華南師範大學
South China Normal University

本 科 学 生 研 究 报 告

院　　系：计算机学院

课　　程：数据仓库与数据挖掘

老　　师：汤娜

开课时间：2018 ~ 2019 年度第 2 学期

专　　业：计算机科学与技术

研究成员：陈钦海 20162180046

数据挖掘期末大作业

一、研究内容

根据数据进行分类模型的构建

要求： 1. 用 python 实现学习算法

2. 至少实现 2-3 种不同类型的学习算法（贝叶斯、神经网络、决策树等）

3. 要求比较和分析通过不同学习算法建立的模型的准确率

4. 数据自行查找合适的数据源，但不得少于 1000 条

二、研究环境

系统环境： Windows 10 学生版

语言环境： Python 3.7

IDE： PyCharm

三、研究过程

（一）数据选择

数据选取： Kaggle 的 San Francisco Crime Classification

数据链接： https://www.kaggle.com/c/sf-crime/data

（二）数据预处理

1. 查看数据

```
>>> data_set.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 878049 entries, 0 to 878048
Data columns (total 9 columns):
Dates         878049 non-null datetime64[ns]
Category      878049 non-null object
Descript      878049 non-null object
DayOfWeek     878049 non-null object
PdDistrict    878049 non-null object
Resolution    878049 non-null object
Address       878049 non-null object
X             878049 non-null float64
Y             878049 non-null float64
dtypes: datetime64[ns](1), float64(2), object(6)
memory usage: 60.3+ MB
```

总共有 87w+的数据，9 列属性，Category 是我们的标签属性。

‘Dates’：犯罪日期时间

‘Category’：犯罪类型

‘Descript’：犯罪描述

‘DayOfWeek’：犯罪时间星期几

‘PdDistrict’：犯罪地区

‘Resolution’：犯罪后的处理

‘Address’：犯罪地址

‘X’ ‘Y’ ：犯罪地址坐标

2. 数据分析

随机抽取 1200 条数据作为训练集，在训练集里抽取 200 条作为测试集。

```
>>> data_set.Category.unique()
array(['OTHER OFFENSES', 'NON-CRIMINAL', 'LARCENY/THEFT',
       'FORGERY/COUNTERFEITING', 'VANDALISM', 'MISSING PERSON',
       'KIDNAPPING', 'SECONDARY CODES', 'WARRANTS', 'ASSAULT',
       'DRUG/NARCOTIC', 'VEHICLE THEFT', 'FRAUD', 'DRUNKENNESS',
       'ROBBERY', 'SUSPICIOUS OCC', 'STOLEN PROPERTY', 'BURGLARY',
       'TRESPASS', 'SEX OFFENSES FORCIBLE', 'WEAPON LAWS',
       'DRIVING UNDER THE INFLUENCE', 'EXTORTION'], dtype=object)
>>> data_set.Category.unique().size
23
```

标签集有 23 种

根据观察，最终决定取‘Dates’，‘PdDistrict’，‘DayOfWeek’作为条件属性

Dates 的格式是 2014-11-27 22:14:00，但是我们取 hour 作为研究属性

可知分布也在 0-23 点之间

```
>>> data_set.Dates.dt.hour.max(),data_set.Dates.dt.hour.min()
(23, 0)
```

PdDistrict 不同地区也达到了 10 个

```
>>> data_set.PdDistrict.unique()
array(['SOUTHERN', 'PARK', 'CENTRAL', 'RICHMOND', 'TARAVAL', 'NORTHERN',
       'TENDERLOIN', 'INGLESIDE', 'MISSION', 'BAYVIEW'], dtype=object)
>>> data_set.PdDistrict.unique().size
10
```

DayOfWeek 星期也达到了 7 个

```
Name: DayOfWeek, Length: 200, dtype: object
>>> data_set.DayOfWeek.unique().size
7
```

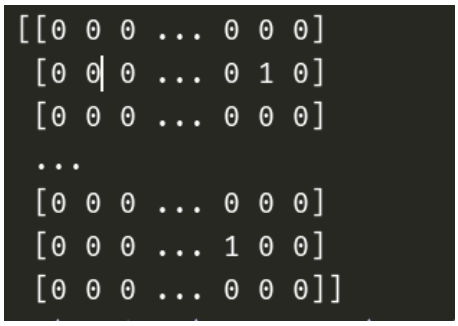
说明这些属性作为条件数据可靠。

3. 特征处理

考虑到作为属性都不是数字，为了方便处理，把条件根据属性值的不同转化为数字。

```
>>> data_set
      Dates  DayOfWeek  PdDistrict  Category
1088     20          2           8         9
1116     15          4           3         9
528      11          1           8        12
1006     16          6           1         9
```

对于神经网络分类算法，需要对标签进行二进制矩阵化



（三）算法实现

1. 朴素贝叶斯实现

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

根据朴素贝叶斯公式，

要求 P（Category=? | Dates，DayOfWeek，PdDistrict）

则可以求 P(Dates，DayOfWeek，PdDistrict | Category=?) * P(Category = ?) / P(Dates，DayOfWeek，PdDistrict)

由于我们要通过此算法模拟分类，所以我们进行比较只需要计算分子部分即可。

对此，实现算法我们只需要通过已知的的训练集，计算出所有 P(Dates，DayOfWeek，PdDistrict | Category=?)与 P(Category = ?)，在测试时，计算每一类的最终概率值，就能比较得出哪一类概率最大。

```
Class Bayes:
     calc_p_y(self,column_y_val):
     calc_p_x_y(self,column_x_val,column_y_val):
     set_test_x(self,Dates,DayOfWeek,PdDistrict):
     predict_one(self,column_x_val):
     predict_all(self,test_set):
```

方法实现：

1. calc_p_y

传入 Category 类别值，通过遍历训练集统计 count 值，最终计算返回占比率

2. calc_p_x_y

传入数据集条件与类别值，遍历数据集该类别下各条件的比例，并进行相乘，最终计算返回占比率

3. set_test_x

传入预测条件，返回列表形式

4. predict_one

传入列表形式的条件，计算每一类的最终概率，最终得出概率最大的类别。

5. predict_all

传入测试集，循环进行 predict_one 方法，最终返回正确率。

2. 决策树分类实现

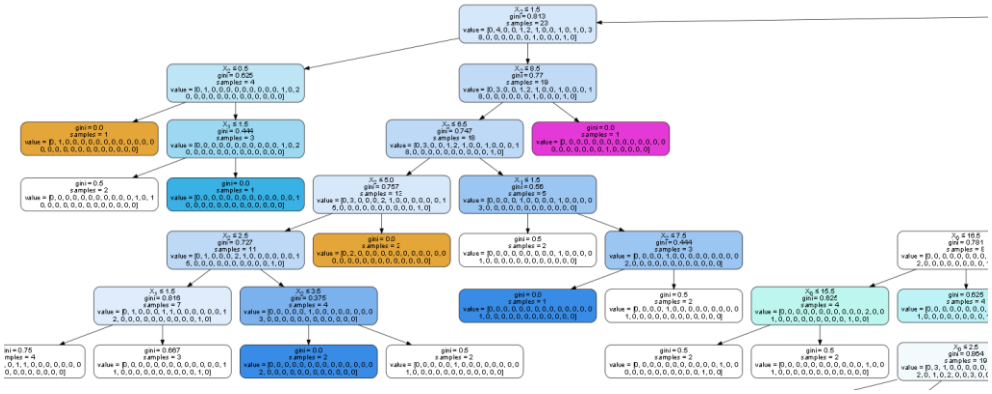
根据决策树分类原理的信息增益熵来按次序分类，通过 Gain(S,A)作为信息增益熵。

$$info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} info(D_j)$$

$$gain(A) = info(D) - info_A(D)$$

决策树分类是通过 sklearn 库的方法调用实现。

模拟完成后，输出准确率，生成决策树文件，需要通过 GvEdit 软件打开。



3. 神经网络实现

```
Class NeuralNetwork:
    __init__(self,neuron_list):
    fit(self, X, y, learning_rate=0.2, epochs=10000):
    predict_one(self, x):
    predict_all(self,test_x,test_y):
```

方法实现：

1. __init__

初始化模型，neuron_list 作为神经元列表传入模型，将权值全部随机赋值。

2. fit

训练函数，将学习速率，更新最大次数，训练集和标签集作为输入，随机选取训练集某一行，对网络进行更新，进行正向更新，之后计算权重后，通过随机梯度函数进行反向更新权值。

3. predict_one

根据传入训练集的一项，预测标签类别。

4. predict_all

一次性预测整个训练集，并输出混淆矩阵和分类情况。

（四）结果对比

	朴素贝叶斯	决策树	神经网络
准确率	0.3	0.07	0.08-0.20

（五）结果分析

1. 朴素贝叶斯由于数据较多，表现较好；
2. 而决策树条件太多 24*7*10=1680 个分类节点，使得决策树过于肿胀，导致准确率很低。
3. 神经网络稳定性不高，很可能是它是局部最优搜索方法，很可能会陷入局部极值，使训练失败，条件太多，可能会出现“过拟合”的现象。

（六）结论

1. 各分类算法各有各的优点，对于贝叶斯实现简单，数据越多，可能表现越好，越准确，但是仅仅在各条件之间相互独立，这个情况在实际应用中往往不成立，随着属性的增多，分类效果会下降。
2. 对于决策树分类算法，前期的数据预处理很重要，如果标签类太多，条件太多，决策树可能会变得很庞大，加入旧的数据效果可能不错，但当加入一个全新的数据，决策树分类效果可能会下降。
3. 对于神经网络分类算法，其能够自适应学习，是现在比较火的学习方法，但是在实际应用中学习成本较高，并且容易被无关数据干扰。

四、研究总结

这次作业主要研究了三种分类算法，其中朴素贝叶斯分类算法是最容易实现的算法，而神经网络和决策树分类算法花了较长时间实现，可能是对算法理解还不是吃的很透。