该项目一共分四步走

1.首先进行数据处理

数据处理，我们用到的是pandas常用库

先进行去重drop_duplicates，把重复的内容处理好之后，我们开始删除一些无效的内容，例如表情包，无效词等

这里首先就是先去掉表情包，然后再判断该文本是否为中文，接着再去用停用词文本，去除无效词

```
def emjio_tihuan(x):
    x1 = str(x)
    x2 = re.sub('(\[.*?\])', "", x1)
    x3 = re.sub(r'@[\w\u2E80-\u9FFF]+:?/\[\w+\]', '', x2)
    x4 = re.sub(r'\n', '', x3)
    return x4

def is_all_chinese(strs):
    for _char in strs:
        if not '\u4e00' <= _char <= '\u9fa5':
            return False
    return True

def get_cut_words(content_series):
    # 读入停用词表
    stop_words = []

    with open("stopwords_cn.txt", 'r', encoding='utf-8') as f:
        lines = f.readlines()
        for line in lines:
            stop_words.append(line.strip())
```

接着我们stylecloud进行词云图，这样方便我们看看整体的分词效果如何，是否有一些词要不要去掉

最后的结果如下：

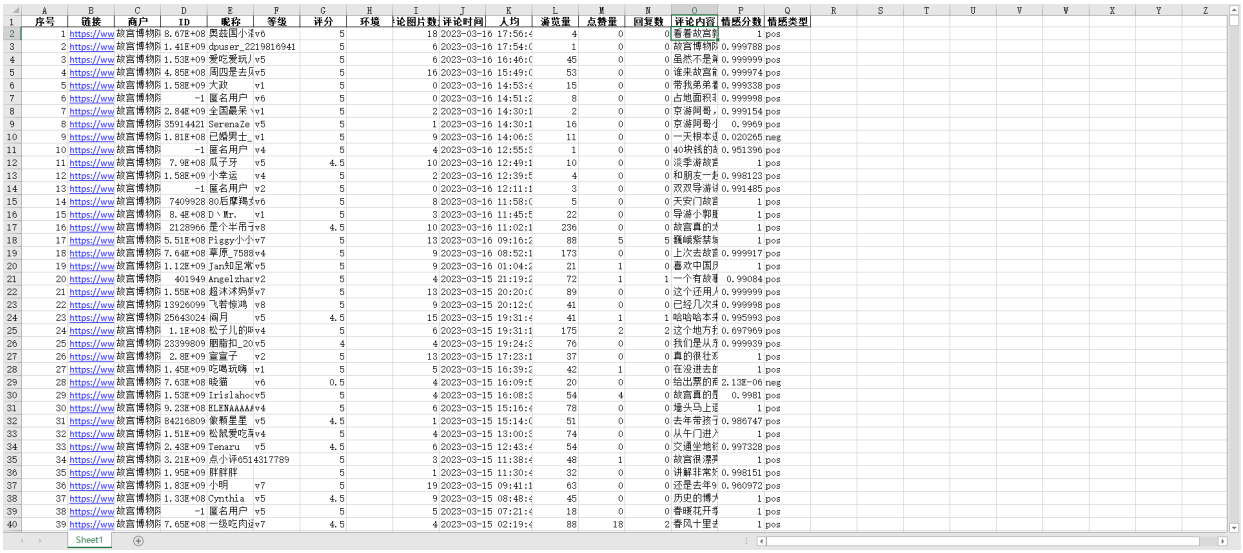
把结果都处理好了就开始NLP，情感分析

2、情感分析

文本挖掘中，情感分析是经常需要使用到，而进行主题模型分析之前，对数据集进行文本分类再进行分析具有必要性，因为分类以后，每一类的主题才会更明显。而snownlp是一个python写的类库，可以方便的处理中文文本内容

这里我们还是根据上面的方法，先进行分词处理，因为这样更有效帮助机器进行文本判断，从而给出正确的评价，这里有一点是要注意的

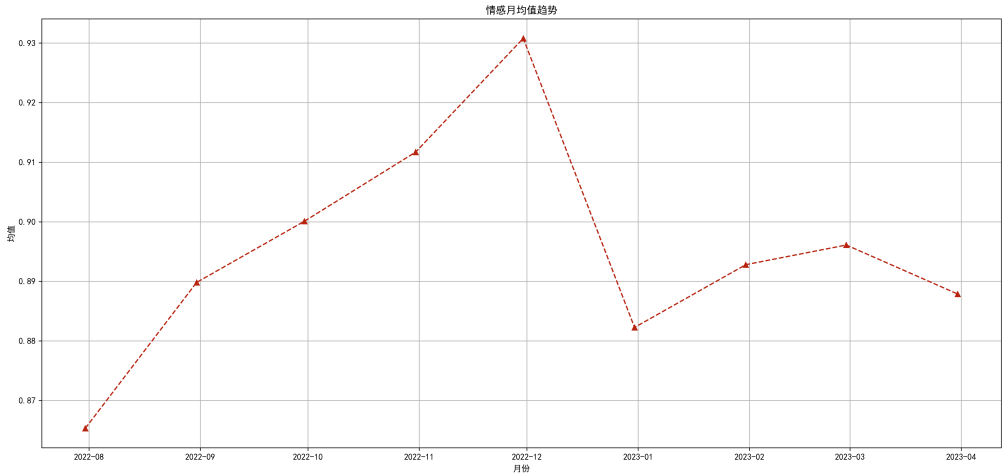
就是snownlp毕竟是广义的，无法做到精准判断该文本的正确评分，只能给出大概的数值，这也是所有机器学习的通病，毕竟是机器，NLP还是主观占比居多，所以提供的数值仅供参考



序号	链接	商户	ID	昵称	等级	评分	环境	评论时间	人均	浏览量	点赞量	回复数	评论内容	情感分数	情感类型
1	https://ww	故宫博物院	8.67E+08	奥兹国小浪	v6	5		18 2023-03-16 17:56:4		4	0	0	看看故宫的	1	pos
2	https://ww	故宫博物院	1.41E+09	dpuser_2219816941		5		6 2023-03-16 17:54:0		1	0	0	故宫博物院	0.999788	pos
3	https://ww	故宫博物院	1.53E+09	爱吃爱玩小	v5	5		6 2023-03-16 16:46:0		45	0	0	虽然不是第	0.999999	pos
4	https://ww	故宫博物院	4.85E+08	周四是去只	v5	5		16 2023-03-16 15:49:0		53	0	0	谁来故宫看	0.999974	pos
5	https://ww	故宫博物院	1.58E+09	大款	v1	5		0 2023-03-16 14:53:4		15	0	0	带我弟弟看	0.999338	pos
6	https://ww	故宫博物院		-1 匿名用户	v6	5		0 2023-03-16 14:51:2		8	0	0	占地面积3	0.999998	pos
7	https://ww	故宫博物院	2.84E+09	全面最呆	v1	5		2 2023-03-16 14:30:1		2	0	0	京湖网哥，	0.999154	pos
8	https://ww	故宫博物院	3591442	SeremaZe	v5	5		1 2023-03-16 14:30:1		16	0	0	京湖网哥小	0.9969	pos
9	https://ww	故宫博物院	1.81E+08	已婚男士_	v1	5		9 2023-03-16 14:06:5		11	0	0	一天根本逛	0.020265	neg
10	https://ww	故宫博物院		-1 匿名用户	v4	5		4 2023-03-16 12:55:3		1	0	0	40块钱的故	0.951396	pos
11	https://ww	故宫博物院	7.9E+08	瓜子牙	v5	4.5		10 2023-03-16 12:49:1		10	0	0	淡季游故宫	1	pos
12	https://ww	故宫博物院	1.58E+09	小幸运	v4	5		2 2023-03-16 12:39:5		4	0	0	和朋友一起	0.998123	pos
13	https://ww	故宫博物院		-1 匿名用户	v2	5		0 2023-03-16 12:11:1		3	0	0	双双导游讲	0.991485	pos
14	https://ww	故宫博物院	7409928	80后摩羯兔	v6	5		8 2023-03-16 11:58:0		5	0	0	天安门故宫	1	pos
15	https://ww	故宫博物院	8.4E+08	D丶Mr.	v1	5		3 2023-03-16 11:45:5		22	0	0	导游小郭是	1	pos
16	https://ww	故宫博物院	2128966	是个半吊子	v8	4.5		10 2023-03-16 11:02:1		236	0	0	故宫真的太	1	pos
17	https://ww	故宫博物院	5.51E+08	Piggy小小	v7	5		13 2023-03-16 09:16:2		88	5	5	巍峨紫禁城	1	pos
18	https://ww	故宫博物院	7.64E+08	草原_7588	v4	5		9 2023-03-16 08:52:1		173	0	0	上次去故宫	0.999917	pos
19	https://ww	故宫博物院	1.12E+09	Jan知足常	v5	5		9 2023-03-16 01:04:2		21	1	0	喜欢中国历	1	pos
20	https://ww	故宫博物院	401949	Angelshar	v2	5		4 2023-03-15 21:19:2		72	1	1	一个有故宫	0.99084	pos
21	https://ww	故宫博物院	1.55E+08	超冰冰奶茶	v7	5		13 2023-03-15 20:20:0		89	0	0	这个证用人	0.999999	pos
22	https://ww	故宫博物院	13926099	飞哥惊鸿	v8	5		9 2023-03-15 20:12:0		41	0	0	已经几次去	0.999998	pos
23	https://ww	故宫博物院	25643024	阎月	v5	4.5		15 2023-03-15 19:31:4		41	1	1	哈哈哈本来	0.995993	pos
24	https://ww	故宫博物院	1.1E+08	松子儿的叩	v4	5		6 2023-03-15 19:31:1		175	2	2	这个地方我	0.697969	pos
25	https://ww	故宫博物院	23399809	胭脂扣_20	v5	4		4 2023-03-15 19:24:3		76	0	0	我们是从方	0.999939	pos
26	https://ww	故宫博物院	2.8E+09	宣宣子	v2	5		13 2023-03-15 17:23:1		37	0	0	真的很壮观	1	pos
27	https://ww	故宫博物院	1.45E+09	吃喝玩确	v1	5		5 2023-03-15 16:39:2		42	1	0	在沧进去的	1	pos
28	https://ww	故宫博物院	7.63E+08	晓笛	v6	0.5		4 2023-03-15 16:09:5		20	0	0	给出票的看	2.13E-06	neg
29	https://ww	故宫博物院	1.53E+09	Irislaho<	v5	5		4 2023-03-15 16:08:3		54	4	0	故宫真的是	0.9981	pos
30	https://ww	故宫博物院	9.23E+08	ELENAAAAA	v4	5		6 2023-03-15 15:16:4		78	0	0	慢头马上逛	1	pos
31	https://ww	故宫博物院	84216809	歌颜星星	v5	4.5		1 2023-03-15 15:14:0		51	0	0	去年带孩子	0.986747	pos
32	https://ww	故宫博物院	1.51E+09	松鼠爱吃海	v4	5		4 2023-03-15 13:00:2		74	0	0	从午门进入	1	pos
33	https://ww	故宫博物院	2.43E+09	Tenaru	v5	4.5		6 2023-03-15 12:43:4		54	0	0	交通坐地铁	0.997328	pos
34	https://ww	故宫博物院	3.21E+09	点小评6514317789		5		3 2023-03-15 11:38:4		48	1	0	故宫很漂亮	1	pos
35	https://ww	故宫博物院	1.95E+09	胖胖胖		5		1 2023-03-15 11:30:4		32	0	0	讲解非常好	0.998151	pos
36	https://ww	故宫博物院	1.83E+09	小明	v7	5		19 2023-03-15 09:41:1		63	0	0	还是去年9	0.960972	pos
37	https://ww	故宫博物院	1.33E+08	Cynthia	v5	4.5		9 2023-03-15 08:48:4		45	0	0	历史的博大	1	pos
38	https://ww	故宫博物院		-1 匿名用户	v5	5		5 2023-03-15 07:21:4		18	0	0	春暖花开季	1	pos
39	https://ww	故宫博物院	7.65E+08	一级吃肉运	v7	4.5		4 2023-03-15 02:19:4		88	18	2	春风十里去	1	pos

Sheet1

接着我们获取到对应的分值之后，我们可以根据数据来做一个时间趋势图，从而得知，在每个月的一个分值走向，这里才去的是均值处理，把每个月的所有分值相加求平均值，所以这里还是有一定的参考价值，可以作为正确的评判标准，这里分值是从0-1直接的，接近0则是负面，接近1则是正面



3、LDA主题建模

LDA参考文献：https://zhuanlan.zhihu.com/p/75222819

https://zhuanlan.zhihu.com/p/76636216

这里我们的步骤和上面和一样

首先我们还是先分词，把无效词给除去掉

接着我们开始构造主题数，寻找最优主题数，这里采用困惑度严格来说，判断标准并不合适，基于此我们这里采用的是另一种方式，也就是通过各个主题间的余弦相似度来衡量主题间的相似程度

（2）寻找最优主题数

基于相似度的自适应最优LDA模型选择方法，确定主题数并进行主题分析。实验证明该方法可以在不需要人工调试主题数目的情况下，用相对少的迭代，找到最优的主题结构。具体步骤如下。

① 取初始主题数k值，得到初始模型，计算各主题之间的相似度（平均余弦距离）。

② 增加或减少k值，重新训练模型，再次计算各主题之间的相似度。

③ 重复步骤②直到得到最优k值。

利用各主题间的余弦相似度来度量主题间的相似程度。从词频入手，计算它们的相似度，用词越相似，则内容越相近。
假定A和B是两个n维向量，A是，B是，则A与B的夹角θ的余弦值通过式（4）计算。

$$P（w_j \mid d_j）= \sum_{s=1}^{K} P（w_i|z=s）\times P(z=s|d_j)（3）$$

$$\cos\theta = \frac{\sum_{i=1}^{n} A_i B_i}{\sum_{i=1}^{n}(A_i)^2 \sum_{i=1}^{n}(B_i)^2} = \frac{AB}{|AB|}（4）$$

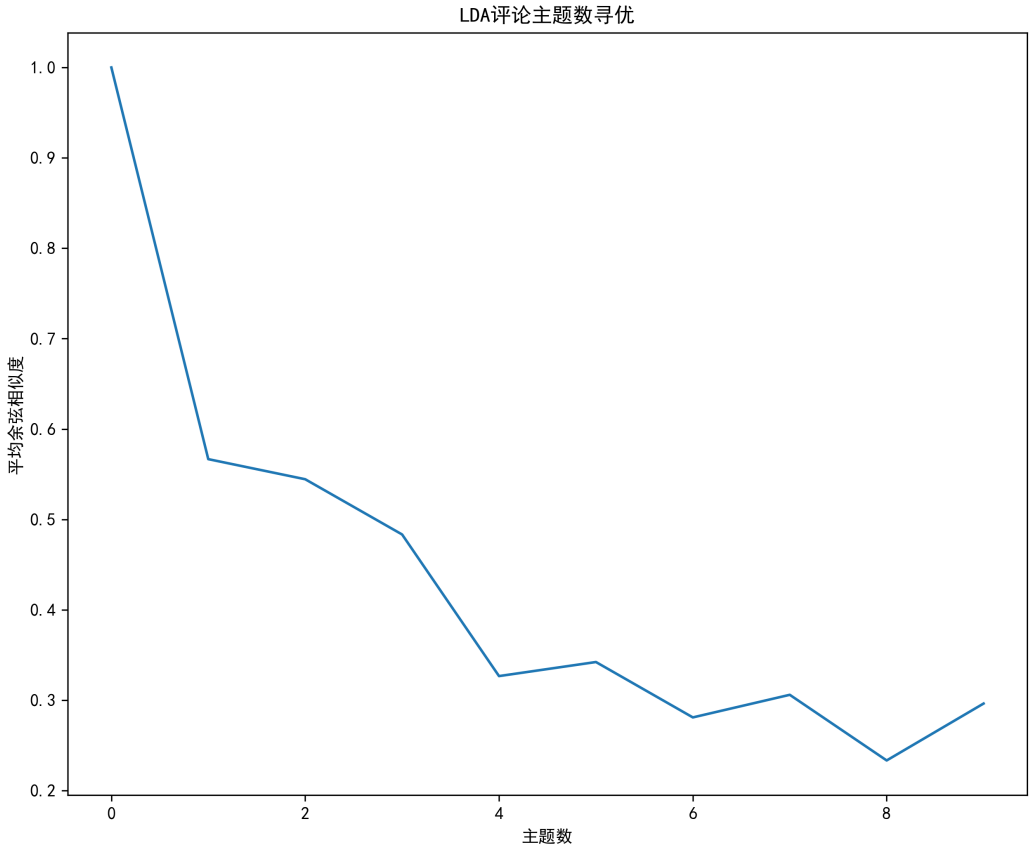
使用LDA主题模型，找出不同主题数下的主题词；每个模型各取出若干个主题词（比如前100个），合并成一个集合；生成任何两个主题间的词频向量；计算两个向量的余弦相似度，值越大就表示越相似；计算个主题数的平均余弦相似度，寻找最优主题数，如以下代码清单所示。

具体代码实现方式

```
# 构造主题数寻优函数
    def cos(vector1, vector2):  # 余弦相似度函数
        dot_product = 0.0
        normA = 0.0
        normB = 0.0
        for a, b in zip(vector1, vector2):
            dot_product += a * b
            normA += a ** 2
            normB += b ** 2
        if normA == 0.0 or normB == 0.0:
            return (None)
        else:
            return (dot_product / ((normA * normB) ** 0.5))

        # 主题数寻优

    def lda_k(x_corpus, x_dict):
        # 初始化平均余弦相似度
        mean_similarity = []
        mean_similarity.append(1)

        # 循环生成主题并计算主题间相似度
        for i in np.arange(2, 11):
            lda = models.LdaModel(x_corpus, num_topics=i, id2word=x_dict)  # LDA模型
训练
            for j in np.arange(i):
```

```
            term = lda.show_topics(num_words=30)

        # 提取各主题词
        top_word = []
        for k in np.arange(i):

            top_word.append(['',join(re.findall('"(.*)"', i)) \
                            for i in term[k][1].split('+')])  # 列出所有词

        # 构造词频向量
        word = sum(top_word, [])  # 列出所有的词
        unique_word = set(word)  # 去除重复的词

        # 构造主题词列表，行表示主题号，列表示各主题词
        mat = []
        for j in np.arange(i):
            top_w = top_word[j]
            mat.append(tuple([top_w.count(k) for k in unique_word]))

        p = list(itertools.permutations(list(np.arange(i)), 2))
        l = len(p)
        top_similarity = [0]
        for w in np.arange(l):
            vector1 = mat[p[w][0]]
            vector2 = mat[p[w][1]]
            top_similarity.append(cos(vector1, vector2))

        # 计算平均余弦相似度
        mean_similarity.append(sum(top_similarity) / l)
    return (mean_similarity)
```

处理好之后，再通过matplotlib来进行作图，在这里有3个低谷，可以选择4或者6或者8，根据最后lda呈现的模型去判断，效果好与坏，这样才用4



从而进行建模，最后呈现的效果图如下：

可以通过点击不同的圆圈，来查看不同主题下，不同主题词的权重，这里可以看出来选择4的效果挺不错的，圈圈分的比较开，不会重叠在一起