# 计算医疗课程疾病预测机器学习实验报告

学号：3200102333，姓名：陈绍文

# 一、问题描述

在日常医学诊断中，我们需要医生根据病人的病情描述判断病人的疾病状况，但是当我们拥有了足够的样本数据，就可以尝试通过机器学习来提取病情描述的特征来进行疾病的归纳总结，并对给出的病情描述做出判断，从而为医生提诊断的有效辅助手段。

# 二、模型介绍

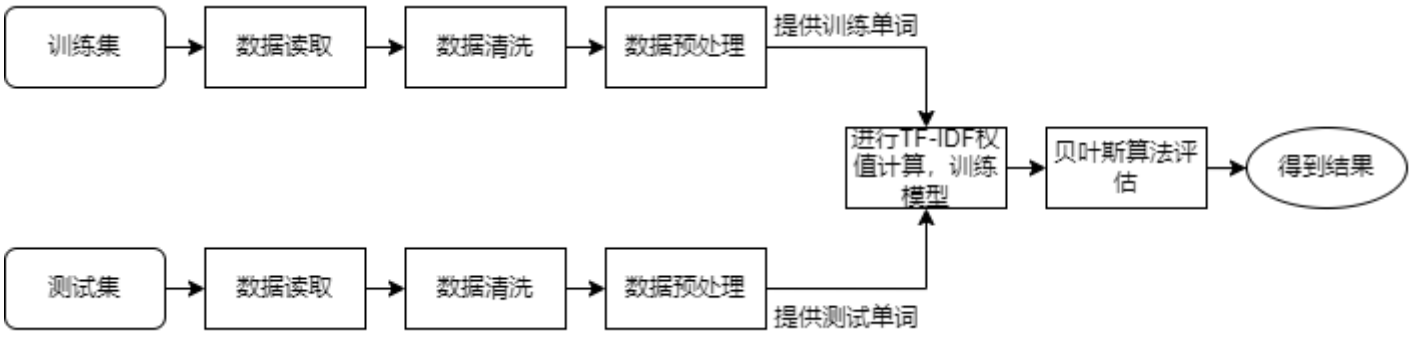
在本次机器学习实验中，主要包含以下内容：

## （一）数据来源

1. 训练集：指导老师提供约十万条数据，包含病情描述和对应的正确病症。
2. 测试集：指导老师提供的约二万六千条数据，包含病情描述和对应的正确病症。

## （二）模型准备

**1. 实验流程图**




**2. 贝叶斯分类算法核心 **

$$P(B|A) = \frac{P(A|B) * P(B)}{P(A)}$$

## （三）代码实现

1. 本次实验代码用python来实现。
2. 数据读写：本次实验数据存储在 *.csv 文件中，首先可以把文件转化为 *.xlsx 文件，接着在在 python 中调用 pandas 包对文件进行读取，再将数据转化为 bunch 结构进行存储。
3. 文本预处理：引用 paddle 包和 jieba 包进行中文文本切割，引入常用停用词表对分割结果进行筛选。
4. TF-IDF算法权值运算：引入 sklearn 包对处理后的文本使用TF-IDF算法进行单词权值的计算。
5. 贝叶斯预测：引入 sklearn.naive_bayes 使用多项式贝叶斯算法进行病情预测，读取预测结果。

# 三、实验过程

## （一）训练集数据处理

### 1. 数据集处理前准备工作

由于本次实验采用百度提供的深度学习框架 paddle ，所以需要启用该功能：

```
paddle.enable_static()
jieba.enable_paddle()
```

### 2. 训练集数据读取

读取分为两个两个部分，一个是停用词的引用，一个是 .xlsx 文件的读取：

```
def readFile(path):
    with open(path, 'r', errors='ignore') as file:  # 文档中编码有些问题，所有用errors过滤错误
        content = file.read()
        return content

stopwordlist = readFile("./stopword.txt").splitlines()

df = pd.read_excel("./train.xlsx",header = 0)
data_train = df.values
bunch = Bunch(target = [], label = [], contents = [])
```

为了方便后续的调用，这里将读取的数据再存入 bunch 结构中。

### 3. 单条数据处理

本次实验中采取 jieba 包对中文进行分割：

```
for i in data_train:
    seg_str = str(i[0])
    seg_str = re.sub('\W*', '',seg_str) #分词前剔除特殊符号、标点符号
    bunch.label.append(i[1])
    bunch.contents.append(" ".join(jieba.lcut_for_search(seg_str)))
bunch.target = list(set(bunch.label))
```

#### 4. TF-IDF算法生成权值

```
tfidfspace = Bunch(target = bunch.target, label = bunch.label, tdm = [], vocabulary = {})
vectorizer = TfidfVectorizer(stop_words=stopwordlist, sublinear_tf=True, max_df=0.5)
transformer = TfidfTransformer()
tfidfspace.tdm = vectorizer.fit_transform(bunch.contents)
tfidfspace.vocabulary = vectorizer.vocabulary_
bunch_train = tfidfspace
# 生成bunch_train用于后面的测试
```

## （二）测试集数据处理

#### 1. 测试数据读取和预处理

过程和上一部分类似。

```
df2 = pd.read_excel("./test.xlsx", header = 0)
data_test = df2.values
bunch_test = Bunch(target = bunch_train.target, label=[], contents=[])
for i in data_test:
    k = k + 1
    seg_str = str(i[0])
    seg_str = re.sub('\W*', '',seg_str) #分词前剔除特殊符号、标点符号
    bunch_test.label.append(i[1])
    bunch_test.contents.append(" ".join(jieba.lcut(seg_str, use_paddle=True)))
```

#### 2. TF-IDF算法生成权值

过程和上一部分类似。

```
testspace = Bunch(target = bunch_test.target, label = bunch_test.label, tdm = [], vocabulary={})
vectorizer = TfidfVectorizer(stop_words=stopwordlist, sublinear_tf=True, max_df=0.5,vocabulary=b
transformer = TfidfTransformer()
testspace.tdm = vectorizer.fit_transform(bunch_test.contents)
testspace.vocabulary = bunch_train.vocabulary
```

## （三）运用贝叶斯算法进行结果预测

将上面得到的训练集数据和测试集数据引入到这里，调用 `MultinomialNB()` 和 `predict_proba()` 进行结果的预测，读取可能性最高的三项病症，写入 `.xlsx` 文件方便后续处理。

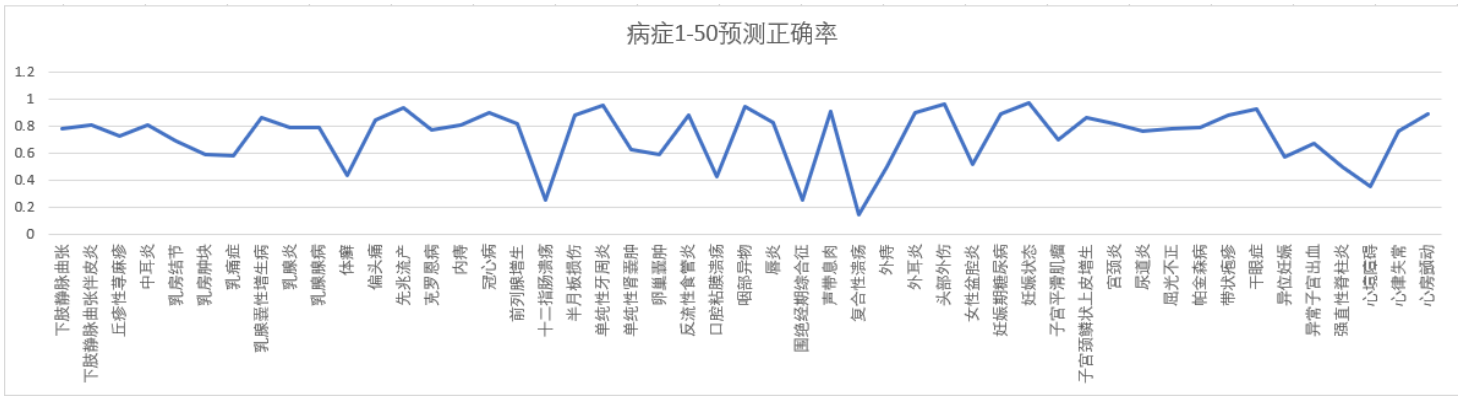
```
#贝叶斯
TrainSet = bunch_train
TestSet = testspace
clf = MultinomialNB(alpha=0.001).fit(TrainSet.tdm,TrainSet.label, None)

predicted = clf.predict_proba(TestSet.tdm)
total = len(predicted)
rate = 0
temp = sorted(TestSet.target)

# 数据写入excel
book = xlwt.Workbook(encoding='utf-8',style_compression=0)
sheet = book.add_sheet('result',cell_overwrite_ok=True)
col = ('实际类别','预测类别1','预测类别2','预测类别3','是否预测正确')
#写入title
for i in range(0,5):
    sheet.write(0,i,col[i])

i = 0
for flabel, expct_cate in zip(TestSet.label, predicted):
    #找到expct里最大的三个数
    result = []
    max_index1 = np.argsort(expct_cate)[-1]
    max_index2 = np.argsort(expct_cate)[-2]
    max_index3 = np.argsort(expct_cate)[-3]
    result.append(temp[max_index1])
    result.append(temp[max_index2])
    result.append(temp[max_index3])
    flag = False
    if flabel in result:
        rate += 1
        flag = True
    i = i + 1
    sheet.write(i,0,flabel)
    for I in range(0,3):
        sheet.write(i,I+1,result[I])
    sheet.write(i,4,flag)
    #print("实际类别：", flabel, "-->预测类别：", result)
book.save("./result.csv")
```

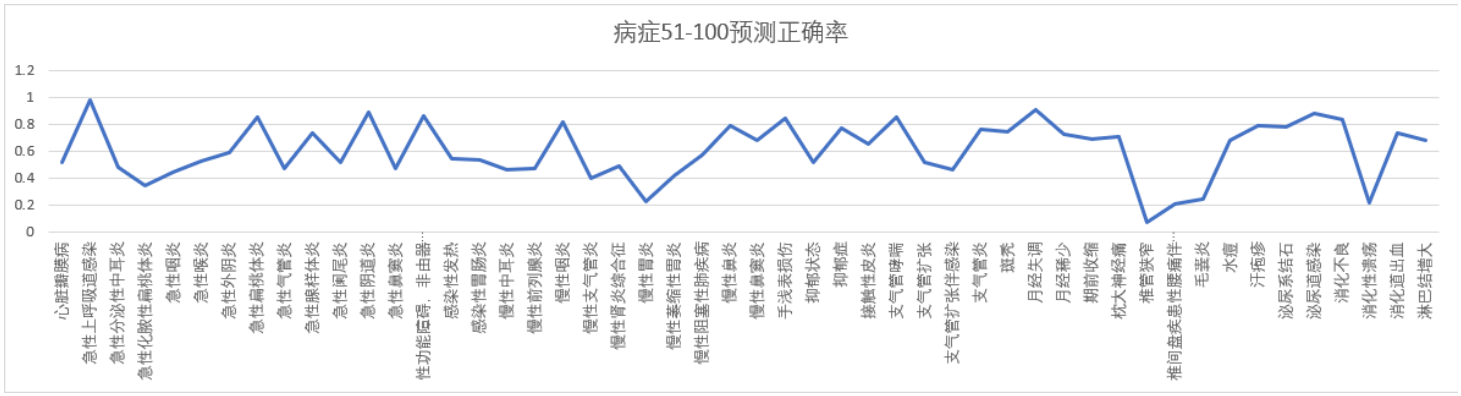
# 四、结果分析

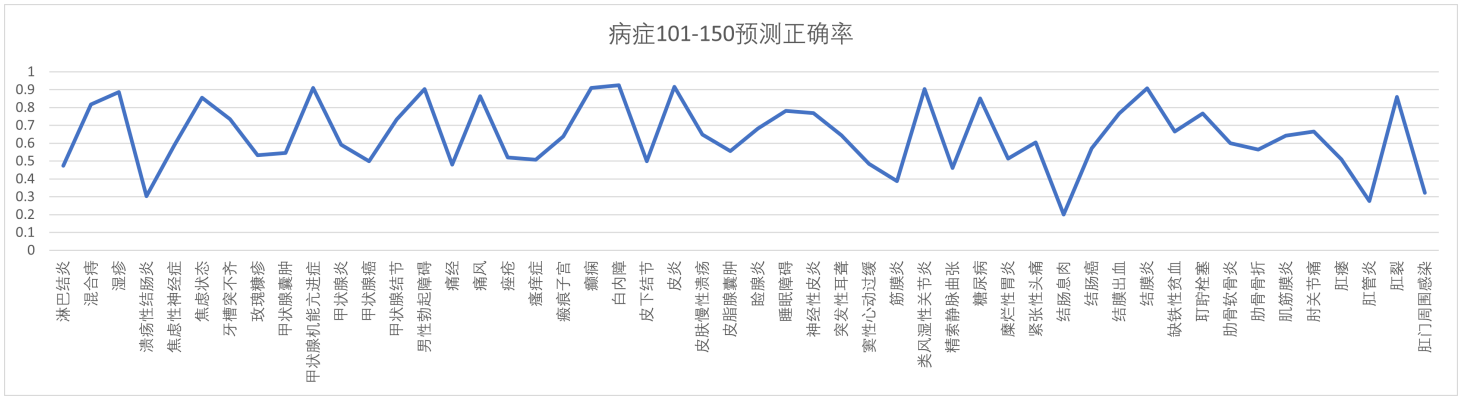
### 1. 整体预测结果

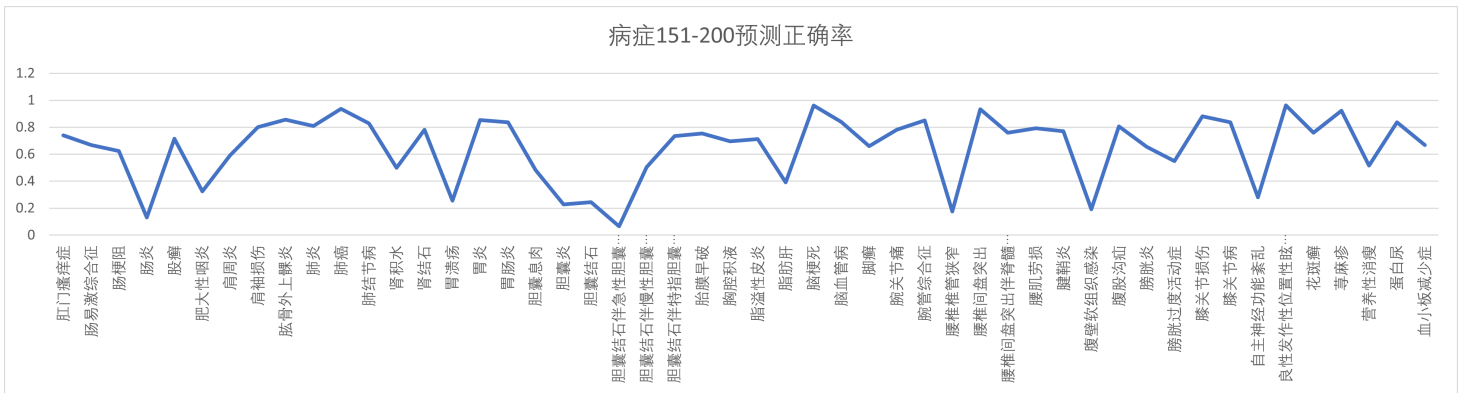
| | 一项预测正确率 | 三项预测正确率 | 五项预测正确率 |
|---|---|---|---|
| 正确率 | 0.5475303765564459 | 0.8020539442500847 | 0.8739419930030471 |

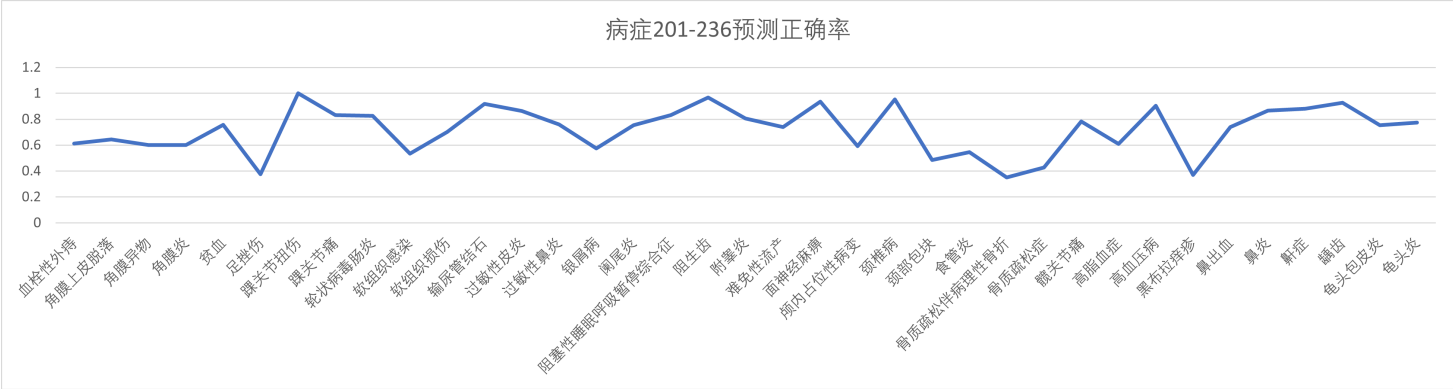
## 2. 分病症预测结果






从单个病症的预估情况来看，大部分病症的预测正确率在0.6以上，少部分病症的预测正确较低，拉低了整体的病症预测率。

**3. 结果分析**

观察实验结果，预估错误的原因主要有以下几点：

1. 该项病症本身数据较少；
2. 在病症类别中，存在过多相似病症；
3. 病情描述过于笼统，出现的关键词也大量存在于其他病症描述中。

# 五、讨论心得

本次实验让我浅显的了解如何运用python进行机器学习的一个流程，这次实验的尝试也给我留下了深刻的印象，同时实验中运用到的自然语言处理技术也让我认识到当下为解决各种问题而引伸出各种技术的魅力，新技术的产生总是伴随着新的需求而诞生。但是在实验中，我也有一些思考。

1. 首先是在进行自然语言处理的时候，有些副词和形容因为和被修饰的词语分开进行标记，个人认为是会对结果造成一定的影响的。
2. 而在对语言的处理中，我上网搜集了一些常用的停用词，但对内容进行浏览后我觉得虽然这些词汇放在日常使用的大环境下比较适用，但是在医学背景下，停用词的侧重会发生一些变化，个人觉得优化医学停用词表也将会优化我们的实验结果。
3. 在浏览结果的过程中，我发现针对具有较多样例的病症的预估正确率明显要高很多，但是对小数目的，或者是具有相似描述的病症则很难区分，我想这也是文本分割还不够准确的带来的影响，以及没有足够的数据来支撑的缘故。
   这次的实验让我感受到了机器学习的强大，不过看了这么多遍这些医学信息，也不禁感慨医学研究在数据收集、整理、分析方面是任重道远，这项事业是需要所有人参与其中的，医学的成绩是需要大家奉献出来的。