支持决策制定的基于论证的系统 （机器学习部分）

新的带先验知识的推导和基于机器学习相似性去重



1.介绍

决策影响因素一直是个值得研究的问题，它可以说是所有机器学习从业者、数据分析师、数据科学家的噩梦， 比如特征选取经常收到人为因素的干扰。

本系统算法部分主要是通过贝叶斯机器学习和Kmean算法来处理收集的数据中如错误、值缺失、拼写错误和值不一致的地方，减少人工修改和干预因素，通过算法自动补全，并且还能够根据概率统计出常识知识来推断信息，计算出的概率分布提供更多信息帮助决策系统，从而摆脱单纯依靠人的常识的做法。

本文主要旨基于python3/numpy/pandas/pgmpy技术栈，我们使用先天贝叶斯算法并且结合其他脏数据去除方法，如给予机器学习的knn分类去掉重复数据等，在工程上融合其他机器学习方法（sklearn/pgmpy)，进一步提升工程的可用性和自动化程度。

2.背景

**2.1** 当前情况

Argumentative-decision-making 的系统是概率计算项目(Probabilistic Computing Project)研究人员编写的针对特定领域的概率编程语言，旨在简化人工智能应用程序的开发并实现自动化，例如时间序列和数据库进行建模)。

根据Anaconda所做的调查，清洗数据可能会占用数据科学家1/4的时间。如何将这个任务自动化，一直以来都是一个具有挑战性的任务。因为不同的数据集需要不同类型、不同层次的清理，而且清晰过程经常需要依赖常识来对世界上的物体进行判断，例如一个城市表中，需要判断哪些值不属于这列。

比如AI模型里80%的工作要放在数据上，而数据清洗又是保证模型质量的关键步骤，它涉及到领域知识等等，往往很难自动化，我们构建啦Argumentative-decision-making工程，能极大手工清洗数据。 目前是当前领域的首次给予bayes模型的自动清理工具的尝试。

**2.2** 项目描述

Argumentative-decision-making 为这类判断提供了一个通用的常识模型，可以根据特定的数据库和错误类型进行定制化操作。Argumentative-decision-making 使用基于知识的方法来自动化数据清洗的过程: 即用户在定义数据的时候，已经隐含包括了数据库的背景知识以及可能出现的各种问题。

例如，有一个场景，当清理公寓列表数据库中的国家名称的时候，如果有人说他们住在比佛利山庄，但是没有留下任何其他信息怎么办？虽然在加州有著名的比弗利山庄，但在佛罗里达州、密苏里州和德克萨斯州也有一个，而且在巴尔的摩有一个被称为比弗利山庄的社区。你怎么知道这个人住在哪里？这就是 Argumentative-decision-making 发挥作用的地方：

用户可以向 Argumentative-decision-making 提供有关域以及数据可能如何损坏的背景知识。Argumentative-decision-making 通过常识性概率推理将这些知识结合起来得出答案。例如，如果对租金类的常识有更多的了解，Argumentative-decision-making 就能推断正确的比弗利山是在加利福尼亚，因为被调查者居住的地方的租金成本很高。

**2.3** 文献调查

**Practical Probabilistic Programming 9787115448743**

**Think Bayes 9787115384287**

**Bayesian Methods for Hackers: Probabilistic Programming and Bayesian Inference 9787115438805**

说明

3.工程架构和功能介绍

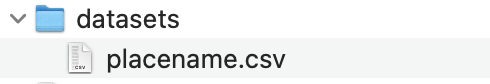
工程分成3部分：

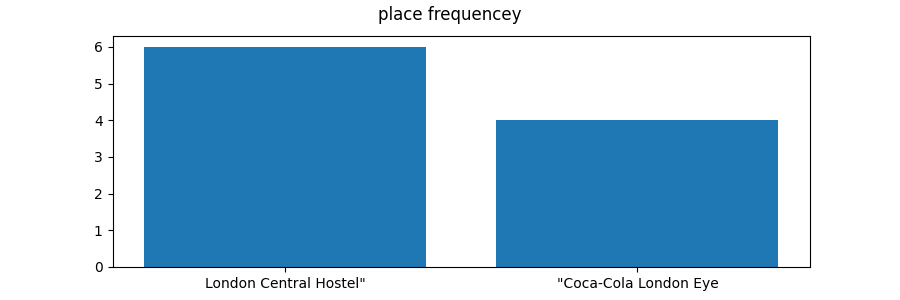
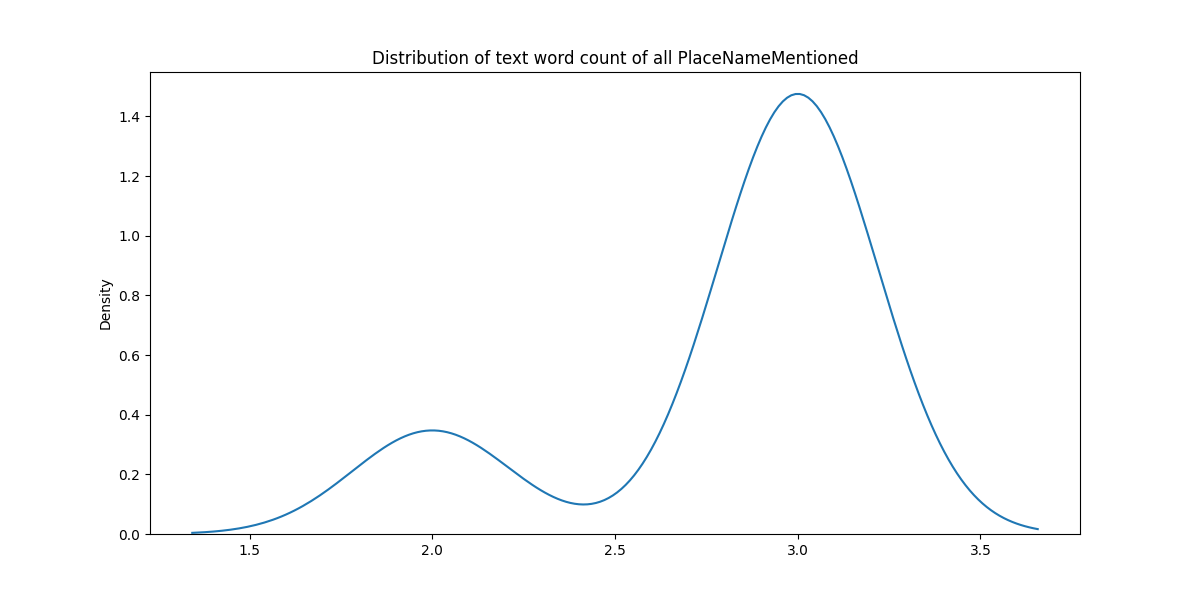
第一部分为：PGM编程部分

第二部分为：模型带先验概率的推断

第三部分为：基于机器学习相似性去重

首先我们看第一部分，我们选择了placename.csv 数据集来做贝叶斯清洗的实验。



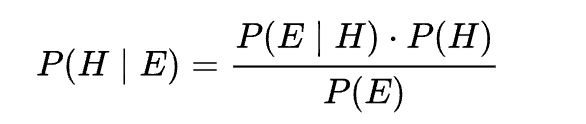
我们先进行啦数据探索： 然后我们开始概率编程：

概率编程语言（主要使用贝叶斯推理），用于利用特定于数据集的知识来清理和规范化脏数据

效果：我们可以自动更正和自动完成这些值：NULL 值、错别字、重复和不一致

这部分就是介绍数据集，我们使用和Argumentative-decision-making数据集其中的hospital(clean和dirty)，

使用了类似的贝叶斯推断方法，根据概率图来补全和纠正一些错误和缺失内容。

贝叶斯反响推断理论支撑：

其中

表示将某[事件](https://zh.wikipedia.org/wiki/%E6%9D%A1%E4%BB%B6%E6%A6%82%E7%8E%87)成立作为条件（因此表示“假定 B 事件成立下，A 事件发生”）

表示假说，其概率可能会受[实验数据](https://zh.wikipedia.org/wiki/%E5%AF%A6%E9%A9%97%E6%95%B8%E6%93%9A)（以下会称为证据）影响。一般来说会有许多互相矛盾的假说，任务是要确认哪一个假说可能性最高。

表示证据。证据对应新的数据，也就是还没用来计算先验概率的数据。

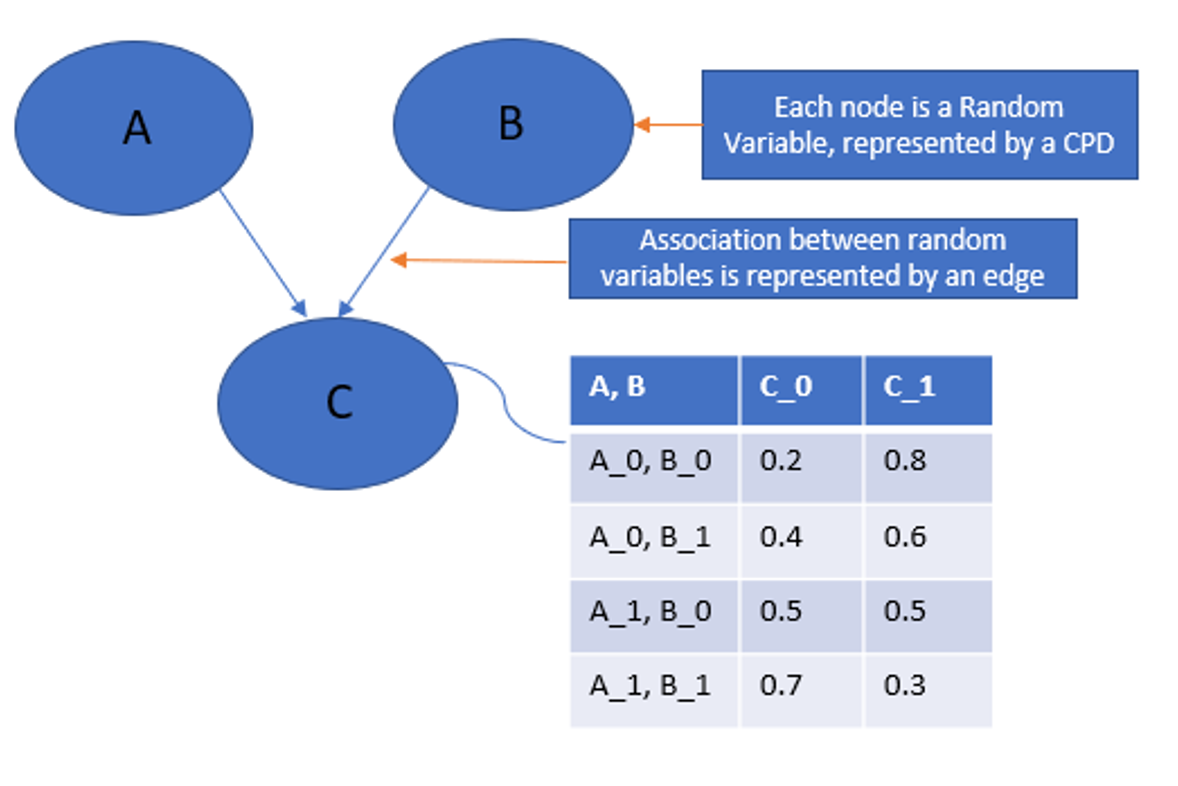
，[先验概率](https://zh.wikipedia.org/wiki/%E5%85%88%E9%AA%8C%E6%A6%82%E7%8E%87)，是观察到数据（目前证据）之前，假说的概率。

，[后验概率](https://zh.wikipedia.org/wiki/%E5%90%8E%E9%AA%8C%E6%A6%82%E7%8E%87)，是在给定证据 之后，假说的概率，是希望求得的资讯，也就是在有目前证据时，假说的概率。

是假定 成立时，观察到的概率。在不变时，这是的函数，也是[似然函数](https://zh.wikipedia.org/wiki/%E4%BC%BC%E7%84%B6%E5%87%BD%E6%95%B0)，指出在给定假设下假说和证据的相容程度。似然函数是证据的函数，而后验概率是假说的函数。

有时会称为[边缘似然率](https://zh.wikipedia.org/w/index.php?title=%E9%82%8A%E7%B7%A3%E4%BC%BC%E7%84%B6%E7%8E%87&action=edit&redlink=1)。此系数对所有可能的假说都是定值，因此在判断不同假说的相对概率时，不会用到这个系数中。

贝叶斯推断最关键的点是可以利用贝斯定理结合新的证据及以前的先验概率，来得到新的概率（这和[频率学派推断](https://zh.wikipedia.org/wiki/%E9%A2%91%E7%8E%87%E5%AD%A6%E6%B4%BE%E6%8E%A8%E6%96%AD)相反，频率论推断只考虑证据，不考虑先验概率）。



贝叶斯概率图推断

现实世界中的案例更加复杂从而会从更多信息中推断出更准确的结果。其实关于证据搜索这部分，大家还可以做的更多，形成一个更加复杂的模型，然后最后再根据已观察的变量来推断出我们想要的结果。

通常，概率图形模型使用基于图形的表示作为对多维空间上的分布进行编码的基础，而图形是在特定分布中保持的一组独立性的紧凑或分解表示。

通常使用分布的图形表示的两个分支，即两个族都包含因式分解和独立性的属性，但它们可以编码的独立性集和它们诱导的分布的因式分解不同。

机器学习的一个核心任务是从观测到的数据中挖掘隐含的知识，而概率图模型是实现这一任务的一种很elegant，

principled的手段。PGM巧妙地结合了图论和概率论。从图论的角度，PGM是一个图，包含结点与边。结点可以分为两类：隐含结点和观测结点。边可以是有向的或者是无向的，分别对应于有向图和无向图。我们构建一个图，用观测结点表示观测到的数据，用隐含结点表示潜在的知识，用边来描述知识与数据的相互关系，最后获得一个概率分布。

给定概率分布之后，通过进行两个任务

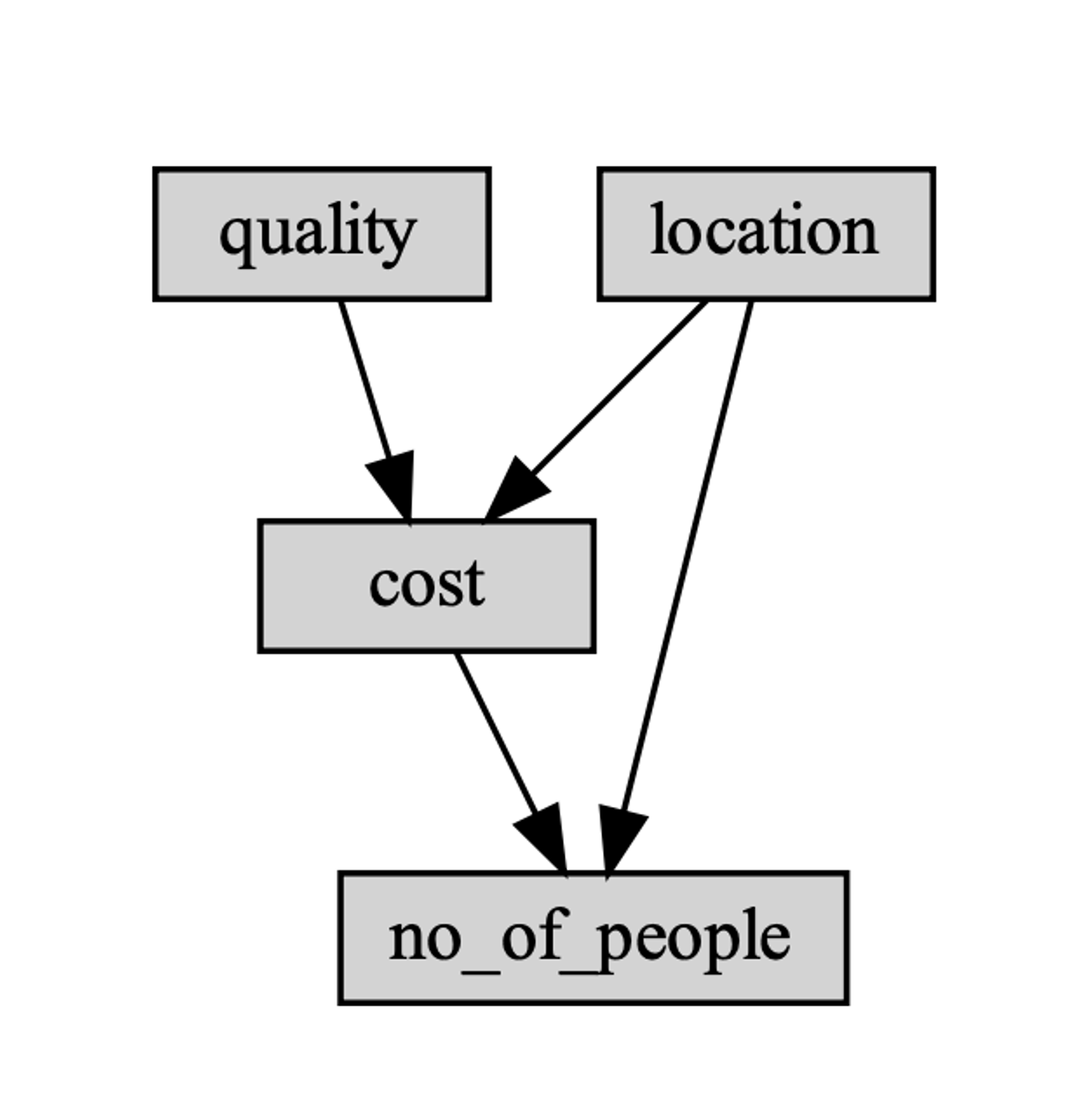
inference 给定观测结点，推断隐含结点的后验分布

learning 学习这个概率分布的参数，来获取知识。

第二部分为：模型带先验概率的推断

从模型中推断与找到一些变量的条件概率分布相同，即 P (Y | E = e ) ，其中 Y ⊆ χ 和 E ⊂ χ 。 此外，如果我们考虑预测新数据点的值，我们基本上是在尝试找到

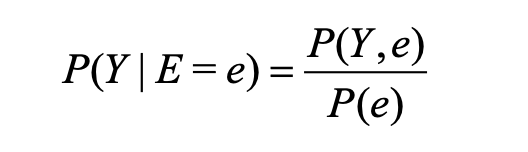
给定其他变量的观测值，未知变量的条件概率。 这些条件分布可以很容易地从变量的联合概率分布中计算出来，通过在变量和状态上边缘化和减少它们。

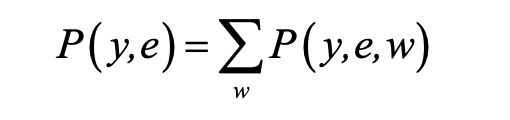


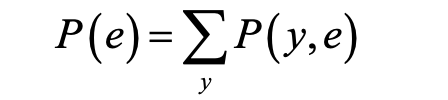
让我们再考虑一下不同地点和服务质量的医院的例子，如上图所示：我们可以考虑可以在模型上尝试的各种推理查询。 例如，假设位置好、成本高、数量多，我们可能想找到一家餐馆质量好的概率。

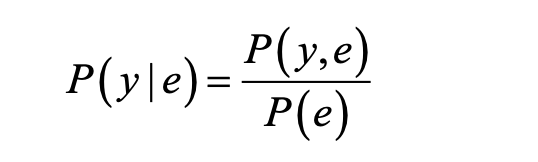
病人们可能来就诊的概率也高，这将导致概率查询：

P(Q = good | L= good,C = high,N = high)

另外，如果我们考虑一个机器学习问题：如果我们想在给定其他变量的情况下预测来医院的人数，这将只是对模型的推理查询，而具有更高概率的状态将是模型的预测。 现在，让我们看看如何从模型中计算这些条件概率。







对随机变量 Y 的每个状态 y 执行类似的计算，我们可以计算 Y 上的条件分布，给定 E = e。

对于给定联合分布时的变量，我们看到了如何找到条件分布。然而，计算联合概率分布会给我们一个指数级的大表，避免这些大表是引入概率图模型的重点。我们将讨论在计算条件分布时可以帮助我们避免完整概率分布的各种算法，但首先，让我们看看计算这些推论的复杂性是什么。

如果我们考虑最坏的情况，我们无法避免图形模型中表格的指数大小，这使得推理成为一个 NP-hard 问题，不幸的是，即使是计算条件分布的近似方法也是 NP-hard。这些结果的证明超出了本书的范围。

但是，这些结果是针对最坏情况的。在现实生活中，我们并不总是遇到最坏的情况。因此，让我们讨论一下用于推理的各种算法，通过它们进行变量消除：

让我们尝试在图 3.1 中的地点网络上做一些推理任务。假设我们要找到 P(C)。我们从概率的链式法则知道以下内容：

P ( C ) = ∑ P ( C | l , q )P ( l , q ) l,q

此外，我们知道如果没有观察到 C，随机变量 L 和 Q 是相互独立的。因此，我们可以将前面的等式写成如下：

P ( C ) = ∑ P ( C | l , q )P ( l ) P ( q ) l,q

现在，我们可以看到我们知道产品中涉及的概率值

P(C)的计算。我们从 C 的 CPD 得到 P(C |l,q) 的值，从 L 的 CPD 得到 P(l) 的值，从 Q 的 CPD 得到 P(q) 的值。在这些概率中，我们可以很容易地找到 C 的概率。

我们还可以注意到，此计算的计算成本将是

, 其中 V a l ( X ) 表示的状态数

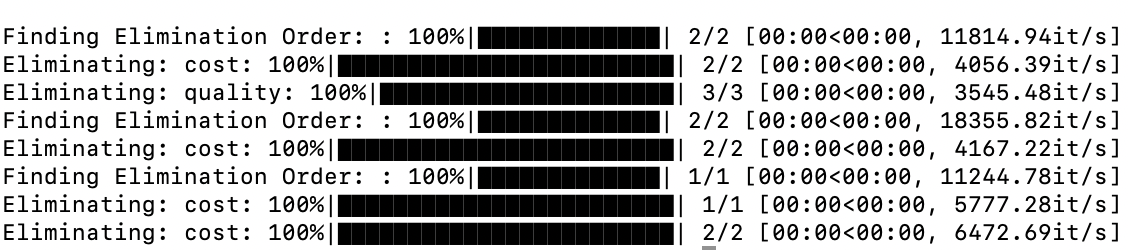
变量 X。我们可以看到，为了计算 C 的每个状态的概率，我们需要计算状态 L 和 Q 的每个组合的乘积，然后将它们相加。这意味着对于 C 的每个状态，我们有 2∗Val(L)∗Val(P) 乘积和 (Val(L)∗Val(Q))-1 个加法。这里，2 出现在产品的数量中，因为等式中有两个产品操作。此外，我们需要为 C 的每个状态计算 Val(C) 次。

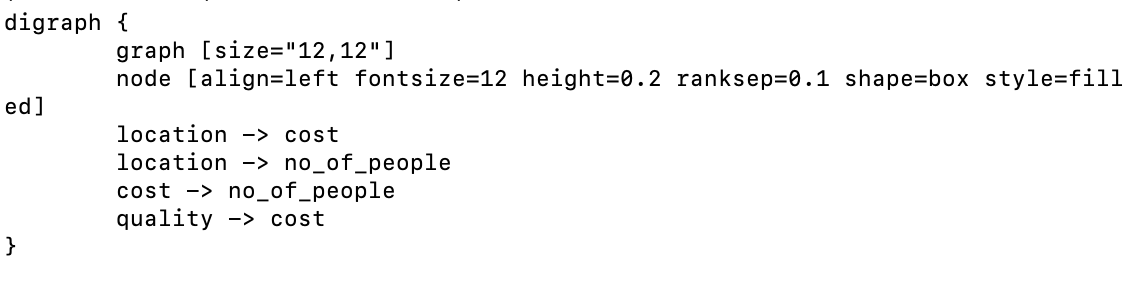
现在，让我们以另一个简单模型 A → B → C → D 为例，尝试找到 P(D)。我们可以简单地找到 P(D)：

P(D)∑P(a)P(b|a)P(c|b)P(D|c)

*P*(*d*0 )= *P*(*a*0 )*P*(*b*0 |*a*0 )*P*(*c*0 |*b*0 )*P*(*d*0 |*c*0 )+ *P*(*a*0 )*P*(*b*0 |*a*0 )*P*(*c*1 |*b*0 )*P*(*d*0 |*c*1)+ *P*(*a*0 )*P*(*b*1 |*a*0 )*P*(*c*0 |*b*1)*P*(*d*0 |*c*0 )+ *P*(*a*0 )*P*(*b*1 |*a*0 )*P*(*c*1 |*b*1)*P*(*d*0 |*c*1)+ *P*(*a*1)*P*(*b*0 |*a*1)*P*(*c*0 |*b*0 )*P*(*d*0 |*c*0 )+ *P*(*a*1)*P*(*b*0 |*a*1)*P*(*c*1 |*b*0 )*P*(*d*0 |*c*1)+ *P*(*a*1)*P*(*b*1 |*a*1)*P*(*c*0 |*b*1)*P*(*d*0 |*c*0 )+ *P*(*a*1)*P*(*b*1 |*a*1)*P*(*c*1 |*b*1)*P*(*d*0 |*c*1)

我们已经看到变量消除操作的计算复杂度如何与消除顺序的选择相关，以及这与诱导图。 与具有较高树宽度的消除顺序相比，较小的树宽度可确保更好的复杂性。 所以，我们的问题现在已经简化为选择一个保持树宽度最小的消除顺序。找到最小树宽度的消除是 NP 完全问题，没有简单的方法通过简单地查看网络结构来找到网络推理的复杂性。 但我们可以使用许多其他技术来找到良好的消除顺序：使用诱导图的弦图属性，如果一个图不包含长度大于 3 的环，并且每个环的两个不相邻节点之间没有边，我们将其定义为弦图。 换句话说，弦图中的每个最小循环都是三个长度。你会发现每个诱导图都是一个弦图。 此外，该定理的反面成立，即这些变量上的每个弦图都对应于一些消除顺序。为了找到消除顺序，我们使用最大基数搜索算法。 在这个算法中，我们基本上迭代 χ 次，并且在每次迭代中，我们尝试找到具有最多标记变量的变量，然后标记该变量，这可以消除排序。





第三部分为：基于机器学习相似性去重

衡量两个句子向量相似度的两种基本的思路：一是比较二者的欧式距离，越小代表二者越相似；二是比较二者的余弦值，越大代表二者越相似。KMeans聚类天然的衡量标准就是欧式距离最小化，那就先从第一个衡量标准进行尝试，这里使用的是sklearn的包。

聚类的个数是通过聚类自身的类里相似度、类间相似度、sklearn用于评价聚类结果的silhouette\_score, calinski\_harabaz\_score, adjusted\_rand\_score综合比较过选择得到的。

我们主要是用kmean机器学习去掉数据集中可能重复的数据records，减少人手工检查和去重的机械工作的工作量。

具体Kmean流程如下：

K-Means聚类首先随机确定 K 个初始点作为质心（这也是K-Means聚类的一个问题，这个K值的不合理选择会使得模型不适应和解释性差）。然后将数据集中的每个点分配到一个簇中， 具体来讲，就是为每个点找到距其最近的质心（这里算的为欧式距离，当然也可以使用其他距离）， 并将其分配该质心所对应的簇；这一步完成之后，每个簇的质心更新为该簇所有点的平均值；重复上述过程直到数据集中的所有点都距离它所对应的质心最近时结束。

创建 k 个点作为起始质心（随机选择）

当任意一个点的簇分配结果发生改变时（不改变时算法结束）

对数据集中的每个数据点

对每个质心

计算质心与数据点之间的距离

将数据点分配到距其最近的簇

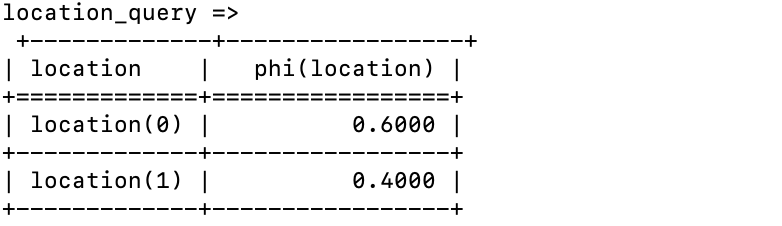
对每一个簇,计算簇中所有点的均值并将均值作为质心

在相似度比较训练过程：两个重要的环节对结果影响也十分重要：1.句子向量化是否合理。2.句子相似度的衡量标准是否合理。

对于第一点，TfidfVectorizer可以把原始文本转化为tf-idf的特征矩阵，从而为后续的文本相似度计算，主题模型，文本搜索排序等一系列应用奠定基础。将文本文档集合转换为计数的稀疏矩阵。内部的实现方法为调用scipy.sparse.csr\_matrix模块。并且，如果在调用CountVectorizer() 时不提供先验词典并且不使用执行某种特征选择的分析器，则特征词的数量将等于通过该方法直接分析数据找到的词汇量。

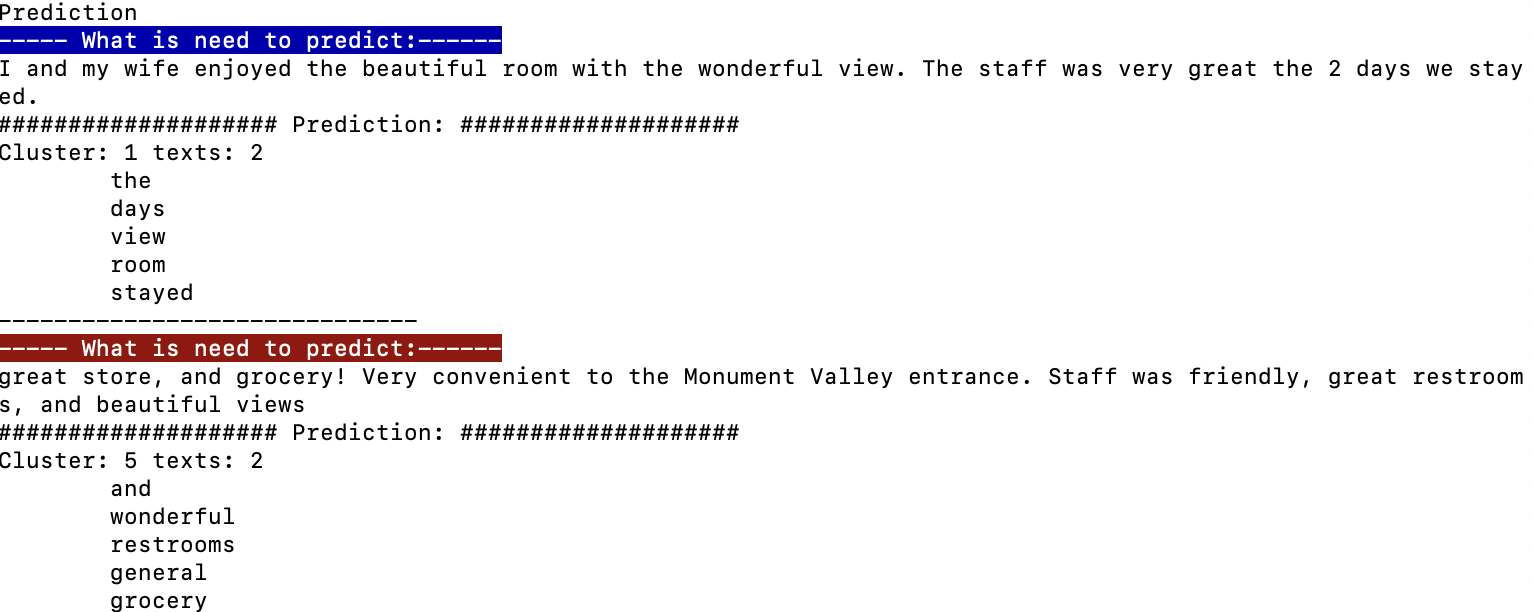
4.方法结果

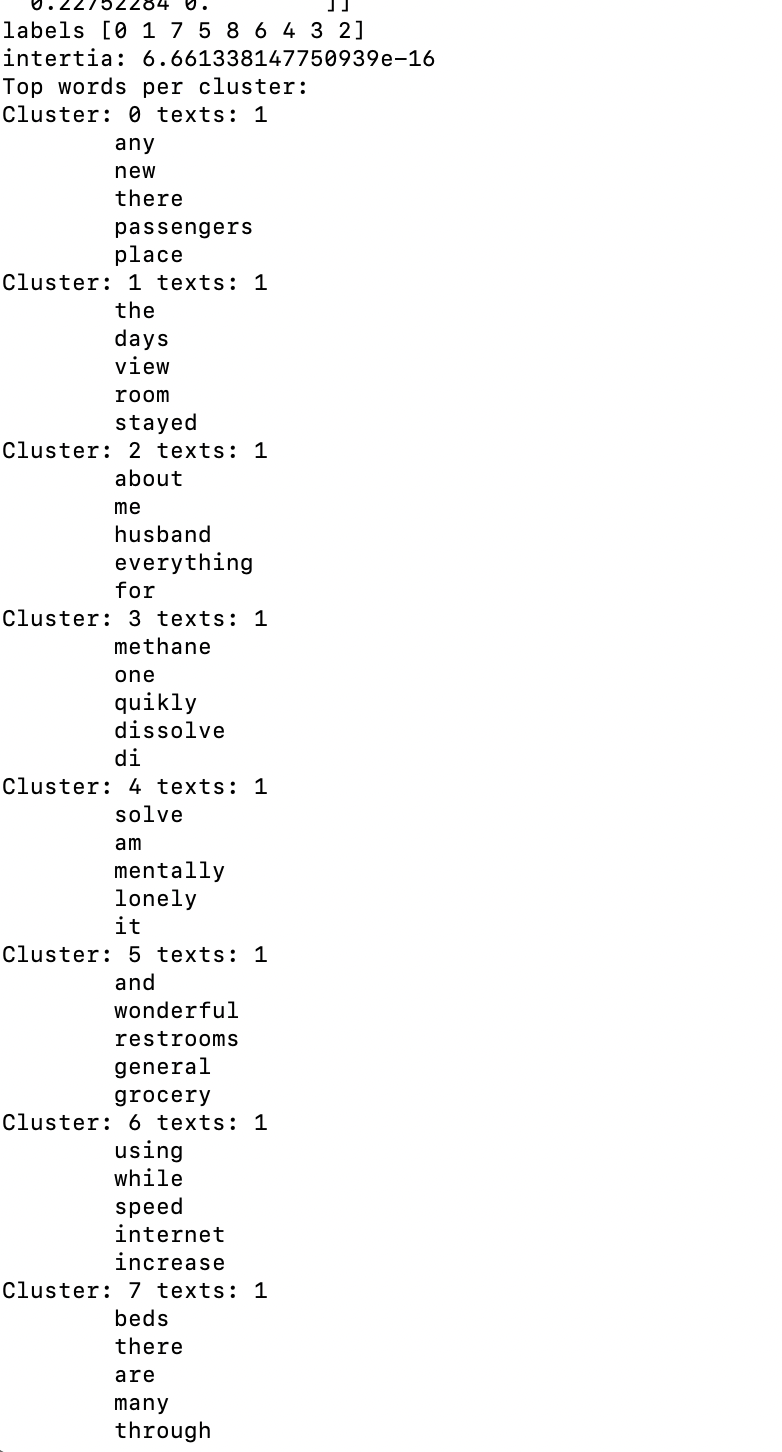
第1部分的运算结果：



因为这块是先验的概率分布，理论上贝叶斯定理起作用，我们就相信我们的结果是符合概率分布的。

第2部分的预测结果：





通过人工检查和核验，我们可以我们的预测句子是候选句子中最准确的。

5.结论

5.1 总结

通过实现Argumentative-decision-making 使得将混乱、不一致的数据集连接到干净的记录中变得更容易，而不需要像人工处理，减少人为因素干扰数据，在给予贝叶斯概率编程和机器学习帮助去重评论数据，这样机器学习和概率编程就能给决策系统提供更多不依赖人为的决策因子，这有潜在的社会利益：减少了枯燥无味的工作，节约了人工人力，有淡出靠人的经验和常识的风险。

有风险的方面：本决策系统，如果通过将来自多个公共来源的不完整信息联合起来，Argumentative-decision-making 可能会让侵犯人们隐私的成本更低、更容易，甚至可能去匿名化，因为很多隐藏在数据、人很难单独分析出来的概率分布和知识推断，会从背景中的原因会被决策系统分析出来。

5.2 未来工作展望

使用更多的机器学习和概率编程结合，进一步减少人为的干预。比如可以考虑和知识图谱进、图神经网络，进一步提升知识库在推理、去重、清理脏数据，推断更准确的概率知识分布和更自动化的决策因素分析。