1. 自己提出的问题的理解
2. 提出的问题1：SVM的核方法把输入样本映射到高维的特征空间，在高维空间中还能保证算法的有效性吗？怎么评估这个误差是不是合理？

讨论后的理解：

由线性不可分变成线性可分是特征空间变化做的事情，核方法使特征变换的过程更加容易，这样SVM就可以将原始特征空间映射到更高维。从原理上讲，把高位向量映射到无穷维的Hilbert空间以后基本上就线性可分了，但是并不能保证他的泛化能力，即可能找到一条超平面将原始数据在新的特征空间线性可分，但是该分类面可能受到极少部分点的影响特别大，以至于在训练样本之外不能得到很好的预测效果。如果假定数据本身无噪声，那么理论上是会存在一个核函数，使得映射后的数据线性可分。关于如何找到最合适的核函数，目前还没有很好的方法。

1. 别人提出的问题的理解
2. 问题2：p83式41下面说使用超平面上的任一点与边缘超平面的距离来作为d+，而不是x+到超平面的距离，为什么会这样计算？两种方式有什么本质上的区别？如果没有的话书上为什么会强调这一点？

自己的理解：

因为边缘超平面上可能不存在点，如果没有点的情况则无法计算该点到分割超平面的距离，则无法得到距离d.

1. 问题3：软间隔svm中的损失函数为什么要选择hinge loss 这种形式，hinge loss有什么好处和局限性，有没有其他可比较的loss函数

自己的理解：

因为hinge loss损失函数保证了SVM解的稀疏性。因为hinge loss的零区域对应的是非支持向量的普通样本，从而所有的普通样本都不参与最终超平面的决定，这是SVM最大的优势所在，对新联样本数目的依赖大大减少，而且提高了训练效率。

1. 问题4：P77拉普拉斯修正是否有可能改变原先的预测结果？

自己的理解：

拉普拉斯修正避免了因为训练集样本不充分而导致概率估计值为0的问题，在训练集比较大时，修正过程所引入的先验对原先预测结果的影响可忽略不计，使得估计值趋向于实际概率值。

1. （必填）读书计划
2. 本周完成的内容章节：如3.6-3.8
3. 下周计划：3.9-3.10
4. （选做）读书摘要及理解或伪代码的具体实现
5. 读书摘要及理解（选做）

贝叶斯分类器：

样本之间的相互关系十分复杂，为了简化模型从而更简单的估计出条件概率，朴素贝叶斯分类器做了“属性条件独立性假设”：对已知类别，假设所有属性相互独立，即假设每个属性独立的对分类结果发生影响，即：



有了条件概率的简化条件之后，可以改写贝叶斯准则，即可得到朴素贝叶斯的表达式：



令表示训练集中第c类样本组成的集合，先验概率的似然估计为：



对于条件概率，通常假设满足某种概率分布，例如在sklearn中有三种假设的分布：GuassianNB、MultinomialNB和BernoulliNB。总的来说，朴素贝叶斯分类的推导主要基于贝叶斯公式，决策方式为在“属性条件独立性假设”条件下的贝叶斯决策论，参数估计的方法为极大似然估计。

2、伪代码的具体实现(选做)

贝叶斯分类器用于情感分类：

def run\_main():

# 预处理数据，如果是第一次，需要处理数据

if flag:

print('处理清洗文本数据', end=' ')

# 将标签和文本数据保存成csv

txt\_to\_csv()

# 读取处理好的csv文件，构造数据集

text = pd.read\_csv(os.path.join(dataset\_path, output\_text\_filename),

encoding='utf-8')

# 处理文本数据

text['text'] = text['text'].apply(process\_text)

text = text[text['text'] != '']

# 保存处理好的文本数据

text.to\_csv(os.path.join(dataset\_path, output\_cln\_text\_filename),

index=None, encoding='utf-8')

print('处理数据完成')

# 分割训练集、测试集

print('加载处理好的文本数据')

clean\_text = pd.read\_csv(os.path.join(dataset\_path, output\_cln\_text\_filename),

encoding='utf-8')

train\_text, test\_text = split\_text(clean\_text)

# 训练集、测试集数据分布

print('训练集中各类的数据个数：', train\_text.groupby('label').size())

print('测试集中各类的数据个数：', test\_text.groupby('label').size())

# 提取特征

n\_common\_words = 200

# 将训练集中的单词拿出来统计词频

print('统计词频')

words\_in\_train = creat\_word\_list(train\_text)

fdisk = nltk.FreqDist(words\_in\_train)

common\_words\_frequences = fdisk.most\_common(n\_common\_words)

print('出现最多的{}个词是：'.format(n\_common\_words))

for word, count in common\_words\_frequences:

print('{}: {}次'.format(word, count))

print()

# 在训练集上提取特征

text\_collection = TextCollection(train\_text['text'].values.tolist())

print('训练样本提取特征', end=' ')

train\_X, train\_y = extract\_feature(train\_text, text\_collection, common\_words\_frequences)

print('完成')

print()

# 在测试集上提取特征

print('测试样本提取特征', end=' ')

test\_X, test\_y = extract\_feature(test\_text, text\_collection, common\_words\_frequences)

print('完成')

# 训练贝叶斯模型

gnb = GaussianNB()

gnb.fit(train\_X, train\_y)

print('训练模型完成')

# 预测

test\_pred = gnb.predict(test\_X)

print('测试模型完成')

# 输出准确率

print('准确率：', cal\_accuracy(test\_y, test\_pred))

if \_\_name\_\_ == '\_\_main\_\_':

run\_main()