

哈尔滨工业大学

软件安全课程实验报告

实验五

## 学院：计算学部

## 班级：1703201

## 成员：1170300422-张昊

**1170300421-贺宗磊**

# 恶意行为异常检测方法

### 一、实验项目描述

###### 1、 理解基于异常检测的恶意攻击行为检测方法

1. 掌握异常检测的流程
2. 学习相关的异常检测算法

###### 2、 基于距离的异常检测方法

1. 掌握欧氏距离的概念
2. 如何选取数据集的属性集
3. 选取合适的检测模型

###### 3、 基于KD树的网络流量异常检测模型

1. 利用KD树构建一个用于多维空间最邻近搜索的数据结构
2. 建立历史数据集合
3. 规格化数据
4. 采用标准分割策略，进行基于维度分割的KD树构建
5. 基于待检测数据X，利用构建好的KD树搜索，找出历史数据集合中与待检测数据X最近的数据；计算二者之间的欧氏距离，与阈值比对，确定是否是异常数据点。

### 二、实验要求

1. 实验数据准备。利用KDD1999数据集(KD树异常检测.pdf p31页-34页)提供的数据进行实验。选取部分正常数据做为训练集，选择部分攻击数据和剩下的正常数据做为测试集。
2. 可以只选择流量属性集(KD树异常检测.pdf p34页)。可只针对DOS(smurf攻击即可)攻击进行异常检测，其他攻击不考虑。
3. 2人一组完成实验。
4. 下载阅读CMS“相关资料”中的KD树异常检测.pdf文件
5. 利用CMS“相关资料”中的kddcup.data\_10\_percent.gz作为数据（看数据说明）。数据中每条都有标记为：NORMAL 或 ATTACK 类型。利用标记为 NORMAL 的数据建模（构建KD树）。利用一部分标记为 NORMAL 的数据 和ATTACK数据作为测试数据。 （数据集中可能存在少量有错误的数据，注意）

### 三、实验结果

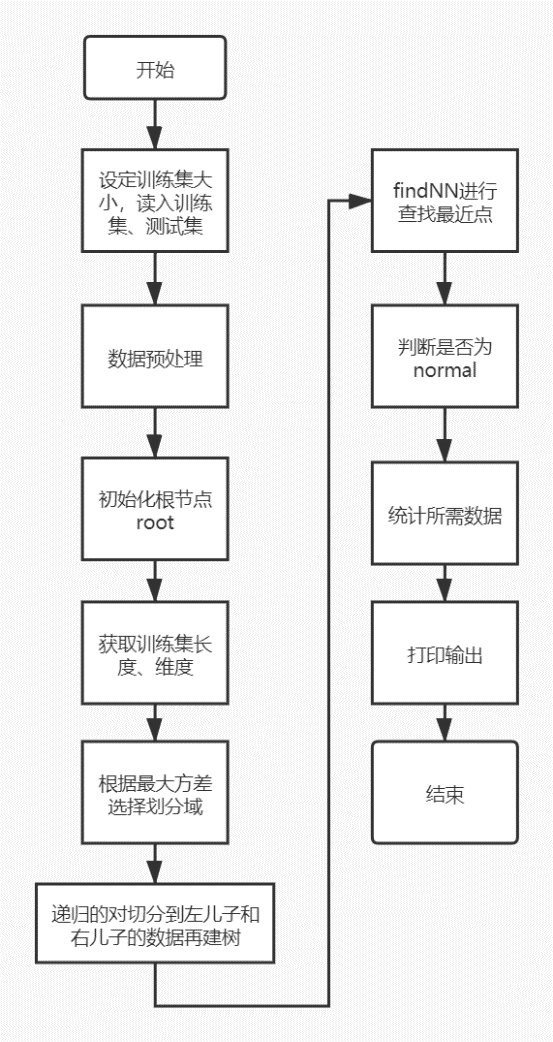
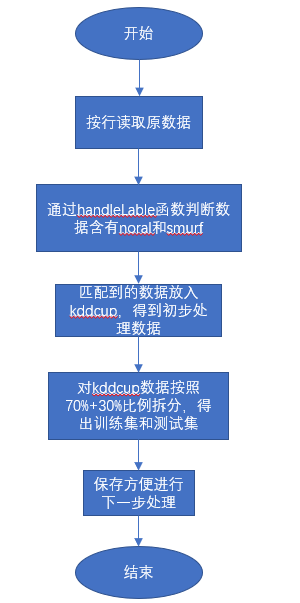
###### 实验分工

贺宗磊：负责数据预处理程序，筛选数据，拆分训练集和测试集的以及帮助算法设计

张昊：负责主要算法设计，实现训练和测试以及结果统计

###### 程序流程图

* KD树及测试程序流程 数据处理流程图

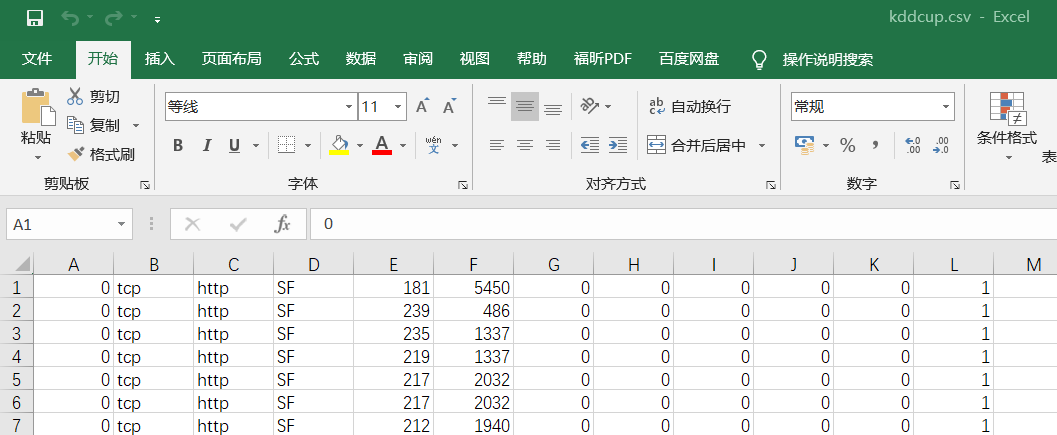
 

###### 数据处理过程和结果

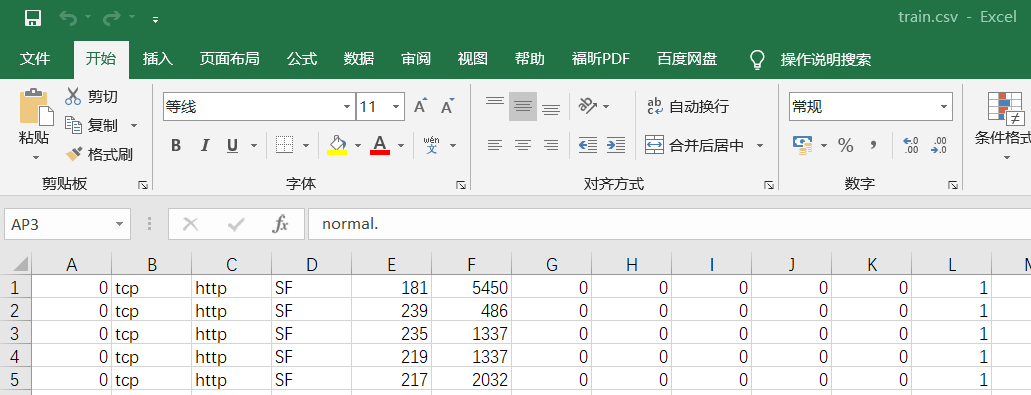
由于KDD1999数据集中包含数据种类较多，而本次实验只针对标记为normal和smurf的数据进行处理匹配，因此，对原文件为kddcup.data\_10\_percent的数据集首先进行读取匹配，匹配标记为normal和smurf的数据，并将其保存在kddcup.csv文件内以进行方便进一步处理和查看验证，再对kddcup.csv文件内的数据按照70%训练集和30%测试集的比例进行拆分，得到train.csv和test.csv，方便程序进行进一步KD树构建和相关异常检测。

数据集结果：

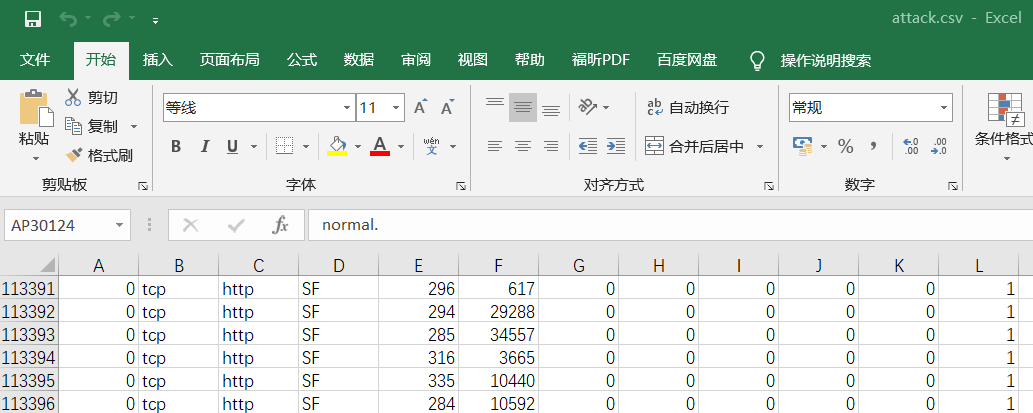
Kddcup初步处理数据集：



训练集train.csv:



测试集test.csv:



训练集中正常数据和smurf数据个数分别为：71225个和193422个，共264647个。

测试集中正常数据和smurf数据个数分别为：26052个和86968个，共113020个。

###### 列出构建KD树所定义的数据结构已经程序主要方法，简单说明其功能

1. KD树结构：

包括point（数据点的特征向量）、split（切分的维度）、left（左儿子）、right（右儿子）

1. computeVariance：

计算方差，以利用方差大小进行判断在哪一维进行切分

1. createKDTree：

建树，首先获取数据列表的长度len、数据的维度dimension、对每一维的方差var进行计算并且找到最大方差max\_var，根据划分域的数据对数据点进行排序，选择下标为len / 2的点作为分割点，递归的对切分到左儿子和右儿子的数据再建树。

1. computeDist：

计算欧氏距离

1. findNN：

初始化为root的节点，用dist\_list来存储前三个点，while循环，二分查找建立路径，对向下走的路径进行压栈处理，计算当前最近节点和查询点的距离大小，当前节点的划分域，更新继续循环直到结束。回溯查找，当查询点和回溯点的距离小于当前最小距离时，另一个区域有希望存在更近的节点，更新dist\_list表并在最后返回

1. judge\_if\_normal：

判断是否为normal

1. pre\_data：

数据预处理

1. my\_test：

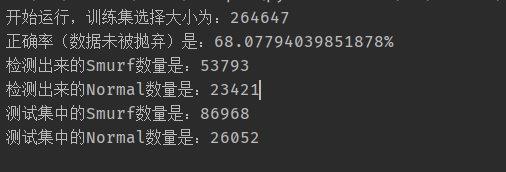
测试数据正确性

1. draw：

运行函数

###### 实验结果和实验数据一起给出。如选择多少攻击数据做为测试集？检测出多少攻击？漏报率是多少？

实验结果截图



如图：

总数据的正确（数据未被抛弃）率是：68.07794039851878%

检测出来的Smurf数量是：53793

检测出来的Normal数量是：23421

从数据处理结果得到测试集中的smurf数量为：86968

从数据处理结果得到测试集中的normal数量为：26052

计算得到漏报率是：38.15%