情感分析入门学习文档

2019 年 11 月 28 日 陈禹豪

一、基于情感词典方法的情感分析

情感词典:

完备程度对情感分析的准确度起决定性作用

（1）直接将常用词赋予唯一分数的情感词典

如 BosonNLP 是基于微博、新闻、论坛等数据来源构建的情感词典

（2）分词性的情感词典

积极、消极、否定、程度副词、停用词、转折词、特殊标点符号词典

识别准确率不高的原因:

（1）语句中含有未登录词

分词时如果出现分词词典中的未登录词，会导致分词错误；情感分析时如果出现情感词典中的未登录词，会影响情绪判别结果。

两种可能的解决方法:

未登录词识别，通过相似性计算获取未登录词的词义

基于语义词典（同义词或反义词）或语料对情感词典的扩充

（2）中文语境情感表达与词性多变

同一个词在不同的语境下可以是代表完全相反的情感意义（说反话以加强表达效果）：

世界上就只有两种人能吸引人，一种是特漂亮的一种就是他这样的（反话正说）

我对你有意见，你太不爱惜自己身体，工作起来太玩命了（正话反说）

同一个词可作多种词性，情感分数也不应唯一相同：

这部宣传片真垃圾 （形容词）

这部宣传片是关于垃圾分类的 （名词）

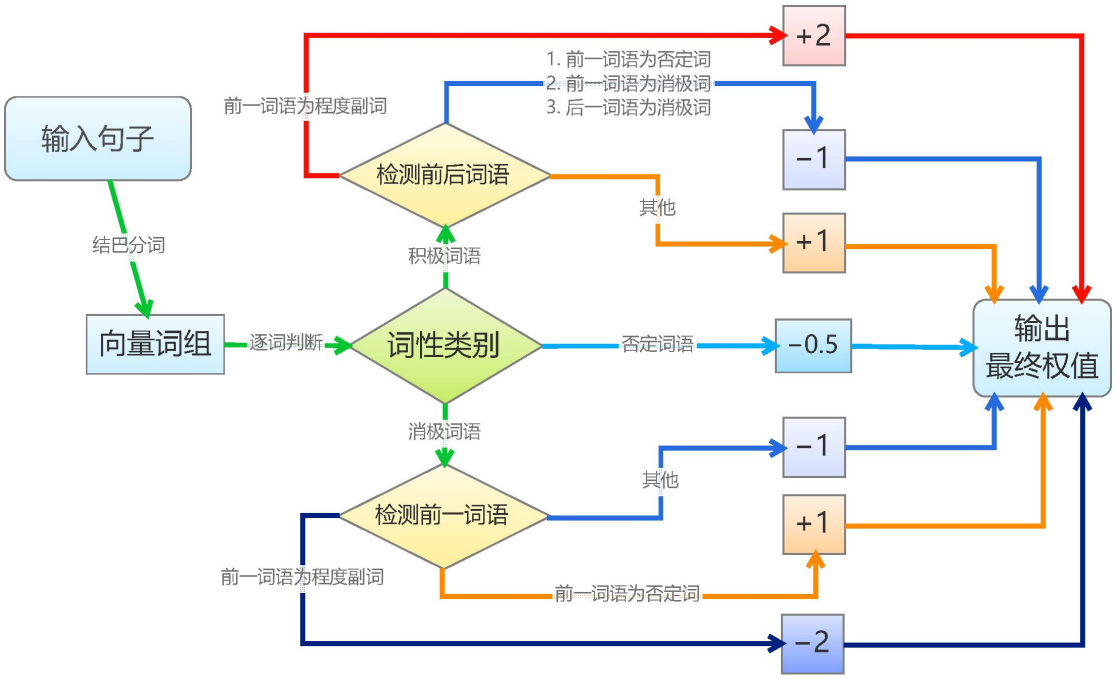
不同语气（标点符号）导致情感表达不同：

我喜欢吃香菜!

我喜欢吃香菜?

可能的解决方法：

通过对积极词、消极词、程度副词、转折词、特殊标点符号的复杂组合进行更为复杂的逻辑判断综合得出情感倾向



二、基于机器学习方法的情感分析

word2vec 的实现过程：

w2v = word2vec.Word2Vec(x,size=500, min_count=20)

通过以上语句将直接实现以下流程，也可分别按照以下流程中的语句进行实现：

1、建立一个空的模型对象：model = gensim.models.Word2Vec()

2、遍历一次语料库建立词典：model.build_vocab(sentences)

3、第 2 次遍历语料库建立神经网络模型：model.train(sentences)

reshape 的作用：

vec = np.zeros(200).reshape((1,200))

通过 np.zeros(200)将生成一个含有 200 个 0 的 array 数组：array([0,0,...,0])

reshape 后将变为：array([[0,0,...,0]])，以便实现将整个评论数据集中的各句评论的词向量和整合到一个 array 中，如下：

```
In [31]: train_vec

Out[31]: array([[-1.44576117,  0.14883759,  0.67165024, ...,  0.04589583,
                  1.75241445,  2.72302817],
                [-0.44684723,  0.81857665, -0.68466676, ..., -0.42288736,
                  1.77500253,  3.67936396],
                [ 1.66910548,  3.46994659,  0.65798305, ..., -0.17190895,
                  0.84573039,  6.62223567],
                ...,
                [ 1.65155563,  0.88857006,  3.01813673, ..., -0.92687118,
                  3.25119123,  3.41844247],
                [ 1.85063745,  3.41633293, -0.15270161, ...,  3.12313088,
                  3.11055096,  2.05659477],
                [ 0.71947383,  0.84346028,  2.15408291, ...,  0.77715345,
                  1.55879849,  0.35928102]])
```

2.1 支持向量机

初始化支持向量机模型：

svm_model = svm.SVC(kernel="rbf")

参数设置解释：

kernel="rbf"：

核函数选择**径向基核函数/高斯核函数**（Radial Basis Function)

（1）常用核函数：

线性核函数 kernel="linear"/多项式核函数 kernel='poly'/sigmod 核函数 kernel='sigmod'

（2）核函数如何选择：

SVM 核函数的选择对于其性能的表现有至关重要的作用，尤其是针对那些线性不可分的数据，因此核函数的选择在 SVM 算法中就显得至关重要。

线性核，主要用于线性可分的情况，我们可以看到特征空间到输入空间的维度是一样的，其参数少速度快，对于线性可分数据，其分类效果很理想，因此我们通常首先尝试用线性核函数来做分类，看看效果如何，如果不行再换别的

多项式核函数可以实现将低维的输入空间映射到高纬的特征空间,但是多项式核函数的参数多，当多项式的阶数比较高的时候，核矩阵的元素值将趋于无穷大或者无穷小，计算复杂度会大到无法计算。

高斯径向基函数是一种局部性强的核函数，其可以将一个样本映射到一个更高维的空间内，该核函数是应用最广的一个，无论大样本还是小样本都有比较好的性能，而且其相对于多项式核函数参数要少,因此大多数情况下在不知道用什么核函数的时候,优先使用高斯核函数。

采用 sigmoid 核函数，支持向量机实现的就是一种多层神经网络。

因此，在选用核函数的时候，如果我们对我们的数据有一定的先验知识，就利用先验来选择符合数据分布的核函数；如果不知道的话，通常使用交叉验证的方法，来试用不同的核函数，误差最下的即为效果最好的核函数，或者也可以将多个核函数结合起来，形成混合核函数。

（3）吴恩达对核函数选择的理解

如果特征的数量大到和样本数量差不多，则选用 LR 或者线性核的 SVM；

如果特征的数量小，样本的数量正常，则选用 SVM+高斯核函数；

如果特征的数量小，而样本的数量很大，则需要手工添加一些特征从而变成第一种情况

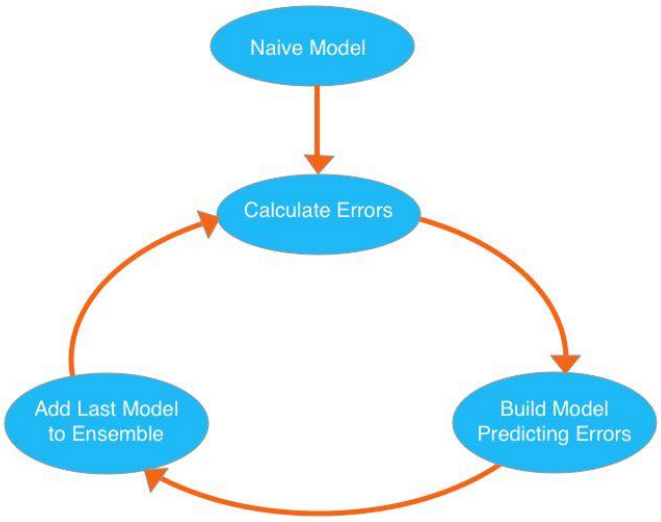
2.3 Xgboost（eXtreme Gradient Boosting）

（1）是对梯度增强决策树(GBDT)的一种实现，既可以用于分类也可以用于回归问题

（2）对 GBDT 进行了改进包括损失函数、正则化、稀疏感知算法、并行化算法设计等等，具体请参考 References 中相关文献。

GBDT 的 boosting 思想：

反复循环构建新的模型并将它们组合成一个集成模型的，从初始 Naive 模型开始，我们从计算数据集中每个观测的误差，然后下一个树去拟合误差函数对预测值的残差，使用了梯度下降算法来最小化损失。



三、基于深度学习方法的情感分析（前向 simple RNN）

语言模型：把一段自然语言文本看作一段离散的时间序列，计算该序列的概率：

$$P(w1,w2,...,wT)$$

P(“我爱统计”)=P(w1,w2,w3,w4)

由于依次生成：P(w1,w2,w3,w4) = P(w1) P(w2 | w1) P(w3 | w1,w2) P(w4 | w1,w2,w3)

马尔可夫假设：一个词的出现只与前面 n 个词相关

只与前面一个词相关： P(w1) P(w2 | w1) **P(w3 | w2) P(w4 | w3)**

***存在的问题*:**

（1）P(“我爱统计”)=P(“我爱**计统**”)

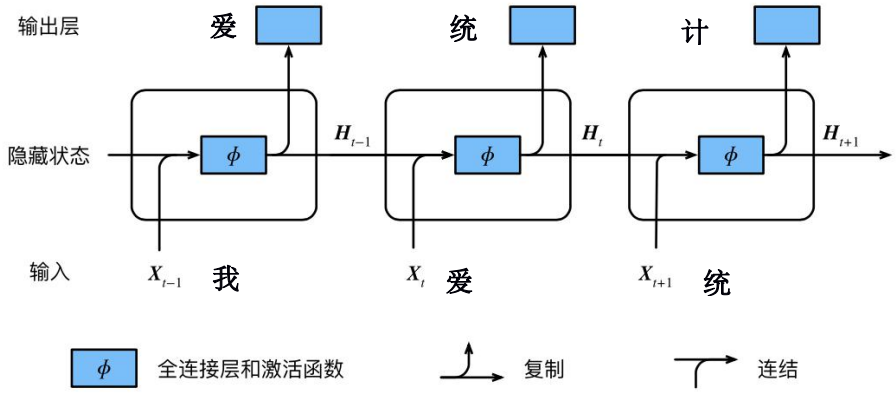
（2）n 为固定长度，增大 n 将导致模型参数呈指数型增长，计算复杂

RNN（循环神经网络 Recurrent Neural Network ）:

并非刚性地记忆所有固定长度的序列，（考虑输入数据存在时间相关性）而是通过隐藏状态来存储之前时间的信息（通过隐藏层生成隐藏变量给下一层），迭代的更新隐藏状态：

$$\boldsymbol{H}_t = \phi(\boldsymbol{X}_t \boldsymbol{W}_{xh} + \boldsymbol{H}_{t-1} \boldsymbol{W}_{hh} + \boldsymbol{b}_h).$$

前向 simple RNN（“我爱统计”）



可能存在的问题：

（1）只将语句前部分的内容不断的向后一个传递，位于句子后半部分的内容没法影响前半部分（双向 RNN）

（2）记忆期短：随着句子长度增加，隐藏状态中含有的距离很远的前面词汇影响不断减弱

（3）可能存在的梯度消失问题（LSTM）

3.1 语料库预处理

分析流程：

（1）原始文本：我喜欢商务与经济统计

（2）分词：我，喜欢，商务，与，经济统计

（3）根据**预训练词向量模型**将词语索引化 tokenize：[1, 23, 45, 678, 890]，将文本数据变为数字特征

（4）词向量化 Embedding：使用 200 维的词向量，上面变成一个（5，200）的 Embedding Matrix，需要截长补短

（5）构建 RNN 循环神经网络模型（BiRNN / LSTM / GRU）

（6）经过全连接阶层激活函数输出分类

词向量（词嵌入）预训练模型

（1）GloVe: Global Vectors for Word Representation：在 wikipedia 基于共现矩阵分解训练得到的词向量模型，400k+

https://nlp.stanford.edu/projects/glove/

（2）Chinese Word Vectors：目前最全的中文预训练词向量集合，词典包含 2600w+词语

https://github.com/Embedding/Chinese-Word-Vectors

（3）自己预训练 word2vec 模型

训练集与测试集构建：sklearn 可以随机分割训练集和测试集（交叉验证），只需要在代码中引入 model_selection 的 train_test_split（*from sklearn import model_selection*）

```
x_train, x_test, y_train, y_test = model_selection.train_test_split(
                                   x, y, test_size=0.2, random_state=12)
```

random_state 用于重复实验时保证程序每次运行都分割一样的训练集合测试集。如果输入 0 或者不输入将导致每次都不一样

Dropout：

（1）缩小神经网络规模（2）权重正则化（L1、L2）

（3）随机失活:在网络层训练期间随机失活（设置为零）该层的许多输出特征（节点），参数表示节点被随机失活的概率，一般 0.2～0.5 之间

Early Stopping： 每个 epoch 之后获取 val_acc 结果，不再改善时停止训练

参数:

monitor: 监控的数据接口，有'acc','val_acc','loss','val_loss'等等。正常情况下如果有验证集，就用'val_acc'或者'val_loss'。但是因为笔者用的是 5 折交叉验证，没有单设验证集，所以只能用'acc'了。

min_delta： 增大或减小的阈值，只有大于这个部分才算作 improvement。这个值的大小取决于 monitor，也反映了你的容忍程度。例如笔者的 monitor 是'acc'，同时其变化范围在 70%-90%之间，所以对于小于 0.01%的变化不关心。加上观察到训练过程中存在抖动的情况（即先下降后上升），所以适当增大容忍程度，最终设为 0.003%。

patience： 能够容忍多少个 epoch 内都没有 improvement。在噪声抖动和真正的准确率下降之间做 trade-off。如果 patience 设的大，那么最终得到的准确率要略低于模型可以达到的最高准确率。如果 patience 设的小，那么模型很可能在前期抖动，还在全图搜索的阶段就停止了，准确率一般很差。patience 的大小和 learning rate 直接相关。在 learning rate 设定的情况下，前期先训练几次观察抖动的 epoch number，比其稍大些设置 patience。在 learning rate 变化的情况下，建议要略小于最大的抖动 epoch number。

mode: 就'auto', 'min', ',max'三个可能。min_delta 和 patience 都和"避免模型停止在抖动过程中"有关系，所以调节的时候需要互相协调。通常情况下，min_delta 降低，那么 patience 可以适当减少；min_delta 增加，那么 patience 需要适当延长；反之亦然。

SVC: Support Vector Classification，一种基于 libsvm 的支持向量机

（1）完整参数形式

```
svm.SVC(C=1.0, kernel='rbf', degree=3, gamma='auto', coef0=0.0, shrinking=True,
        probability=False, tol=0.001, cache_size=200, class_weight=None,
        verbose=False, max_iter=-1, decision_function_shape='ovr',
        random_state=None)
```

（2）相关参数默认值及介绍

C	float 参数 默认值为 1.0	表示错误项的惩罚系数 C 越大，即对分错样本的惩罚程度越大，因此在训练样本中准确率越高，但是泛化能力降低；相反，减小 C 的话，容许训练样本中有一些误分类错误样本，泛化能力强。对于训练样本带有噪声的情况，一般采用后者，把训练样本集中错误分类的样本作为噪声。
kernel	str 参数 默认为'rbf'	该参数用于选择模型所使用的核函数，算法中常用的核函数有： -- linear：线性核函数 -- poly：多项式核函数 --rbf：径向核函数/高斯核 --sigmod：sigmod 核函数 --precomputed：核矩阵，该矩阵表示自己事先计算好的，输入后算法内部将使用你提供的矩阵进行计算
degree	int 型参数 默认为 3	该参数只对'kernel=poly'(多项式核函数)有用，是指多项式核函数的阶数 n，如果给的核函数参数是其他核函数，则会自动忽略该参数。
gamma	float 参数 默认为 auto	该参数为核函数系数，只对'rbf','poly','sigmod'有效。如果 gamma 设置为 auto，代表其值为样本特征数的倒数，即 1/n_features，也有其他值可设定。
coef0	float 参数 默认为 0.0	该参数表示核函数中的独立项，只有对'poly'和'sigmod'核函数有用，是指其中的参数 c。
probability	bool 参数 默认为 False	该参数表示是否启用概率估计，这必须在调用 fit()之前启用，并且会使 fit()方法速度变慢
shrinkintol	bool 参数 默认为 True	该参数表示是否选用启发式收缩方式
tol	float 参数 默认为 1e^-3	svm 停止训练的误差精度，也即阈值。
cache_size	float 参数 默认为 200	该参数表示指定训练所需要的内存，以 MB 为单位，默认为 200MB
class_weight	字典类型或'balance'字符串 默认为 None	该参数表示给每个类别分别设置不同的惩罚参数 C，如果没有给，则会给所有类别都给 C=1，即前面参数指出的参数 C。如果给定参数'balance'，则使用 y 的值自动调整与输入数据中的类频率成反比的权重

verbose	bool 参数 默认为 False	该参数表示是否启用详细输出。此设置利用 libsvm 中的每个进程运行时设置，如果启用，可能无法在多线程上下文中正常工作。一般情况都设为 False
max_iter	int 参数 默认为-1	该参数表示最大迭代次数，如果设置为-1 则表示不受限制
random_state	Int//None /Random State instance 默认为 None	该参数表示在混洗数据时所使用的伪随机数发生器的种子，如果选 int，则为随机数生成器种子；如果选 RandomState instance，则为随机数生成器；如果选 None，则随机数生成器使用的是 np.random

朴素贝叶斯: scikit-learn 中的三种算法

scikit-learn 中，一共有 3 个朴素贝叶斯的分类算法类:

（1）GaussianNB：先验为高斯分布，用于样本特征的分布大部分是连续值

（2）MultinomialNB: 先验为多项式分布，主要用于离散特征分类，例如文本分类单词统计，主题分类等，以出现的次数作为特征值

（3）BernoulliNB: 先验为伯努利分布，主要用于离散特征分类，如情感分析。MultinomialNB 以出现的次数为特征值，若样本特征是二元离散值或者很稀疏的多元离散值，应该使用 BernoulliNB

（1）GaussianNB Naive_Bayes

GaussianNB 类的主要参数仅有一个，即先验概率 priors，对应 Y 的各个类别的先验概率 P(Y=Ck)。这个值默认不给出，如果不给出此时 P(Y=Ck)=mk/m。其中 m 为训练集样本总数量，mk 为输出为第 k 类别的训练集样本数。如果给出的话就以 priors 为准。

在使用 GaussianNB 的 fit 方法拟合数据后，我们可以进行预测。此时预测有三种方法，包括 predict，predict_log_proba 和 predict_proba。

predict 方法就是我们最常用的预测方法，直接给出测试集的预测类别输出。

predict_proba 则不同，它会给出测试集样本在各个类别上预测的概率，即预测出的各个类别概率里的最大值对应的类别，也就是 predict 方法得到类别。

predict_log_proba 和 predict_proba 类似，会给出测试集样本在各个类别上预测的概率的一个对数转化。转化后 predict_log_proba 预测出的各个类别对数概率里的最大值对应的类别。

（2）MultinomialNB Naive_Bayes

参数 alpha，如果没有特别需要，用默认的 1 即可，如果发现拟合的不好，需要调优时，可以选择稍大于 1 或者稍小于 1 的数。

参数 fit_prior 表示是否要考虑先验概率，如果 false，则所有样本类别输出都有相同类别先验概率，否则可以自己用第三个参数 class_prior 输入先验概率，或不输入，默认为 True

参数 class_prior 让 MultinomialNB 自己从训练集样本来计算先验概率，此时的先验概率为 P(Y=Ck)=mk/m。其中 m 为训练集样本总数量，mk 为输出为第 k 类别的训练集样本数。默认为 None

（3）BernoulliNB Naive_Bayes

BernoulliNB 共 4 个参数，其中 3 个参数的名字和意义和 MultinomialNB 完全相同。

唯一增加参数是 binarize，用于处理二项分布，可以是数值或者不输入。如果不输入，则 BernoulliNB 认为每个数据特征都已经是二元的。否则的话，小于 binarize 的会归为一类，大于 binarize 的会归为另外一类。

References：

一、基于情感词典方法的情感分析

《中文短文本未登录词发现及情感分析方法研究》

http://cdmd.cnki.com.cn/Article/CDMD-10005-1018704873.htm

《一种基于微博类短文本的未登录词识别和词义发现研究》

http://cdmd.cnki.com.cn/Article/CDMD-10005-1019700962.htm

《中文情感词典的自动构建及应用》

http://cdmd.cnki.com.cn/Article/CDMD-10359-1015714099.htm

《文本情感分类（一）：传统模型》

https://kexue.fm/archives/3360

二、基于机器学习方法的情感分析

《numpy 中 reshape 函数的三种常见相关用法》

https://blog.csdn.net/qq_29831163/article/details/90112000

2.1 支持向量机

《scikit-learn 代码实现 SVM 分类与 SVR 回归以及调参》

https://blog.csdn.net/qq_41076797/article/details/101037721

《sklearn-核函数使用对比》

https://blog.csdn.net/houhuipeng/article/details/94319398

《sklearn 支持向量机 SVM》

https://www.jianshu.com/p/a9f9954355b3?utm_campaign=maleskine&utm_content=note&utm_medium=seo_notes&utm_source=recommendation

2.2 朴素贝叶斯

《scikit-learn 官方文档：1.9. Naive Bayes》

https://scikit-learn.org/stable/modules/naive_bayes.html

《朴素贝叶斯的三个常用模型：高斯、多项式、伯努利》

https://blog.csdn.net/abcd1f2/article/details/51249702

2.3 Xgboost

xgboost 作者讲义

https://homes.cs.washington.edu/~tqchen/pdf/BoostedTree.pdf.

《xgboost 入门与实战》

https://blog.csdn.net/sb19931201/article/details/52557382

《史上最详细的XGBoost 实战》

https://zhuanlan.zhihu.com/p/31182879

《scikit-learn 梯度提升树(GBDT)调参小结》

https://www.cnblogs.com/pinard/p/6143927.html

三、基于深度学习方法的情感分析

《深度学习之RNN 循环神经网络》

https://blog.csdn.net/huacha__/article/details/80652384

《Keras 过拟合相关解决办法》

https://www.deeplearn.me/2351.html

《正则化方法：L1 和L2 regularization、数据集扩增、dropout》

https://www.jianshu.com/p/ffb6808d54cd

《keras 的EarlyStopping 使用与技巧》

https://blog.csdn.net/zwqjoy/article/details/86677030