DSC Pre Content - Process of LDA Modeling

1. Topic Modeling using LDA

After data cleaning, we obtain **list of tokenized posts** of each MBTI type
Now we're going to use LDA model to determine the topics and their distribution in the whole corpus
这一页是简单引入

2. Introduction of LDA

LDA (Latent Dirichlet Allocation) is a **generative probabilistic model** that uses very complex statictical models to represent topics in a corpus in the form of `word: possibility`

Priciple of LDA model:

1. Two hypothesises: Every document consists of several topics; Every topic consists of several words.
2. LDA model will first "guess" the topic label of a word and the topic distribution of a document
3. After many iterations, LDA model will optimize the topic and word distribution and find **underlying pattern** in the corpus
4. Eventually, it will output **topic-word lists** and **document-topic lists**
   简单介绍LDA模型的原理（复杂的讲不来QAQ），大概1-2页

3. Process of LDA modeling

在这部分开头可以展示一个建模的全局流程图，我画一下

1. Construct a dictionary
   Set filter condition: `no_above` , `no below` to ensure the word is neither too ordinary nor too rare

Python

```python
initial_dict=constract_initial_dict(source=cleaned_data,
                                    no_above=0.20,
                                    no_below=50)
```

2. Convert all the post to BoW (Bag of Words) representation and check word frequency to access the dictionary
   这里我也画一个简单的映射图，并展示我检查词典的csv文件
3. Train several temporary LDA models to optimize number of topic

Python

```python
def optimize_topic_num(
  start,
  end,
  step,
  dict=initial_dict["overall_dict"],
  corpus=initial_dict["all_corpus"],
  text=initial_dict["all_original_text"]
):

  output=pd.Series({},dtype=float)
  topic_num_range=range(start, end+1, step)
  for topic_num in tqdm(topic_num_range, desc="计算全局模型主题数"):
  # 训练 LDA 模型 (在所有帖子数据上)
    temp_lda_model=LdaMulticore(
        corpus=corpus,    # 使用所有帖子的词袋语料
        id2word=dict,          # 使用全局词典
        num_topics=topic_num,
        random_state=100,
        chunksize=100,         # 减小chunksize以加快更新频率
        passes=10,             # 减少passes以缩短总训练时间
        iterations=50,         # 明确指定每次迭代的次数
        alpha=0.01,            # 使用较低的固定值促进主题稀疏性
        eta=0.01,              # 使用较低的固定值促进词汇稀疏性
        per_word_topics=False,
        workers=None
    )

    # Evaluate the model
    temp_chmodel=CoherenceModel(
        model=temp_lda_model,
        texts=text,
        dictionary=dict,
        coherence="c_v"
    )
    output[topic_num]=temp_chmodel.get_coherence()
  print(output)
```

Adjust the coefficients:

1. `chunksize`
2. `passes` , `itereations`

Observe the c_v score and infer the interval where optimal topic number lies

这里我展示一下我算出来的c_v分数结果，举个例子

4. Train the final LDA model

Python

```
# Train the optimized final model with enhanced parameters for better convergence and topic separation
topics=19

lda_model = LdaMulticore(
    corpus=initial_dict["all_corpus"],
    id2word=initial_dict["overall_dict"],
    num_topics=topics,
    random_state=100,
    chunksize=100,            # 大幅减小：提高更新频率，改善收敛性
    passes=300,               # 适当减少：配合其他优化参数
    iterations=150,           # 新增：增加每个pass的迭代次数
    alpha=0.01,               # 从asymmetric改为小值：促进文档-主题稀疏性，提高主题区分度
    eta=0.01,                 # 从auto改为小值：促进词汇-主题稀疏性，减少主题混杂
    decay=0.5,                # 新增：控制学习率衰减，改善收敛稳定性
    offset=1.0,               # 新增：学习率起始值          # 新增：每个chunk后更新模型
    minimum_probability=0.01, # 新增：过滤低概率主题分配
    per_word_topics=False,
    workers=None,               # 启用并行化加速训练
    eval_every=20             # 减少评估频率，降低计算开销
)
```

5. Manually check the modeling result with the assistance of `pyLDAvis` dynamic visualization graph

Python

```
import pyLDAvis.gensim_models as gensimvis
def create_pyldavis_visualization(self,save_path=f"final_output/lda_visualizat
        print("Creating pyLDAvis visualization...")
        # Prepare pyLDAvis visualization
        vis_data=gensimvis.prepare(
            self.model,
            self.corpus,
            self.dictionary,
            sort_topics=False
        )
        # Save as HTML file
        pyLDAvis.save_html(vis_data,save_path)
```

这里直接上链接

https://dominicmin.github.io/Intro_to_DS_Assignment/lda_visualization.html

现场演示建模结果可视化和λ值的作用

6. Update stopwords and remove them from the corpus

```python
# Load custom stopwords
with open("custom_stopwords.json","r") as f:
    custom_stopwords=json.load(f)
stop_words.update(custom_stopwords)
```

这里可以放上我停用词列表的截图

7. Repeat the process above until the result is acceptable

4. Transition to the next part

After obtaining the final topic modeling result, we will analyse topic preference between different MBTI dimensions and groups, and use clustering to find pattern inside of the 16 MBTI types.

Let Mao Yifei to tell you more.