1、数据处理部分

数据预处理，主要是：分词，数据清洗，停用词，情感分析，中文判断 首先拿到数据，会先去掉标点符号，以及机械压缩，和去掉表情包，这样做的目前是减少噪音

```
#去掉标点符号，以及机械压缩
def preprocess_word(word):
    word1 = str(word)
    word1 = re.sub(r'#\w+#', '', word1)
    word1 = re.sub(r'【.*?】', '', word1)
    word1 = re.sub(r'@[\w]+', '', word1)
    word1 = re.sub(r'[a-zA-Z]', '', word1)
    word1 = re.sub(r'\.\d+', '', word1)
    return word1
```

```
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
```

接着会进行分词处理，在分词的过程中，只保留名词 形容词 动词等，并且判断该分词是否为中文，是否是两个词以上，是否在该词不在停用词库里面，减少噪音

```
def get_cut_words(content_series):
    try:
        # 对文本进行分词和词性标注
        words = pseg.cut(content_series)
        # 保存名词和形容词的列表
        nouns_and_adjs = []
        # 逐一检查每个词语的词性，并将名词和形容词保存到列表中
        for word, flag in words:
            #判断是否为名词或者形容词或者动词
            if flag in ['Ag','a','ad','an','Ng','n','v']:
                if word not in stop_words and len(word) >= 2 and is_all_chinese(word) == True:
                    # 如果是名词或形容词，就将其保存到列表中
                    nouns_and_adjs.append(word)
        if len(nouns_and_adjs) != 0:
            return ' '.join(nouns_and_adjs)
        else:
            return np.NAN
    except:
        return np.NAN
```

```
原数据总数: 6352
Building prefix dict from the default dictionary ...
Dumping model to file cache C:\Users\Administrator\AppData\Local\Temp\jieba.cache
Loading model cost 0.419 seconds.
Prefix dict has been built successfully.
清洗过后数据总数: 6306
```

最后保留新的文件

原始数据为：6352

清洗过后的数据为：6306

接着根据处理好的数据进行，TF-IDF处理

TF-IDF是词频和逆文档频率的乘积，其中词频是单词出现次数在总词数的占比，逆文档频率是总文档数与含某单词文档数比值的对数。换言之，一个特征词出现的频率越高，同时在总评论集中频率越小，其对评论的重要性越高，因为它可以标记出重要但并非通用词的特征词。TF-IDF随词频上升而增大，随逆文档频率上升而减小，因而出现频次高且对评论具有代表性的词会被赋予较高的 TF-IDF 值。

它的计算公式如下：

1. **TF (Term Frequency, 词频)**:
   词频用于衡量一个词在文档中出现的频率。对于某个词 t 在某个文档 d 中，词频的计算公式为：

$$\mathrm{TF}(t,d) = \frac{\text{词语 } t \text{ 在文档 } d \text{ 中出现的次数}}{\text{文档 } d \text{ 中所有词语出现的总次数}}$$

2. **IDF (Inverse Document Frequency, 逆文档频率)**:
   逆文档频率用于衡量词语在整个文档集中的普遍性。某个词 t 的IDF计算公式为：

$$\mathrm{IDF}(t,D) = \log \frac{N}{1 + |\{d \in D : t \in d\}|}$$

   其中：

   - N \) 表示文档集中总的文档数。
   - 加1是为了避免除零错误。

3. **TF-IDF 的计算公式**:
   结合TF和IDF，某个词 t 在某篇文档 d 中的TF-IDF值可以表示为：

$$\text{TF-IDF}(t,d,D) = \mathrm{TF}(t,d) \times \mathrm{IDF}(t,D)$$

代码实现如下：

```
    # 将文本中的词语转换为词频矩阵 矩阵元素a[i][j] 表示j词在i类文本下的词频
vectorizer = CountVectorizer()

# 该类会统计每个词语的tf-idf权值
transformer = TfidfTransformer()

# 第一个fit_transform是计算tf-idf 第二个fit_transform是将文本转为词频矩阵
tfidf = transformer.fit_transform(vectorizer.fit_transform(corpus))
# 获取词袋模型中的所有词语
word = vectorizer.get_feature_names_out()

# 将tf-idf矩阵抽取出来 元素w[i][j]表示j词在i类文本中的tf-idf权重
weight = tfidf.toarray()

data = {'word': word,
        'tfidf': weight.sum(axis=0).tolist()}

df2 = pd.DataFrame(data)
df2['tfidf'] = df2['tfidf'].astype('float64')
df2 = df2.sort_values(by=['tfidf'],ascending=False)
df2.to_csv('{}-TF-IDF相关数据.csv'.format(name),encoding='utf-8-sig',index=False)
```

根据计算的词，我们把原数据保存下来，并且用可视化的方式，展示tf-idf top30的词的具体情况

最后我们再去查看LDA的建模

LDA(Latent Dirichlet Allocation)是一种主题模型，通常用于文本分析中。在聚类分析中，使用LDA的目的是对数据进行主题建模，从而捕捉数据的隐含特征，提高聚类效果。LDA能够根据数据内在结构，自动发现“话题”，对于探索文本主题或按照主题进行文本分类分析具有较强的实用价值。

LDA具有以下意义：

1. 挖掘数据内在结构：在聚类分析中，LDA可以通过挖掘样本数据背后的主题，识别隐藏于数据中的内在结构，从而能够更好地理解数据特征之间的相关性和相互关系。
2. 降低数据维度：在聚类分析中，为了避免过度拟合数据、提高模型泛化能力，通常需要对数据进行降维处理。LDA可以提取数据中的主题特征，从而降低数据维数，减少数据处理的复杂度。
3. 提高聚类精度：通过LDA建模分析，可以对文本数据进行更精细的划分，将数据划分为相似的主题类别。基于这种类别划分，可以建立更准确的聚类模型。LDA也可以基于主题相似性，自动进行聚类，从而提高聚类精度。
4. 数据可视化：LDA减小了高维度的数据量，更容易进行可视化建模，可使聚类结果更有可解释性。同时，LDA也可以通过可视化技术，可视化各类主题和其相关的文档，便于分析者进行数据挖掘、数据分析甚至决策。

通过LDA方法进行聚类分析，能够在一定程度上缓解高维数据的复杂度，提高聚类的准确性，进而增强聚类分析的可理解性、可解释性，具有良好的可视化效果。因此，在聚类分析中，采用LDA方法对数据进行主题建模，可以帮助我们更好的理解数据的内在结构和相关性，获得更高效和更精确的聚类结果。

而在这个过程中，我们需要通过困惑度和一致性来确定我们的最优主题数

困惑度（perplexity）和一致性（coherence）是选择最优LDA主题模型时常用的指标，通常使用困惑度和一致性来评估主题模型中隐含狄利克雷分布(Dirichlet Distribution)的质量和准确度。

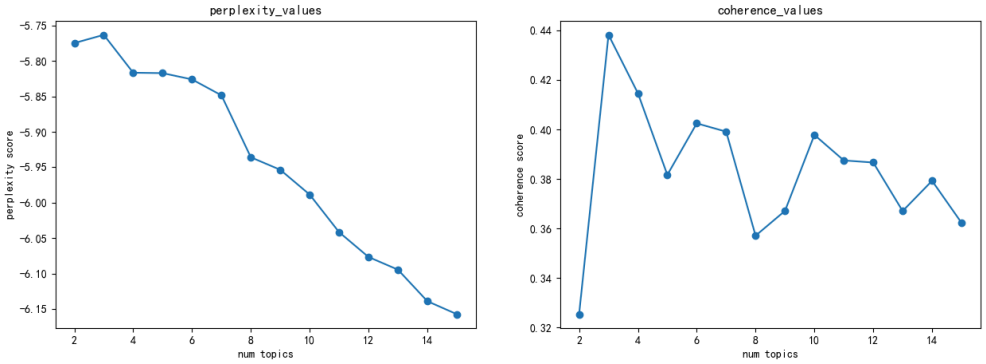
困惑度主要用于评估主题模型对未标注语料库的拟合能力，即在该模型下，这些隐含主题与实际背景密切相关的程度。困惑度越小，表示模型对新语料库的拟合效果越好，有更好的预测效果。

一致性主要用于评估主题模型中被提取的主题的质量和稳定性，主要从主题词语（Topic Term）出现的频率和相关性来评估主题的一致性。一致性越高，表示这些词语在主题下存在更密切的相关性，反之则表明主题下的词汇较为松散，难以对主题进行概括。

使用困惑度、一致性的目的是为了选择最优LDA主题模型，从而获得更好的聚类效果和更高的可解释性。通过困惑度和一致性评估，可以比较不同的主题分布参数等模型参数，并选择效果最优的模型。

在实践中，对于LDA主题模型的选择，我们通常会使用不同的主题数，计算困惑度和一致性，并选择困惑度最小、一致性最高的主题数作为最佳模型参数，以获得更好的聚类效果和模型拟合度。

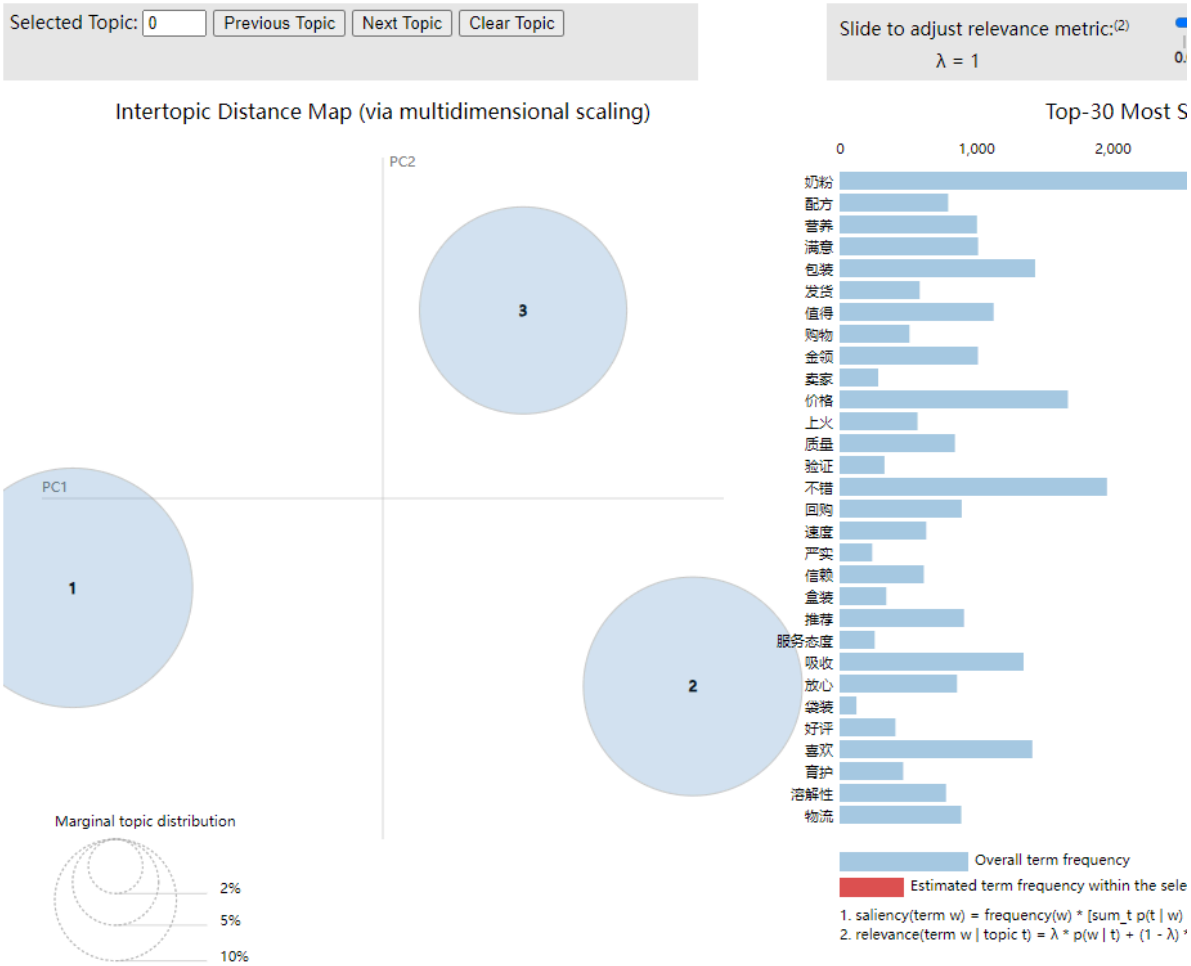
因此，通过困惑度、一致性指标的综合评估，可以帮助我们选择最佳的LDA主题模型，提高聚类效果和模型的可解释性。



一般而言，最佳主题数是由困惑度最小，一致性最高所对应的主题数确定的。根据提供的数据，可以看到困惑度在不同主题数下的变化趋势，以及随着主题数的增加，一致性的变化趋势。

通常情况下，困惑度会随着主题数的增加而减少，一致性则会随着主题数的增加先升高再下降。综合考虑困惑度和一致性指标，选择主题数使得困惑度最小、一致性最高，可以得到最优模型。

根据提供的数据，可以看到，主题数为3时困惑度最小，而主题数为3时一致性最高，因此，可以考虑选择该主题数(3)作为最佳主题数。但值得注意的是，选择最佳主题数时应该基于完整的数据集进行评估，并在保证较小的困惑度和较高的一致性的同时，尽量减小主题数，以获取更好的可解释性和模型准确性。



每个气泡也分布较为均匀，对应这里的建模，这里解释一下

LDA（Latent Dirichlet Allocation）气泡图是一种用于展示主题模型的可视化方式，通常用于表示文本或语料库的主题分布情况。LDA气泡图以一个二维平面上的主题分布图为基础，展示了不同主题之间的关系，主要由主题气泡和箭头构成。

气泡大小的代表意义：气泡大小通常代表LDA模型中的主题数量或主题词频，较大的气泡反映出该主题的权重较高，更重要或更具代表性。

气泡距离的代表意义：气泡之间的距离通常代表LDA模型中主题之间的相对距离关系。距离越近的主题，它们之间的主题词汇相关性越高，可能存在相近的主题标签或规律;反之，距离较远的主题之间可能存在更加明显的主题差异。

LDA气泡图通过多角度的呈现方式，可以帮助我们对LDA模型进行更加深入全面的分析。在可视化过程中，可以通过气泡的大小和距离，直观的感受到主题间的相对重要性、联系紧密程度等重要信息，进而进行可视化分析和解释，有助于更好地理解文本数据背后的主题结构。