李航统计学习：

第四章讨论部分：

读书报告内容：

1. （必填）自己提出的问题的理解（罗列全部）：
2. 提出的问题1：:拉普拉斯修正是否有可能改变原先的预测结果？

讨论后的理解：

首先拉普拉斯修正可以有效的避免因训练样本不足而导致概率估值为零的问题，并且在训练集变大的时候，修正所引入的先验的影响也会逐渐变得可忽略，使得估值趋向于实际概率值

并且因为拉普拉斯引入的分母n是类别c的个数，使得修正较为平滑，所以拉普拉斯修正是比较优秀的。

1. 提出的问题2：

朴素贝叶斯算法的优缺点有哪些？

朴素贝叶斯的主要优点有：

1）朴素贝叶斯模型发源于古典数学理论，有稳定的分类效率。

2）对小规模的数据表现很好，能个处理多分类任务，适合增量式训练，尤其是数据量超出内存时，我们可以一批批的去增量训练。

3）对缺失数据不太敏感，算法也比较简单，常用于文本分类。

朴素贝叶斯的主要缺点有：

1） 理论上，朴素贝叶斯模型与其他分类方法相比具有最小的误差率。但是实际上并非总是如此，这是因为朴素贝叶斯模型假设属性之间相互独立，这个假设在实际应用中往往是不成立的，在属性个数比较多或者属性之间相关性较大时，分类效果不好。而在属性相关性较小时，朴素贝叶斯性能最为良好。对于这一点，有半朴素贝叶斯之类的算法通过考虑部分关联性适度改进。

2）需要知道先验概率，且先验概率很多时候取决于假设，假设的模型可以有很多种，因此在某些时候会由于假设的先验模型的原因导致预测效果不佳。

3）由于我们是通过先验和数据来决定后验的概率从而决定分类，所以分类决策存在一定的错误率。

4）对输入数据的表达形式很敏感。

二、（必填）别人提出的问题的理解（选择几个问题罗列，并给出理解）：

问题3：多项式模型和伯努利模型在实现上有什么区别，如何评价优劣？

自己的理解：

伯努利模型并不考虑词在文档中出现的次数，只考虑出不出现，在这个意义上相当于假设词是等权重的。多项式模型考虑此在文档中的出现次数。二者的计算粒度不一样，多项式模型以单词为粒度，伯努利模型以文件为粒度。当训练集文本较短时，我们更倾向于使用贝努利模型。而文本较长时，我们更倾向于多项式模型，因为，在一篇文档中的高频词，会使该词的似然概率值相对较大。这两种方法是用来处理离散数据的，对于连续型数据采用高斯模型。

问题4：为什么说朴素贝叶斯是稳定的算法？

自己的理解：

因为朴素贝叶斯是基于古典数学理论，使用已经获得的数据的先验概率，通过先验概率来计算出后验概率，以此对一条数据进行预测，得到需要的与该数据所对应的类别，因为先验概率是固定的，计算的方法和路径的都是固定的，因此每次计算所得的后验概率都是相同的，所以这是一种稳定的方法。

问题5：贝叶斯和朴素贝叶斯的区别？

自己的理解：

1）贝叶斯方法是以贝叶斯原理为基础，使用概率统计的知识对样本数据集进行分类。由于其有着坚实的数学基础，贝叶斯分类算法的误判率是很低的。贝叶斯方法的特点是结合先验概率和后验概率，即避免了只使用先验概率的主管偏见，也避免了单独使用样本信息的过拟合现象。贝叶斯分类算法在数据集较大的情况下表现出较高的准确率，同时算法本身也比较简单。

2）朴素贝叶斯算法

朴素贝叶斯算法（Naive Bayesian）是应用最为广泛的分类算法之一。朴素贝叶斯方法是在贝叶斯算法的基础上进行了相应的简化，即假定给定目标值时属性之间相互条件独立。也就是说没有哪个属性变量对于决策结果来说占有着较大的比重，也没有哪个属性变量对于决策结果占有着较小的比重。虽然这个简化方式在一定程度上降低了贝叶斯分类算法的分类效果，但是在实际的应用场景中，极大地简化了贝叶斯方法的复杂性。

1. （必填）读书计划
2. 本周完成的内容章节：

（1）第四章完成并且梳理知识点，寻找问题，自己思考并且于小组会之前完成了自己的思考。

2、下周计划：第五章的阅读.

四、（选做）读书摘要及理解或伪代码的具体实现（读书摘要、伪代码的具体实现代码等可以写到这个部分）

基于朴素贝叶斯的文本 分类：

def loadDataSet():

postingList=[['my', 'dog', 'has', 'flea', 'problems', 'help', 'please'],

['maybe', 'not', 'take', 'him', 'to', 'dog', 'park', 'stupid'],

['my', 'dalmation', 'is', 'so', 'cute', 'I', 'love', 'him'],

['stop', 'posting', 'stupid', 'worthless', 'garbage'],

['mr', 'licks', 'ate', 'my', 'steak', 'how', 'to', 'stop', 'him'],

['quit', 'buying', 'worthless', 'dog', 'food', 'stupid']]

classVec = [0,1,0,1,0,1]#1 侮辱性文字 ， 0 代表正常言论

return postingList,classVec

def createVocabList(dataSet):#创建词汇表

vocabSet = set([])

for document in dataSet:

vocabSet = vocabSet | set(document) #创建并集

return list(vocabSet)

def bagOfWord2VecMN(vocabList,inputSet):#根据词汇表，讲句子转化为向量

returnVec = [0]\*len(vocabList)

for word in inputSet:

if word in vocabList:

returnVec[vocabList.index(word)] += 1

return returnVec

def trainNB0(trainMatrix,trainCategory):

numTrainDocs = len(trainMatrix)

numWords = len(trainMatrix[0])

pAbusive = sum(trainCategory)/float(numTrainDocs)

p0Num = ones(numWords);p1Num = ones(numWords)#计算频数初始化为1

p0Denom = 2.0;p1Denom = 2.0 #即拉普拉斯平滑

for i in range(numTrainDocs):

if trainCategory[i]==1:

p1Num += trainMatrix[i]

p1Denom += sum(trainMatrix[i])

else:

p0Num += trainMatrix[i]

p0Denom += sum(trainMatrix[i])

p1Vect = log(p1Num/p1Denom)

p0Vect = log(p0Num/p0Denom)

return p0Vect,p1Vect,pAbusive#返回各类对应特征的条件概率向量

#和各类的先验概率

def classifyNB(vec2Classify,p0Vec,p1Vec,pClass1):

p1 = sum(vec2Classify \* p1Vec) + log(pClass1)#注意

p0 = sum(vec2Classify \* p0Vec) + log(1-pClass1)#注意

if p1 > p0:

return 1

else:

return 0

def testingNB():

listOPosts,listClasses = loadDataSet()#加载数据

myVocabList = createVocabList(listOPosts)#建立词汇表

trainMat = []

for postinDoc in listOPosts:

trainMat.append(setOfWords2Vec(myVocabList,postinDoc))

p0V,p1V,pAb = trainNB0(trainMat,listClasses)#训练

#测试

testEntry = ['love','my','dalmation']

thisDoc = setOfWords2Vec(myVocabList,testEntry)

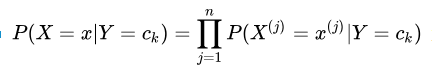
print testEntry,'classified as: ',classifyNB(thisDoc,p0V,p1V,pAb)

读书摘要：

朴素贝叶斯法是一种多分类算法，它的基础是“朴素贝叶斯假设”（假设实例的各个特征具有条件独立性）。根据训练集估计模型的先验概率、条件概率，再按照后验概率最大化的准则，给出输入实例的分类预测。它的算法实现很简单，但理论证明并不容易。具体来说，通过极大似然估计法估计先验概率、条件概率，计算过程比较复杂。本章主要分为3个部分：

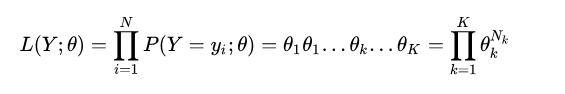
朴素贝叶斯分类器，介绍它的基本假设；

基本假设即为：



在 Y=Ck 的条件下，特征向量 X=x 的概率等于它们各个分量同时相等的概率。朴素贝叶斯假设（又称特征条件独立假设）的目的在于简化计算，但实际上特征之间有可能是相关的，所以此假设有可能导致误差增大。

先验概率、条件概率的极大似然估计；



注意，这里隐含两个约束条件：

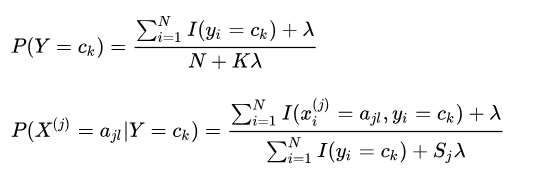
（1）各种类别比例之和等于1；

（2）各种类别的实例数相加等于实例总数。

贝叶斯估计与拉普拉斯平滑。

如果在训练集里面，某一类别的实例数为0，将导致条件概率的分母为0 。

为解决这个问题，引入拉普拉斯平滑，增加一个正数lamda>0，使得



上问题一证明了拉普拉斯平滑在数据量很大的情况下最终会收敛于lamda=0

知识图谱部分：

工作报告内容：

本周主要做了两方面的工作

1.一个是安卓开发的自学为吴老师项目做安卓做准备，之前没有做过安卓开发，只能从头自学，看了《Android第一行代码》，看到了50页作业，进度大概在活动组件。

2.做了type inference方向方法的调研，因为新冠域数据方面主要是rdf三元组和已经完成的entity\_type键值对。在跟吴老师讨论后，重点看了TransE模型进行type inference的推断。

TransE

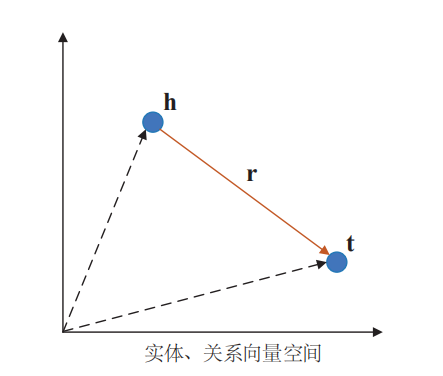
TransE 的核心思想源于 word2vec 训练出的词向量在向量空间中存在的平移不变性

质，例如：



这里 V (w) 表示利用 word2vec 学习得到的单词 w 的词向量，也就是说，词向量能够捕

捉到单词 king 和 queen 之间、man 和 woman 之间共同存在的某种潜在语义关系。



对于每个三元组 (h, r, t)，TransE 用关系 r 的向量 r 作为头实体向量 h 和尾实

体向量 t 之间的转移操作。

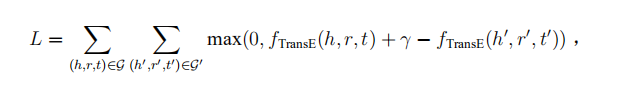
如图2.1所示，对于每个三元组 (h, r, t)，TransE 希望 h + r ≈ t，因此，三元组的评

分函数定义为：

fTransE(h, r, t) = −∥h + r U t∥

在学习过程中，为了增强知识表示的区分能力，TransE 采用了最大间隔法，定义了

以下目标函数：



其中，G 是合法三元组 (正样本) 的集合，即用于训练的三元组集合，G′ 为错误的三元组

(负样本) 的集合，γ 为正样本的评分与负样本的评分之间的间隔。该函数希望负样本的

评分与正样本的评分之间的差值大于间隔 γ，否则将 γ 与该差值之差加入损失，用来更

新参数，使得负样本与正样本的评分之差进一步增大，从而提升模型对正负样本的区分

能力。

负样本通过将正样本的头实体或尾实体等可能地随机替换为另一个实体得到，即

G′ = {(h′ , r, t)|h′ ∈ E} ∪ {(h, r, t′)|t′ ∈ R}，其中 E 表示知识图谱中所有实体的集合，R

表示知识图谱中所有关系的集合。尽管这样构造出来的负样本可能是实际成立的三元

组，但 TransE 认为这样的概率很小，对模型的影响也很小。

训练时，给定向量维数 k，TransE 首先随机初始化两个矩阵作为整个模型的参数，

E ∈ Rm×k 表示实体的向量矩阵，每一行对应一个实体的向量，在训练过程中可以通过

实体的编号从中进行检索；R ∈ R|n|×k 表示关系的向量矩阵，每一行对应一个关系的向

量。然后，对于每一小批 (mini-batch) 数据，TransE 为其中每个正样本构造一个负样本，

并利用目标函数计算损失，更新参数。

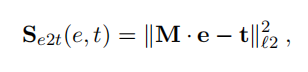
通过这个模型：

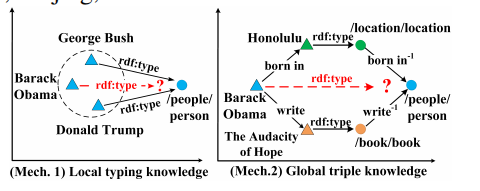
我自己的想法是：

讲每个entity\_type键值对当作一个rdf三元组（entity，IsA，type），将之前获得的先验数据作为正例，再随机用错误的type构建负例进行训练。在跟吴老师讨论之后，吴老师指出这个方法一定结果不会太好。后来我又寻找了别的方法：

ConnectionE模型

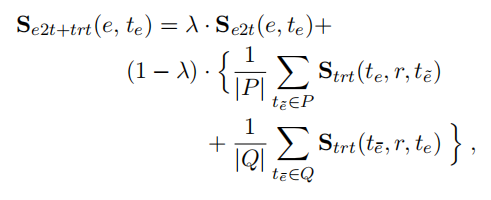
模型的核心思想是：其实和之前的TransE很像。基于embeding，一种方法是，embeding后较好的低维向量聚类是和type有一定语义关联的，然后训练数据集得到矩阵m，就能获得entity的type，方法过程如图1所示。

通过函数：计算相似度。

另一种方法的思想是，type替换掉本来的entity所对应的rdf应该是 仍然成立的。那么有如图所示的

Mech.2成立，即为

最后通过函数计算两个模型综合考虑的置信度：



这个模型和吴老师讨论过后，吴老师也认为可以试试。也能在github上找到源码。

但是存在如下问题 ：

1. 源码太老了，实际下载跑出有很多问题，正在调试。作者自己说不是最新的版本，跑出大量报错，我在改改，试试能不能用。
2. 该算法的trt模型需要尾实体的type进行推断，现有的数据只有头实体的type，需要重新做出尾实体的entity\_type的键值对。

下周工作内容：

1. 继续推进安卓开发的学习
2. 把他原算法跑起来
3. 重新做出尾实体的entity\_type键值对。