMA(0,1,1)，也就是对一阶差分对数序列的 MA（1）模型较好的反映出了数据的走势，但在大多数情况下都比真实的销售额要小，特别是在增长

较快的 6 月和 12 月，简单的滑动平均很难预测出高增长。我们可以继续尝试加权的 MA 模型，但是很多时候有意识地做出较为保守的预测并时刻保持更改策略的警觉或许是更好的选择。

5.2 一些思考

除了 ARIM 这样的传统的统计学预测模型之外，我们还尝试了一些机器学习的算法，比如，先对数据做线性回归，再在残差上训练 XGBoost(Extreme Gradient Boosting)。由于在拆分了训练集和测试集之后，可训练的数据量太少，复杂的机器学习算法大多都会在训练集上过拟合，导致在测试集上的表现较差。下图是训练后的线性回归+XGBoost 在训练集和测试集给出的预测，其中黑线是原序列，蓝色是训练集预测序列，红色是测试集预测序列。

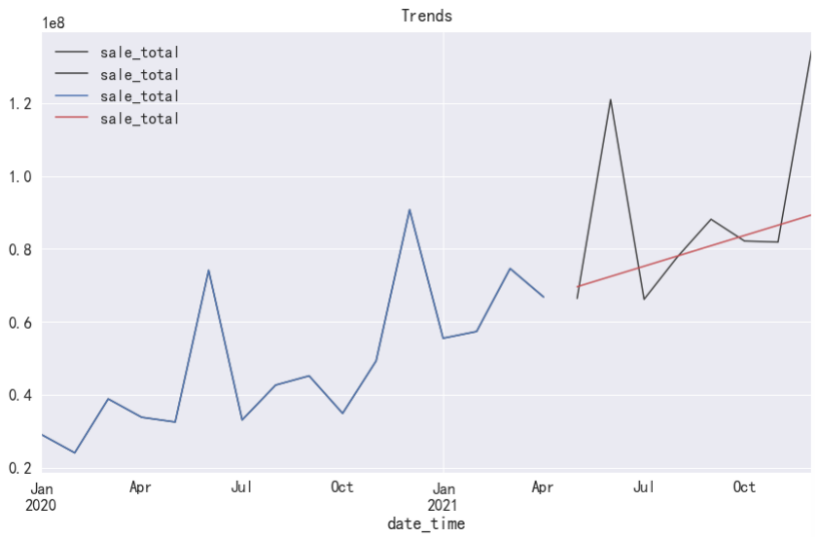

图 23：线性回归+XGBoost

不难看出，该模型泛化能力很差，所以我们并不在此使用相关的机器学习模型来预测。

预测的意义在于具有一定的前瞻性，而不是全盘相信数据，因为市场的波动是由很多看得到或者看不到的手一起控制的，比如，政府政策的调整、舆论走向的改变、金融市场的变动等等。站在 2023 年的年初往回看，突然的疫情开放让各种止痛、退烧和感冒药的需求激增，线下药店供不应求。同时，因为补充维生素对免疫系统有好处，电商上维生素类的药品大概率又获得了新的增长。但，在 2022 年开放之前，没有药企可以断言会出现需求激增并且要加大生产，因为卖不掉的药品会给企业带来巨大的亏损。在此，我们想再次说明就算统计模型给出了平稳增长的乐观结果，时时关注市场的走向并且做好随时改变生产或销售策略的准备工作也是至关重要的成功要素之一。

6. 总结 & 讨论

本此分析为了探索药企如何更好地经营电商，对两年内天猫维生素类药品的销售数据进行了清洗、分析与挖掘。在完成包括删除无用数据，文字信息提取，销售额计算等一系列数据预处理后，我们依次对店铺、药品和品牌进行了分析并使用 ARIMA 模型预测了天猫类药品未来三个月的销售总额。

在对店铺的分析中，我们发现 2020-2021 年间总销量前四的店铺都是与淘宝官方（阿里巴巴集团）相关的企业，其中阿里健康大药房占有了 45%的销售额，是占比最高的店铺。在对药品的分析中，我们发现药品数量随着时间在波动中稳步增加，销量前十的商品来自于七个品牌和四个店铺，其中星鲨的维生素 D 滴剂、康恩贝的维生素 C 片和 21 金维他的多种维生素符合药片都是其细分维生素药品的销售额第一名。在对品牌的分析中，我们发现天猫上一共有 516 个品牌参与了维生素类药品的销售，其中销售额前十的品牌中既有进口的定位较为高端的品牌，也有在本土销售多年薄利多销的品牌。

本次的分析有以下几个优点：

- 多维度，并且较为全面
- 结合了多种分析方法和学科理论
- 对找到的现象给出了较为合理的解读

本次的分析也存在以下几点需要改进的地方：

- 预处理中构造的变量并没有在每个分析中都被完备的考虑
- 很多原因分析和建议的提出都是基于个人经验和知识，背景调查不够
- 对销售额的预测由于时间关系所尝试的模型和方法还不足

7. 医药电商经营策略

基于疫情后 2020-2021 年维生素类天猫销售情况，我将从以下几个方面对本次在电商品牌上销售新的维生素品牌的策略制定提供见解和建议：

1）店铺选择

首先，入驻各个官方店铺包括阿里健康大药房的官方店铺和海外官方店铺，天猫超市以及其国际版，是享受平台巨大用户群体好处最快速的方式之一；如果有条件能够达成广告投放或者优先推荐的协议，那也是非常值得投入的前期成本。这是因为官方认证店具有更丰富的商品选择，更优秀的品控以及更完善的售后服务，人们更倾向于优先考虑阿里健康大药房等官方认证店铺。

如果品牌是消费者熟知和认可的成熟企业，将新的商品上架至自己的旗舰店并且通过追加赠品或适度降低定价来吸引消费者尝试新品也是十分必要的。因为与官方药品综合店铺相比，旗舰店的商品不需要给店铺佣金并且拥有更自由地定价权。

2）品牌定位

很多时候，选择品牌的定位决定了药品的选择和定价。如果打算定制一个较为高端的维生素药品品牌，那就要求保证品牌有良好的技术背书并且在营销上让更多的人认可其高端性。这样定价较高的品牌更适合售卖一些“锦上添花”或者“一应俱全”类的药品，比如美容美发补剂或者多种维生素矿物质复合片等，以顺应消费力较好的潜在客户的需求

相反的，品牌也可以选择走低价亲民的路线，但这对企业本身的要求较高：如何快速的打通销路，抢占市场和避免不必要的低价竞争？需要更仔细地考虑药品的选择与定价。

如果研发和产品运营的精力充足，品牌也可结合以上两种方式，像汤成倍健一样，将自身产品覆盖到所有可能的客户群体。

3）药品选择

药品选择不仅要参考品牌的定位、研发生产的优势、消费者的需求，还要考虑细分市场的潜力和竞争情况。虽然不论是在阿里健康大药房，还是整个天猫平台，这两年消费者购买的最多的都是刚需类的药品，例如孕妇补充维 D 的药片或滴剂，但该品类的增长潜力并不如别的品类，因为孕妇的数量不会快速增长而且长久以来，孕妇都会补充维生素 D。所以找到可以惠及到更多消费者并且尚未完全开发的维生素药品领域很重要。

还有一个值得借鉴的思路就是将自身的招牌商品做大并且与品牌名称捆绑，例如的 swisse 靠复合维生素起家，星鲨 c 长时间主要售卖维生素 D 滴剂等情况一样，如果想复制这些品牌的成功，那就要先找到新兴的未被消费者捆绑信任的品类，并为其下足功夫。

4）药品定价

商品定价不仅仅应是简单由价值导向的，也需要考虑一些消费心理学的内容。消费者对于这些本身广为人知的牌子的商品的定价更为宽容，高价和低价对于销售额的影响更少；这就意味着，新兴的品牌在没有大企业的背书的情况下，人们会更多地考虑是否物有所值。我们建议对于复合了所种维生素类的药品，定价可以适当较高；然而，对于单种维生素的药品，在不能证明原料的稀有性和生产工艺的前沿性的情况下，应定价稍低以复合消费者预期。

5）其他

线上医药服务是如今仍在探索中增长的电商领域，并且预测未来还有增长空间。但，新的领域意味着新的未知风险，时时关注市场、政策和舆论的走向并且做好随时改变生产或销售策略的准备工作是很有必要，甚至不可或缺的。

参考文献

[1] 杨丽霞. 孕妇血维生素 D 水平的影响因素及维生素 D 与妊娠并发症的相关性研究[D].南方医科大学,2016.

[2] Swisse.com.关于 SWISSE 斯维诗 探索之旅.http://swisse.com.cn/our-story/our-history/.[2022/01/06]

[3] Avasla.使用 Python 建立时间序列(ARIMA、MA、AR)预测模型[CP/OL]. https://blog.csdn.net/WHYbeHERE/article/details/109277597. [2022/01/06]

附录：代码

带输出的代码详见支撑材料

- **全局设置：**

```
#导入基本的模块
import re
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt


# 设置 Jupyter Notebook 在运行时显示所有的输出结果，而不仅仅是最后一个结果
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"


#画图设置
from matplotlib import rcParams
%matplotlib inline
rcParams['figure.figsize'] = 12,8 #统一图片大小
sns.set_style('whitegrid') #统一图片样式
sns.set(font_scale=1.5) #统一图片字体
plt.rcParams['font.sans-serif'] = ['KaiTi'] #用来正常显示中文标签
plt.rcParams['axes.unicode_minus'] = False #用来正常显示负号
```

- **数据预处理：**

```
df = pd.read_excel("data.xlsx")
df.dtypes #数据类型
df.shape #数据大小
df.isnull().sum(axis=0) #空值情况
```

```
df.duplicated().sum(axis=0) #重复情况
#可视化空值
import missingno as msno
msno.matrix(df)
```

```
df = df.drop("id",axis=1) #删除 id 列


#销售额计算
df['discount_number'] = df['discount'].apply(
    lambda x: float(re.findall(r'-?\d+\.?\d*e?-?\d*?',str(x))[0]) if len(re.findall(r'-?\d+\.?\d*e?-?\d*?',str(x)))!=0 else 10
) #提取数字，如果空值默认不打折（返回 10）


df['sale_total'] = df['price']*df['sold']*df['discount_number']*0.1
```

```
#找到 parameter 中所有可能含有的信息
content = []
for i in range(len(df.parameter)):
    #一条记录所含的 content
    splitted = str(df.parameter[i]).split('||')
    content.extend( [splitted[j].strip().split(':')[0] for j in range(len(splitted))] )


content = list(set(content))#去重
print(content)


def find_information(feature,x):


    #feature  输入上面四个 feature 中的一个
```

```
    #x：要从中找信息的字段


    if feature == '品类':
        feature = [ '药品通用名','药品名称', '品名', '药品商品名', '产品名称', '系列']
    elif feature == '品牌':
        feature == ['品牌']
    elif feature == '厂商':
        feature = ['厂名', '生产企业']
    elif feature == '规格':
        feature = ['药品规格', '规格']


    splitted = str(x).split('||')
    info = []
    for i in range(len(splitted)):
        if splitted[i].split(':')[0] in feature:
            info.append(splitted[i].split(':')[1])
    info = list(set(info)) #去重
    if len(info) != 0:
        return info[0]
    else:
        return None


find_information(x = df.parameter[3], feature = '品牌') #example


product,brand,factory,size = [],[],[],[]
for i in range(df.shape[0]):
    product.append(find_information(x = df.parameter[i], feature = '品类'))
    brand.append(find_information(x = df.parameter[i], feature = '品牌'))
    factory.append(find_information(x = df.parameter[i], feature = '厂商'))
```

```
        size.append(find_information(x = df.parameter[i], feature = '规格'))
df['品类'],df['品牌'],df['厂商'],df['规格'] = product,brand,factory,size


df = df.drop("parameter",axis=1) #删除 parameter 列
df = df.drop("discount",axis=1) #删除 discount 列
```

```
#用 brand 填补缺失的品牌
df['品牌'].fillna(df['brand'],inplace = True)
df = df.drop('brand',axis=1)


#找到用不同品牌名的同一品牌
brands = df['品牌'].dropna().unique()
b= [] #重复的品牌
for i in range(len(brands)-1):
    for j in range(int(i+1),len(brands)):
        if brands[i] in brands[j]:
            b.append([brands[i],brands[j]])
        if brands[j] in brands[i]:
            b.append([brands[j],brands[i]])


#删除第一个是 on 的元素
on = []
for i in range(len(b)):
    if b[i][0] == 'ON':
        on.append(b[i])


for j in on:
    b.remove(j)
b.remove(['朗迪', '精朗迪'])
```

```
b_dict = dict(zip([b[i][1] for i in range(len(b))],[b[i][0] for i in range(len(b))]))
df = df.replace(b_dict)
'哈药六' in list(df['品牌']) #example
```

```
#将 title 和 sku_name 中有用的文字部分重新整合
new_title = [df.title[i]+df.sku_name[i] if type(df.sku_name[i])==str else df.title[i] for i in range(len(df.title))]
#品类中也会有规格信息
new_title = [new_title[i]+df['品类'][i] if type(df['品类'][i])==str else new_title[i] for i in range(len(df.title))]
df['new_title'] = new_title

#删除 new-title 中没有用的部分
df['new_title'] = df['new_title'].apply(
    lambda x: re.sub('[^\u4E00-\u9FD5,a-z,A-Z./*0-9]+', ' ', x) #用空格替代？
)

#从 title 和 sku_name 找到规格
pattern = re.compile(r"[\d+]+[\u4E00-\u9FD5]+/[\u4E00-\u9FD5]")
df['size']   = df['new_title'].apply(
    lambda x: re.findall(pattern,x)[0] if len(re.findall(pattern,x))!=0 else None
)

df['规格'].fillna(df['size'],inplace = True)
df = df.drop('size',axis=1)
#从 title 和 sku_name 找到性别
df['性别'] = df['new_title'].apply(
    lambda x: '通用' if '男女' in x else '男性' if '男' in x else '女性'   if '女' in x else '通用'
```

```
)
#从 title 和 sku_name 中找到是否为复合维生素
p = re.compile(r"['维生素']+[a-e,A-E]+")
df['复合'] = df['new_title'].apply(
    lambda x: 1 if ('复合' in x) or ('综合'in x) or ('多维'in x) or ('维生素 ABCDE' in re.findall(p,x)) else 0
)

df = df.drop('new_title',axis=1)
#新增加一栏 title 中是否有推荐/同款
df['推荐/同款'] = df['new_title'].apply(
    lambda x : 1 if ('推荐' in x) or ('同款' in x) else 0
)

df = df.drop(df[df.price > 4000].index,axis=0) #异常值

#最终数据集展现
data = df.copy()
data.columns = [['日期','店铺','标题','SKU','单价','销量','折扣','销售额','品类','品牌','厂商','规格','性别','复合','推荐/同款']]
data.head()
```

- **店铺分析：**

```
df.shop_name.unique()
len(df.shop_name.unique()) #number
#计算每个店铺的销售额和销售占比
shops = pd.DataFrame(df.groupby(by='shop_name')['sale_total'].sum()).sort_values('sale_total',ascending=False).reset_index()
shops['proportion'] = shops['sale_total']/sum(shops['sale_total'])
```

```
shops
#总销售额饼图
#define data
data = list(shops.sale_total[0:5])
data.append(shops.sale_total.sum()-shops.sale_total[0:5].sum())
labels = list(shops.shop_name[0:5])
labels.append('其他')

#define Seaborn color palette to use
colors = sns.color_palette('pastel')[0:6]

#create pie chart
plt.pie(data, labels = labels, colors = colors, autopct='%.0f%%')
plt.rcParams['figure.figsize'] = (20.0, 8.0)
plt.legend(loc ='upper left',bbox_to_anchor=(0.9,0.9))
plt.title('各店铺总销售额占比')
plt.show()
```

```
#只含阿里健康大药房的数据
data_1 = df[df.shop_name == '阿里健康大药房'].drop(['shop_name'],axis=1).reset_index(drop=True)
data_1.head()
data_1.isnull().sum(axis=0)
data_1.shape
data_1.describe()

#number of products
len(data_1.title.unique())
len(data_1.品牌.unique())
```

```
len(data_1.品类.unique())

#品牌

data_1_brand = pd.DataFrame(data_1.groupby('品牌')['sale_total'].sum()).sort_values('sale_total',ascending=False)

data_1_brand['proportion'] = data_1_brand['sale_total']/sum(shops['sale_total'])

data_1_brand.head()


data_1[data_1['品牌']=='星鲨']['品类'].unique()

data_1[data_1['品牌']=='elevit/爱乐维']['品类'].unique()

data_1[data_1['品牌']=='伊可新']['品类'].unique()
```

```


#销售额和销售量随月份变化图

plt.rcParams['figure.figsize'] = (12.0, 8.0)

data_1_monthly = pd.DataFrame(data_1.groupby('date_time')[['sale_total','sold']].sum())

data_1_monthly.plot(subplots=True)


#2021-06 销量最高的十个商品

data_1[data_1.date_time == '2021-06-01'].sort_values('sale_total',ascending = False)['title'][0:10]


#与总体销量较高的商品对比

data_1_product = pd.DataFrame(data_1.groupby(by='title')['sale_total'].sum()).sort_values('sale_total',ascending=False).reset_index()

data_1_product['proportion'] = data_1_product['sale_total']/sum(shops['sale_total'])

data_1_product.head(10).title

#性别和复合

data_1['log_sale'] = np.log(data_1['sale_total'])

fig,axes=plt.subplots(1,3,figsize = (16,6))
```

```
for i, sex in enumerate(data_1['性别'].unique()):
    sns.histplot(data_1[data_1['性别']==sex], x='log_sale', hue="复合", kde=True,ax = axes[i],palette='hls').set(title=sex)
    axes[i].set(xlim=(2, 17))
#Two-Sample Kolmogorov-Smirnov Test
from scipy.stats import ks_2samp
for sex in list(data_1['性别'].unique()):
    print(sex)
    Data = data_1[data_1['性别']==sex]
    ks_2samp(data1=Data[Data['复合']==1]['sale_total'],data2 = Data[Data['复合']==0]['sale_total'])


#性别和复合
fig,axes=plt.subplots(1,3,figsize = (16,6))
for i, sex in enumerate(data_1['性别'].unique()):
    sns.histplot(data_1[data_1['性别']==sex], x='log_sale', hue="discount_number", kde=True,ax = axes[i],palette='hls').set(title=sex)
    axes[i].set(xlim=(2, 17))


#title 词云
import jieba
import itertools
from wordcloud import WordCloud


with open('stopwords.txt', 'r', encoding='utf-8') as f:
    stop = f.read()
stop = stop.split()
stop = [' ','片','粒','瓶','盒'] + stop
data_cut = data_1['title'].apply(jieba.lcut) # 分词
# 去除停用词
```

```
data_after = data_cut.apply(
    lambda x: [i for i in x if i not in stop]
)
print(data_cut.head())
print(data_after.head())

#统计词频并绘制词云
num = pd.Series(list(itertools.chain(*list(data_after)))).value_counts() #  统计词频

wc = WordCloud(font_path='./data/simhei.ttf', background_color='White')
wc2 = wc.fit_words(num)
plt.imshow(wc2)
plt.axis('off')
plt.show()
```

- 药品分析：

```
len(df.title.unique()) #商品种类数
T = pd.DataFrame(df.groupby('date_time')['title'].unique())

title_number = []
for i in range(T.shape[0]):
    title_number.append(len(T.loc[T.index[i],'title']))
T['title_number'] = title_number
T['title_number'].plot()

#销售额最高的十个药品的销售占比
products = pd.DataFrame(df.groupby(by='title')['sale_total'].sum()).sort_values('sale_total',ascending=False).reset_index()
```

```
products['proportion'] = products['sale_total']/sum(products['sale_total'])
products.head(10)
```

```
data_2 = df[df.title.isin(list(products.head(10).title))]
data_2.head()


data_cut = data_2['title'].apply(jieba.lcut) #  分词
data_after = data_cut.apply(
    lambda x: [i for i in x if i not in stop]
)


#统计词频并绘制词云
num = pd.Series(list(itertools.chain(*list(data_after)))).value_counts() #  统计词频


wc = WordCloud(font_path='./data/simhei.ttf', background_color='White')
wc2 = wc.fit_words(num)
plt.imshow(wc2)
plt.axis('off')
plt.show()
```

```
data_2 = df[df.title.isin(list(products.head(10).title))]
data_2.head()


data_cut = data_2['title'].apply(jieba.lcut) #  分词
data_after = data_cut.apply(
    lambda x: [i for i in x if i not in stop]
)
```

```
#统计词频并绘制词云
num = pd.Series(list(itertools.chain(*list(data_after)))).value_counts() #  统计词频


wc = WordCloud(font_path='./data/simhei.ttf', background_color='White')
wc2 = wc.fit_words(num)
plt.imshow(wc2)
plt.axis('off')
plt.show()


data_2.shape
data_2.shop_name.unique()
data_2.品牌.unique()
data_2.品类.unique()


data_2_monthly = data_2.drop(['price','sold','discount_number'],axis=1)
data_2_monthly.shape
#相同商品，同月的销售额相加
duplicated = data_2_monthly[data_2_monthly[['date_time','title']].duplicated()==True].reset_index()
duplicated.shape
data_2_monthly = data_2_monthly.drop(data_2_monthly[data_2_monthly[['date_time','title']].duplicated()].index,axis=0)
for i in range(148):
    t,d,s = duplicated.title[i],duplicated.date_time[i],duplicated.sale_total[i]
    index = data_2_monthly[(data_2_monthly.title==t)&(data_2_monthly.date_time==d)].index
    data_2_monthly.loc[index,'sale_total'] += s
sns.lineplot(x="date_time", y="sale_total",hue='title',    data=data_2_monthly)
plt.legend(loc ='upper left',bbox_to_anchor=(1,0.9))
plt.show()
```

```
sns.lineplot(x="date_time", y="sale_total",hue='品牌',    data=data_2_monthly)
plt.rcParams['figure.figsize'] = (12.0,8.0)
plt.legend(loc ='upper left',bbox_to_anchor=(1,0.9))
#plt.title('各店铺总销售额占比')
plt.show()
sns.lineplot(x="date_time", y="sale_total",hue='性别',    data=data_2_monthly)
plt.rcParams['figure.figsize'] = (12.0,8.0)
plt.legend(loc ='upper left',bbox_to_anchor=(1,0.9))
#plt.title('各店铺总销售额占比')
plt.show()
```

- 品牌分析：

```
len(df['品牌'].unique())
T = pd.DataFrame(df.groupby('date_time')['品牌'].unique())
brand_number = []
for i in range(T.shape[0]):
    brand_number.append(len(T.loc[T.index[i],'品牌']))

T['brand_number'] = brand_number
T['brand_number'].plot()
```

```
#mutual infomation
#Mutual information (non-linear relation)
from sklearn.feature_selection import mutual_info_regression

def make_mi_scores(X, y, discrete_features):
    mi_scores = mutual_info_regression(X, y, discrete_features=discrete_features)
    mi_scores = pd.Series(mi_scores, name="MI Scores", index=X.columns)
```

```
    mi_scores = mi_scores.sort_values(ascending=False)
    return mi_scores
X = df.copy().drop(['date_time'],axis=1) #计算原数据的 MI
y = X.pop('sale_total')
# Label encoding for categoricals
for colname in X.select_dtypes("object"):
    X[colname], _ = X[colname].factorize()
# All discrete features should now have integer dtypes (double-check this before using MI!)
discrete_features = X.dtypes == int
mi_scores = make_mi_scores(X, y, discrete_features)
mi_scores
```

```
brands = pd.DataFrame(df.groupby(by='品牌')['sale_total'].sum()).sort_values('sale_total',ascending=False).reset_index()
brands['proportion'] = brands['sale_total']/sum(brands['sale_total'])
brands.head(10)


data_3 = df[df.品牌.isin(list(brands.head(10).品牌))]
data_3.head()


data_3 = data_3.drop(['sku_name','厂商','规格'],axis = 1)
data_3.isnull().sum(axis=0)
data_3.shape
pd.DataFrame(data_3.groupby(['品牌','shop_name'])['sale_total'].sum()) #不同品牌在不同店铺的销售情况
data_3.describe()
sns.catplot(data=data_3, x="price", y="品牌", kind="box")
data_3['log_price'] = np.log(data_3['price'])
sns.catplot(data=data_3, x="log_price", y="品牌", kind="box")
```

```
data_3['log_sold'] = np.log(data_3['sold'])

sns.catplot(data=data_3, x="log_sold", y="品牌", kind="box")

X = data_3.copy().drop(['date_time','log_sold'],axis=1)

y = X.pop('sale_total')

# Label encoding for categoricals

for colname in X.select_dtypes("object"):

    X[colname], _ = X[colname].factorize()

# All discrete features should now have integer dtypes (double-check this before using MI!)

discrete_features = X.dtypes == int

mi_scores = make_mi_scores(X, y, discrete_features)

mi_scores #data_3 中的 MI

#Linear correlation

X['sale_total'] = y

corr = X.corr()

sns.heatmap(corr, xticklabels=corr.columns, yticklabels=corr.columns,annot = True,cmap="GnBu")
```

```
data_3.groupby('品牌')['title'].count()

data_3.groupby('品牌')['品类'].count()

data_3_monthly = data_3.drop(['price','sold','discount_number'],axis=1)

data_3_monthly.shape

#相同商品，同月的销售额相加

duplicated = data_3_monthly[data_3_monthly[['date_time','品牌']].duplicated()==True].reset_index()

duplicated.shape

data_3_monthly = data_3_monthly.drop(data_3_monthly[data_3_monthly[['date_time','品牌']].duplicated()].index,axis=0)

for i in range(duplicated.shape[0]):

    t,d,s = duplicated.品牌[i],duplicated.date_time[i],duplicated.sale_total[i]

    index = data_3_monthly[(data_3_monthly.品牌
```

```
==t)&(data_3_monthly.date_time==d)].index
        data_3_monthly.loc[index,'sale_total'] += s


sns.lineplot(x="date_time", y="sale_total",hue='品牌',    data=data_3_monthly)
plt.rcParams['figure.figsize'] = (12.0,8.0)
plt.legend(loc ='upper left',bbox_to_anchor=(1,0.9))
plt.title('各品牌每月总销售额')
plt.show()
```

- 销量预测：

```
ts = pd.DataFrame(df.groupby('date_time')['sale_total'].sum())
from statsmodels.tsa.stattools import adfuller


def test_stationarity(timeseries):
        #Determing rolling statistics
        rolmean=timeseries.rolling(3).mean()
        rolstd=timeseries.rolling(3).std()
        #Plot rolling statistics:
        orig=plt.plot(timeseries,color='blue',label='Original')
        mean=plt.plot(rolmean,color='red',label='Rolling Mean') #均值
        std=plt.plot(rolstd,color='black',label='Rolling Std') #标准差
        plt.legend(loc='best')
        plt.title('Rolling Mean & Standard Deviation')
        plt.show(block=False)


        #Perform Dickey-Fuller Test:
        print('Results of Dickey-Fuller Test:')
        dftest=adfuller(timeseries,autolag='AIC')
        dfoutput=pd.Series(dftest[0:4],index=['Test  Statistic','p-value','#Lags  Used','Number
```

```
of Observations Used'])
    for key,value in dftest[4].items():
        dfoutput['Critical Value (%s)'%key]=value
    print(dfoutput)
test_stationarity(ts)
ts_log=np.log(ts)
plt.plot(ts_log)
#Differencing
ts_log_diff=ts_log-ts_log.shift(1)
# reduced trend considerably
ts_log_diff.dropna(inplace=True)
test_stationarity(ts_log_diff)


from statsmodels.stats.diagnostic import acorr_ljungbox
acorr_ljungbox(ts_log_diff.sale_total, lags = [i for i in range(1,11)],boxpierce=True)
#将趋势和季节性分别建模，并返回序列的其余部分
from statsmodels.tsa.seasonal import seasonal_decompose
decomposition = seasonal_decompose(ts_log)
trend = decomposition.trend
seasonal = decomposition.seasonal
residual = decomposition.resid
#作图
plt.subplot(411)
plt.plot(ts_log,label='Original')
plt.legend(loc='best')
plt.subplot(412)
plt.plot(trend,label='Trend')
plt.legend(loc='best')
plt.subplot(413)
```

```
plt.plot(seasonal,label='Seasonality')
plt.legend(loc='best')
plt.subplot(414)
plt.plot(residual,label='Residuals')
plt.legend(loc='best')
plt.tight_layout()
from statsmodels.tsa.seasonal import STL


stl = STL(ts,robust=True)
res_robust = stl.fit()
fig = res_robust.plot()
```

```
import statsmodels as sm
import statsmodels as sm
import statsmodels.formula.api as smf
import statsmodels.tsa.api as smt
from statsmodels.tsa.arima_model import ARIMA
sm.graphics.tsaplots.plot_acf(ts_log_diff,lags=[i for i in range(11)])
sm.graphics.tsaplots.plot_pacf(ts_log_diff,lags=[i for i in range(11)])


p,q=sm.tsa.stattools.arma_order_select_ic(ts_log_diff,max_ar=4,max_ma=4,ic='aic')['aic
_min_order']


#对比 aic 和 bic
arma_mod20 = ARIMA(ts_log,(2,1,3)).fit()
arma_mod30 = ARIMA(ts_log,(0,1,1)).fit()
arma_mod40 = ARIMA(ts_log,(1,1,1)).fit()
values                                                                    =
[[arma_mod20.aic,arma_mod20.bic,arma_mod20.hqic],[arma_mod30.aic,arma_mod30.
bic,arma_mod30.hqic],[arma_mod40.aic,arma_mod40.bic,arma_mod40.hqic]]
```

```
df = pd.DataFrame(values,index=["AR(2,1,3)","MA(0,1,1)","ARMA(1,1,1)"],columns=["AIC","BIC","hqic"])

df

from statsmodels.tsa.arima_model import ARIMA

model_1 = ARIMA(ts_log,order=(0,1,1))

results_ARIMA=model_1.fit()

plt.plot(ts_log_diff,label='原一阶差分对数序列',marker='o')

plt.plot(results_ARIMA.fittedvalues,color='red',label= '预测一阶差分对数序列',marker='o')

plt.legend()


print(results_ARIMA.summary())


pred = results_ARIMA.fittedvalues.append(results_ARIMA.predict(start='2022-01-01',end='2022-03-01',dynamic=True))

prediction_log = pd.DataFrame(pd.Series(pred,copy=True).cumsum()+ts_log['sale_total'][0])

prediction_log.columns=['sale_total']

predictions_ARIMA_diff =pd.Series(results_ARIMA.fittedvalues,copy=True)

predictions_ARIMA_diff_cumsum= predictions_ARIMA_diff.cumsum()

predictions_ARIMA_log=pd.Series(ts_log['sale_total'][0],index=ts_log.index)

predictions_ARIMA_log=predictions_ARIMA_log.add(predictions_ARIMA_diff_cumsum,fill_value=0)


predictions_ARIMA=np.exp(prediction_log)

predictions_ARIMA.reset_index()['sale_total'][23:26]

predictions_ARIMA=np.exp(prediction_log)

print('预测的销售额为：',predictions_ARIMA.reset_index()['sale_total'][23:26])

plt.plot(ts,label='原数列',marker='o')

plt.plot(predictions_ARIMA,label='预测',marker='o')

differences = predictions_ARIMA.reset_index().drop([23,24,25],axis=0)['sale_total']-
```

```
ts.reset_index().drop(0,axis=0).reset_index()['sale_total']

RMSE_1 = np.sqrt(sum((differences)**2)/len(ts))

plt.legend()

plt.title('RMSE: %.4f'% RMSE_1)
```