数据预处理篇

1. 数据预处理与清洗

- **符号清洗**：通过正则表达式去除微博话题标签（#...#）、特殊方括号内容（【...】）、@用户标记、英文字母和数字等干扰符号
- **表情处理**：专门处理包含在方括号中的表情符号（如[哈哈]）和换行符
- **机械压缩**：使用`yasuo()`函数消除重复字符（如将"哈哈哈"压缩为"哈"）
- **空值处理**：删除处理过程中产生的空文本行

LDA篇

1. ## 一、代码实现核心思路

    1. **数据预处理阶段**
        - 加载已分词的文本数据（`fenci`列），通过过滤停用词和短词（长度≥2）进行二次清洗
        - 构建`gensim`词典对象和文档-词袋（BOW）格式的语料库，为LDA模型提供输入结构
    2. **主题数量选择**
        1. 遍历2-15个主题数，计算每个主题数下的**困惑度（Perplexity）**和**一致性（Coherence）**指标
        2. 通过一致性得分最大化自动选择最优主题数（`best_topic_number`）
    3. **模型训练与评估**
        1. 使用`gensim.LdaModel`训练最终LDA模型，设置400次迭代保证收敛
        2. 通过`pyLDAvis`生成交互式可视化HTML报告，展示主题间距离和关键词分布
    4. **结果输出**
        1. 保存主题概率分布、类型到原始数据文件（`lda_data.csv`）
        2. 导出各主题的特征词及其权重（`主题词分布表.csv`）
        3. 统计主题强度（文档占比）并保存为结构化表格

关键函数逻辑说明

困惑度与一致性计算

```
Python# 计算困惑度（对数形式）
perplexity = lda_model.log_perplexity(corpus)
# 计算一致性（c_v指标）
coherence_model_lda = CoherenceModel(model=lda_model, texts=train,
                                     dictionary=dictionary, coherence='c_v')
```

- 困惑度

  反映模型对新数据的预测能力（值越低越好）

- **一致性**评估主题内部词汇语义相关性（值越高越好），使用`c_v`指标更稳健

主题词提取逻辑

```
# 正则匹配提取词和权重
c1 = re.compile('\*"(.*?)"')  # 提取特征词
c4 = re.compile(".*?(\d+).*?")  # 提取概率权重
```

- 解析LDA输出的

  ```
  (词*权重)
  ```

  字符串，分离出特征词列表及其概率分布