利用数据库编程进行LDA模拟和检验

10230327 屈良煜

一、本项目原理：LDA模型简述

LDA将文档生成过程建模成一个概率图模型，即假设文档是由多个单词混合生成的，其中单词的出现是由潜在主题决定的。具体过程如下：

1. 假设每篇文档的主题分布服从一个狄利克雷分布（$\theta \sim \mathrm{Dirichlet}(\alpha)$），每个主题的单词分布也服从一个狄利克雷分布（$\phi \sim \mathrm{Dirichlet}(\alpha)$）
2. 对于每篇文档，我们先从主题分布（θ）中抽取一个主题记作（z），再从主题（z）的单词分布（ϕ_z）中抽取一个单词。

3. 则给定（K）个主题和文档（d），文档中由n个词语组成的文本（ω）被生成的概率为：

$$P(w) = \prod_{n=1}^{N} \sum_{z=1}^{K} P(w_n \mid z; \phi) P(z \mid d; \theta)$$

（边缘似然）

二、Dirichlet Distribution概述

密度函数

狄利克雷分布 $\mathrm{Dir}(\boldsymbol{\alpha})$ 是定义在 $K-1$ 维辛形（simplex）上的连续概率分布，是 **Beta 分布在多元（K 元）情况下的推广。

假设随机向量 $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_K)$ 服从 $\mathrm{Dirichlet}(\boldsymbol{\alpha})$，其中 $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_K)$ 是超参数，且 $\alpha_k > 0$。

其概率密度函数 (PDF) 为：$f(\boldsymbol{\theta} \mid \boldsymbol{\alpha}) = \frac{1}{B(\boldsymbol{\alpha})} \prod_{k=1}^{K} \theta_k^{\alpha_k - 1}$ 约束条件: $\theta_k \in [0, 1]$ 、$\sum_{k=1}^{K} \theta_k = 1$

其中 $B(\boldsymbol{\alpha})$ 是多元 Beta 函数（归一化常数）： $B(\boldsymbol{\alpha}) = \frac{\prod_{k=1}^{K} \Gamma(\alpha_k)}{\Gamma\left(\sum_{k=1}^{K} \alpha_k\right)}$

选择狄利克雷分布作为先验的理由

1.它是多项式分布的“共轭先验” (Conjugate Prior)

当我们使用狄利克雷分布作为多项式分布的先验时，后验分布仍保持狄利克雷分布的形式。后验分布与先验分布的形式相同，使得我们在**推断**过程中可以进行代数运算，而不是进行复杂的积分运算（尤其是在使用 MCMC 或吉布斯采样时），这在计算上是极其高效的。

2.它适合建模比例

狄利克雷分布完美地定义在满足**和为 1**的向量空间上。LDA 的两个核心参数 $\boldsymbol{\theta}$ 和 $\boldsymbol{\phi}$ 本质上都是概率向量（即所有元素和为 1）

- **文档主题比例 $\boldsymbol{\theta}$**: 一篇文档必须 100% 由所有主题构成。$\sum_{k=1}^{K} \theta_k = 1$。

--

- **主题词语比例 ϕ**: 一个主题必须 100% 由所有词汇构成。$\sum_{v=1}^{V} \phi_v = 1$。

三、本项目背景与目标

实际上，使用LDA等主题模型分析文本时，模型推断出的结果（文档主题比例）只是模型对真相的猜测，猜测准确度未知。我采用统计模拟的方式对模型的性能进行科学量化，检验LDA模型推断的准确度如何，从而为LDA在推荐系统、文本挖掘等领域的可靠使用提供坚实的统计学基础。

具体来说：

1. 生成$N = 3000$篇文档。每一篇文档都附带一个**已知的**、**真实的**主题比例向量 ($\boldsymbol{\theta_{true}}$)，并存储在数据库中，作为验证的基础。

2. 以这3000篇文档作为输入数据，利用Python训练LDA模型，推断出每篇文档的预测主题分布($\boldsymbol{\theta_{pred}}$)。

3. 使用两个向量的**余弦相似度**作为量化指标，对比 $\boldsymbol{\theta_{true}}$ **向量** 和 $\boldsymbol{\theta_{pred}}$ **向量** 在向量空间中的夹角。夹角越小，相似度（分数越接近 1）越高，表明模型推断越准确。

4. 将数据生成、存储、模型推断、结果验证等一系列工程产生的所需的样本和产生的结果完整地储存在数据库中，以便批量进行分析。

注：

1. 本项目主要用到Python 、sqlite3。本项目所用方法为统计模拟。
2. 使用SQLite的好处：Python3自带 `sqlite3` 模块；没有权限验证问题，SQLite本身是db文件；适合功能演示，更加专注于LDA算法本身。

四、本项目流程具体介绍

1、初始化与参数

在 `config.py` 定义了K个主题、V个词汇，N篇文档，ALPHA_PARAM=0.5等实验常量，循环生成V = 1000个词汇，并用“主题+编号”的格式对词汇编码以作区分。

在`db_manager.py`中，定义数据库连接函数`get_db_connection`为项目与本地sqlite3数据库文件交互作准备；定义`initialize_database`函数，利用sql语句创建sim_parameters等表，为记录参数和实验结果做准备。紧接着，定义`record_simulation_parameters`函数，记录参数。

在主控程序`main_run.py`中，调用上述函数，并获取实验编号（run_id）

2、统计模拟（核心步骤）

在`stat_sim.py`中，构建K行V列的ϕ矩阵，模拟主题—词汇概率分布。**分配策略**：将90%的概率分配给某个主题的专属词汇，将10%的概率分配给其它词汇，同时保证概率归一化。

在`stat_sim.py`中，定义`generate_documents`函数，根据LDA原理进行狄利克雷抽样，生成N篇文档和对应的真实θ向量。最后再进行数据格式化处理，如将词语连接成完整字符串文本、将实验序号和真实θ打包成元组。

在主控程序中调用上述函数，进行统计模拟。

3、数据入库

在`db_manager.py`中，定义`bulk_insert_documents`函数，通过SQL语句，将生成的文档和真实θ向量批量导入数据库保存。

在主控程序中调用该函数，实现数据入库。

4、模型训练与推断

在`db_manager.py`中，定义`fetch_documents_for_analysis`函数，通过SQL语句从数据库中提取第三步的文档和向量作为比对。

在`stat_sim.py`中，定义`train_and_predict_lda`函数，配置LDA模型、训练模型，并推断主题分布，生成`theta_pred_matrix`，用来和真实向量做比较。

在主控函数中调用上述函数。

5、量化检验

在`stat_sim.py`中定义`calculate_cosine_similarity`函数，计算两向量余弦相似度。

在主控函数中，对比θ_{pred}和真实θ向量，计算余弦相似度。将结果储存在数据库中。最终结果为0.7088。

五、本项目使用说明

由于本项目核心功能只有一个，即对比θ_{pred}和真实向量，计算余弦相似度，因此GUI功能设计较为直接。

只需运行`GUI.py`，则会弹出运行窗口。左侧边栏部分为实验参数。点击左上角"运行LDA模拟"键，则开始模拟。等待时间约为10秒，可以同时打开终端观察模拟过程。结果显示在左侧边栏下方。模拟完成后，点击右上角"退出系统"，即可结束模拟。

本项目使用AI情况（Gemini 3pro & Gemini 2.5 Flash）： 1.参数和函数的命名及格式化输出上，首先写出中文命名，然后借助AI翻译为英文并调整格式。2. `GUI,py`文件中，窗口的大小、颜色等设计，使用了AI。自主完成了`GUI.py`中前后端接口部分。

本项目不足：1.由于时间原因，没有实现ALPHA_PARAM参数变化的不同情况 2.对于LDA理论了解较为充分，但代码部分仍以调包为主，包内部更多还是黑箱。

六、本项目实际意义

- 为 LDA 模型的应用提供统计可靠性背书（主要）
- 在推荐系统、文本聚类和情感分析等任务中，$\boldsymbol{\theta}$ 向量经常被用作文档的精简特征表示。只有经过高相似度验证（即模型推断准确）的 $\boldsymbol{\theta}$ 向量，才能确保后续算法建立在高质量的数据基础之上，避免“垃圾输入，垃圾输出”的问题。