

**课程报告**

**题 目 Webshell中恶意函数检测**

学生姓名 王栋

学 号 201983290327

院 系 计算机与软件学院

专 业 信息安全专业

指导教师 闫雷鸣

**二Ｏ二 1 年 十二 月 十 日**

目 录

[1 引言 1](#_Toc118158772)

[1.1应用背景 1](#_Toc118158774)

[1.2工作流程 1](#_Toc118158774)

[2 数据集 1](#_Toc118158773)

[2.1黑样本 1](#_Toc118158774)

[2.1白样本 1](#_Toc118158774)

[3 特征提取 1](#_Toc118158774)

[4 研究方法和模型评估 26](#_Toc118158775)

5 性能比较 [30](#_Toc118158776)

[6 总结与展望 35](#_Toc118158777)

[参考文献 36](#_Toc118158778)

**关于目录格式的补充说明：**

1.目录两个字使用居中，黑体，三号， “目”与“录”字中间留九个空格。

2.目录页内容使用黑体，四号。

**Webshell中恶意函数检测**

王栋、杨云龙

南京信息工程大学计算机与软件学院 江苏 南京 21044

**摘要：**随着网络安全行业的发展，攻击技术呈现多元化发展的态势，导致原有的木马检测技术在当今并不适用，以至于Web应用系统易被攻击入侵,危害了国家安全和社会稳定。当今,随着机器学习、深度学习等技术的发展，可以应用到Webshell中的恶意函数检测中。通过机器学习，来对Webshell和普通文件中的函数进行特征学习,来完成对未知页面的预测.数据集包括了恶意后门3000个,来自互联网上常见的Webshell脚本,普通PHP文件3000个,来自普通开源CMS.对数据进行预处理,使用Python匹配Webshell中的危险函数,并用2-Gram进行分组,用TF-IDF进行进一步分组.使用朴素贝叶斯、随机森林等方法进行训练模型测试。

**关键词:**webshell,危险函数

**Malicious function detection in webshell**

Wang Dong, Yang Yunlong

School of computer and software, Nuist, Nanjing, Jiangsu 21044

**Abstruct:**With the development of the network security industry, the attack technology presents a diversified development trend, resulting in the original Trojan horse detection technology is not applicable today, so that the web application system is easy to be attacked and invaded, endangering national security and social stability. Nowadays, with the development of machine learning, deep learning and other technologies, it can be applied to malicious function detection in webshell. Through machine learning, we can learn the features of functions in webshell and ordinary files to complete the prediction of unknown pages The data set includes 3000 malicious backdoors, from common webshell scripts on the Internet, 3000 ordinary PHP files, and from ordinary open source cms Preprocess the data, match the dangerous functions in webshell with Python, group with 2-gram and further group with TF-IDF Naive Bayes, random forest and other methods are used to test the training model.

**Keywords**: webshell, dangerous function

1.引言

1.1 应用背景

webshell就是以 asp、aspx、php、jsp 或者cgi等网页文件形式存在的一种命令执行环境，也可以将其称做为一种网页后门。目前杀软将文件分为了三个等级：无害、可疑、确认病毒。

而杀软查杀木马的方式包含了静态查杀与动态查杀。其中静态查杀包括了危险函数、文件名和MD5、Shellcode特征、可疑加密。动态查杀多为启发式扫描,重点监控对象为服务、注册表、组策略、防火墙、敏感程序与IAT导出表。

其中系统木马代码中的危险函数指是:virtualloc,rtlmovememory,ntcreatthread等,大多数是和内存,堆,线程相关的Windowsapi函数。另外比如Python中的subprocess、Popen等涉及线程执行命令的函数。

其中文件名和MD5指的是杀软会分析文件的MD5值和已知病毒库中的MD5值作比较,继而进行分段的模糊匹配,继而判定.

通常来说,Webshell的利用方式一般为攻击者先上传小马。小马体积小,但是可以通过特定的Webshell管理工具,譬如中国蚁剑,哥斯拉等链接上去,再通过管理工具上传大马,大马体积大,功能包括文件管理、命令执行、杀软检测、系统提权等功能,一般会通过设置加密密码来达到隐蔽的目的。

1.2 工作流程

本项目通过对来自于Github上的300个Webshell以及两个开源CMS:Anchor CMS、Joomla CMS 中的文件进行分析,实现对PHP文件中涉及到Webshell行为操作中的危险函数的检测.

具体工作如下:

1. 基于PHP语言特征,对PHP文件进行探索性的分析,研究危险函数的特征,进行特征提取
2. 分别采用随机森林、决策树、对预处理后的数据构建预测模型，并通过对比率等方法评价模型.
3. 总结和展望，提出进一步改进方法，指出进一步研究方向.

2.数据集

数据集中900个样本是Webshell,2000个白样本是开源CMS的文件

2.1 黑样本

Webshell是黑客经常使用的一种恶意脚本，其目的是获得服务器的执行操作权限，常见的webshell编写语言为asp、jsp和php。比如执行系统命令、窃取用户数据、删除web页面、修改主页等

Webshell 数据来源于开源社区Github,全部采用PHP语言编写的Webshell样本,如图1所示

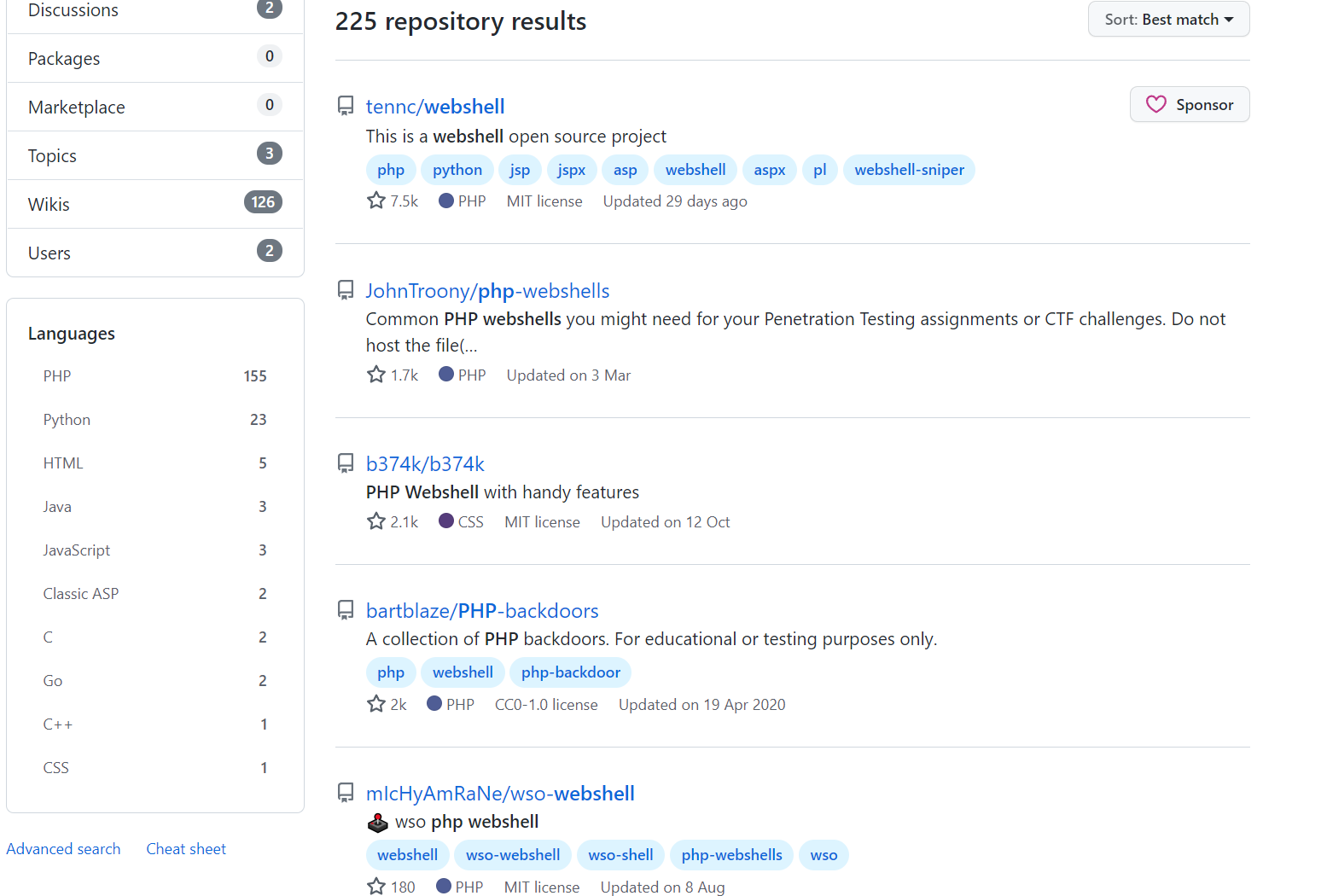


图1 开源社区PHP webshell项目

2.2 白样本

白样本来源于常见的PHP开源CMS,分别为如下几种:

2.2.1 Anchor CMS

Anchor CMS是一个免费的开源博客系统，具有轻量级功能，特别是设计用于关注写作和艺术指导博客。管理界面仅显示基本功能，如创建新帖子、创建页面、管理评论、类别和用户以及创建自定义字段。

详细文件目录结构可见图2:

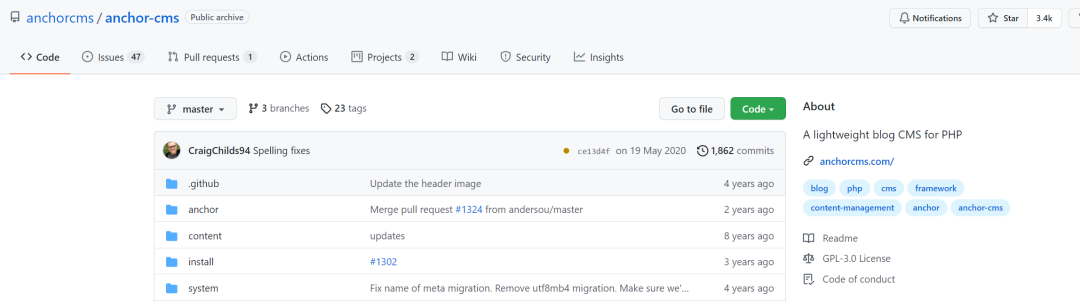


图2 开源CMS Anchor CMS

2.2.2 Joomla CMS

Joomla是一个用于发布web内容的免费开源内容管理系统（CMS）。多年来，它构建在模型-视图-控制器web应用程序框架上，可以独立于CMS使用，CMS允许构建在线应用程序。具有平台用户友好、可扩展、多语言、可访问、响应迅速、搜索引擎优化等优势。

详细文件目录结构可见图3:

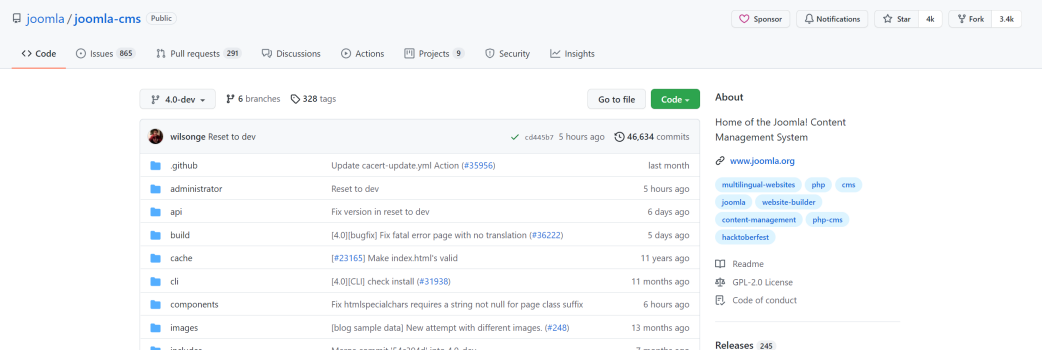


图3 开源CMS Joomla CMS

2.2.3 October CMS

October CMS是一个内容管理系统（CMS）和web平台，其唯一目的是使的开发工作流程再次变得简单。由于目前建立网站已经成为一个复杂和混乱的过程，降低了开发者的效率。所以October CMS具有更高的效率和更快的速度。

详细文件目录结构可见图4:

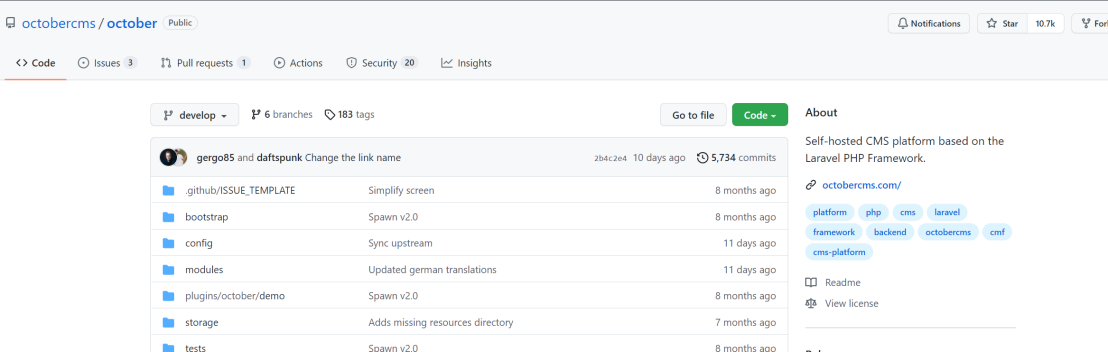


图4 开源CMS October CMS

3 特征提取

首先定义处理函数Slove\_File(),将PHP文件处理返回为简单字符串,过滤换行符,单引号,并将过滤后的PHP文本同意返回到列表Word\_list中.

* 1. jieba分词和关键词提取

Jieba分词是基于前缀词典实现高效的词图扫描,采用动态规划查找最大概率路径,找出基于词频的最切分组合.使用基于TF-IDF算法和基于TextRank算法的关键词提取。基本思想为:

1.将待抽取关键词的文本进行分词

2.以固定窗口大小(默认为5，通过span属性调整)，词之间的共现关系，构建图。

3.计算图中节点的PageRank，注意是无向带权图

我们加载处理后的Webshell文本内容,并统计样本个数,进行关键词提取和匹配,若干存在危险函数,则返回1，若不存在危险函数,则返回0

其中PHP webshell中常见恶意函数如下:

|  |  |  |  |
| --- | --- | --- | --- |
|  | 函数名 | 功能 | 使用方法 |
| 1 | eval() | 把字符串当成 PHP 代码来计算 | eval($\_POST[‘hack’] |
| 2 | Sytem() | 执行系统命令 | System($\_POST[‘hack’]) |
| 3 | assert() | assert() 会检查指定的 assertion 并在结果为 false 时采取适当的行动。 | assert($\_POST[‘hack’]) |
| 4 | passthru | passthru()函数也是用来执行外部命command）的.当所执行的 Unix 命令输出二进制数据， 并且需要直接传送到浏览器的时候,需要用此函数来替代 exec() 或 system() 函数 | Passthr($\_POST[‘hack’]) |
| 5 | call\_user\_func() | 把第一个参数作为回调函数（callback）调用，把参数数组作（param\_arr）为回调函数的的参数传入。 | create\_function('$a',"echo $a") |

表1 恶意函数表

通过对关键词语进行提取和匹配,统计恶意函数在文件中出现的次数,以助于我们来预测其是否为Webshell。

代码如图5所示

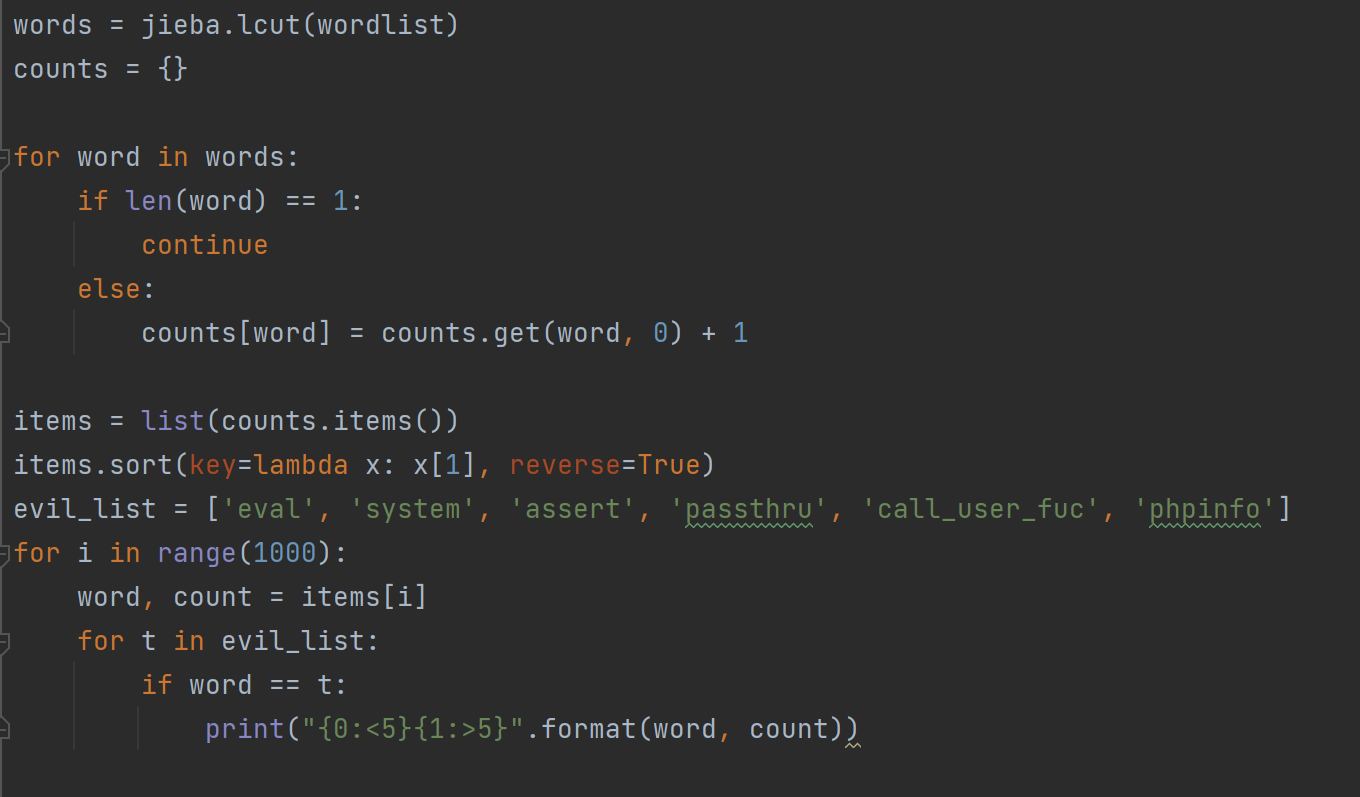


图5 匹配恶意函数出现次数算法

通过对关键词语进行提取,我们匹配了在所有的样本文件中存在的恶意函数的出现次数,如图6所示

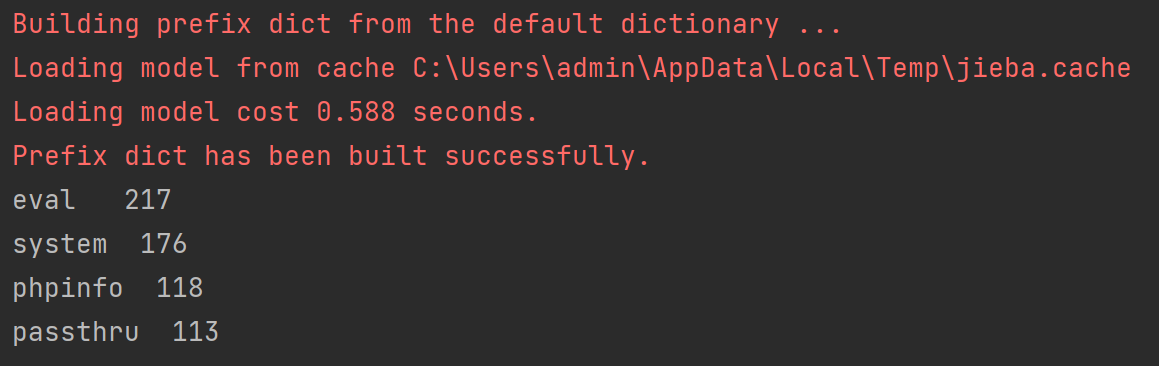


图6 匹配恶意函数出现次数

* 1. 研究恶意函数的Opcode特征码

3.2.1 Opcode简介:

Opcode是计算机系统中的字节码,它是指计算机指令中的一部分。用来解决处理器中的指令处理的问题。由于PHP运行在Zend虚拟机之上,则Zend虚拟机的操作指令即PHP的opcode。Zend虚拟机会根据操作码和操作数进行相应的指令操作,从而达到运行整个PHP文件的功能。

将PHP文件转化成opcode再进行特征提取,在研究危险函数的特征提取上具有决定性的作用。相较与传统的检测方法可能具有效率性。

3.2.2:PHP webshell危险函数特征提取

PHP类型的Webshell会通过绕过、混淆等技术改变本身的Webshell特征。因此我们需要通过将PHP代码转换为中间层次的OPCDOE码,来提取相应的训练特征。转化为Opcode码需要用到PHP的VLD插件,其中简要思路如图7所示:

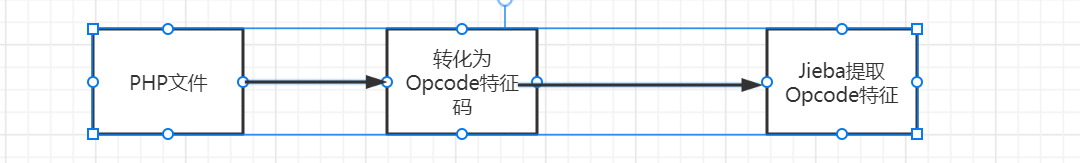


图7:PHP特征提取思路

3.2.4:PHP中恶意函数Opcode特征码

通过Opcode提取,我们提取了常见5个函数中的Opcode特征码,作为模型评估依据.

|  |  |
| --- | --- |
| BEGIN\_SILENCE | SEND\_VAR |
| FETCH\_R | SEND\_VAL ADD\_STRING |
| FETCH\_DIM\_R | ADD\_VAR SEND\_VAL |
| INCLUDE\_OR\_EVAL | DO\_FCALL RETURN |
| END\_SILENCE RETURN |  |

3.2.5:PHP文件Opcode提取比对

我们对Webshell函数和普通的PHP文件进行批量的opcode提取,继而将其提取结果按行保存在white.txt和black.txt中.经过长时间的运行,得到了两个文件,并且将上述恶意Opcode代码作为评判的依据.将存在恶意函数中Opcode的PHP文件标记为1记为恶意Webshell,不存在恶意函数的PHP文件标记为0,记为普通文件.

4 研究方法和模型评估、

4.1研究设计

由于Webshell的检测问题可以归纳成为一个二分类的问题,因此本文将分类设计为精确率、召回率和F值来作为模型评估的依据,如表2

|  |  |  |
| --- | --- | --- |
| 实际 | 预测 | |
| Webshell | 正常文件 |
| Webshell | True Positvie(TP) | False Negative(FN) |
| 正常文件 | False Positive(FP) | True Negative(TN) |

表2 模型评估依据

其中评判值计算公式为:

4.1.1 精确率

4.1.2 召回率

4.1.3 F值

4.2 KNN

KNN（K- Nearest Neighbor）法即K最邻近法最初由 Cover和Hart于1968年提出，是一个理论上比较成熟的方法，也是最简单的[机器学习算法](https://baike.baidu.com/item/%E6%9C%BA%E5%99%A8%E5%AD%A6%E4%B9%A0%E7%AE%97%E6%B3%95/18635836" \t "https://baike.baidu.com/item/%E9%82%BB%E8%BF%91%E7%AE%97%E6%B3%95/_blank)之一。该方法的思路非常简单直观：如果一个样本在特征空间中的K个最相似（即特征空间中最邻近）的样本中的大多数属于某一个类别，则该样本也属于这个类别。该方法在定类决策上只依据最邻近的一个或者几个样本的类别来决定待分样本所属的类别.KNN算法的核心思想是，如果一个样本在特征空间中的K个最相邻的样本中的大多数属于某一个类别，则该样本也属于这个类别，并具有这个类别上样本的特性。该方法在确定分类决策上只依据最邻近的一个或者几个样本的类别来决定待分样本所属的类别

算法处理流程如下:

1.使用VLD提取Webshell与开源CMS中的Opcode

2.随机划分训练集与测试集

3.在训练集上使用KNN进行训练,获取模型数据

4.使用模型数据在测试集上进行预测

5.验证KNN预测结果

在训练集上训练数据,针对测试集进行检测,图9显示了特征提取Opcode特征后,在最大特征数取100的情况下精确率为97.19%，召回率为98.21%，F值为97.70%，如图8所示

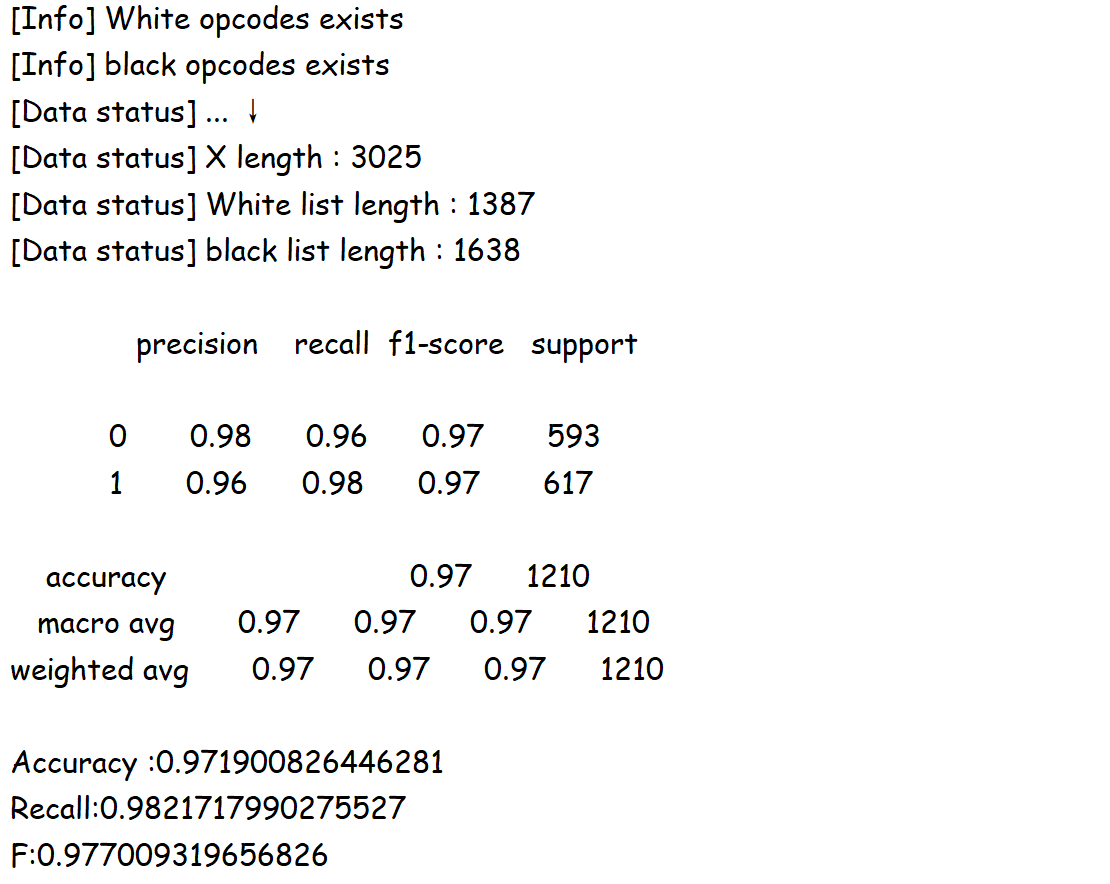


图8 KNN

|  |  |  |  |
| --- | --- | --- | --- |
| 训练类型 | 精确率 | 召回率 | F值 |
| KNN(第一次) | 0.97 | 0.97 | 0.97 |
| KNN(第二次) | 0.97 | 0.97 | 0.97 |
| 平均值 | 0.97 | 0.97 | 0.97 |

表3 KNN实现结果

4.3 决策树算法

决策树算法是机器学习算法中的一种,通过样本中某一维度属性的数值，将样本划分到不同的类别中。随机森林是指利用多棵树对样本进行训练并预测的一种分类器,通过投票进行选择具有优化性质的分类器。如果通过决策树来判断投票结果。具有解决过拟合现象的优点。

决策树算法构造决策树来发现数据中蕴涵的分类规则．如何构造精度高、规模小的决策树是决策树算法的核心内容。决策树构造可以分两步进行。第一步，决策树的生成：由训练样本集生成决策树的过程。一般情况下，训练样本[数据集](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E9%9B%86/4745883" \t "https://baike.baidu.com/item/%E5%86%B3%E7%AD%96%E6%A0%91%E7%AE%97%E6%B3%95/_blank)是根据实际需要有历史的、有一定综合程度的，用于数据分析处理的数据集。第二步，决策树的剪枝：决策树的剪枝是对上一阶段生成的决策树进行检验、校正和修下的过程，主要是用新的样本数据集（称为测试数据集）中的数据校验决策树生成过程中产生的初步规则，将那些影响预衡准确性的分枝剪除。

算法处理流程如下:

1.使用VLD提取Webshell与开源CMS中的Opcode

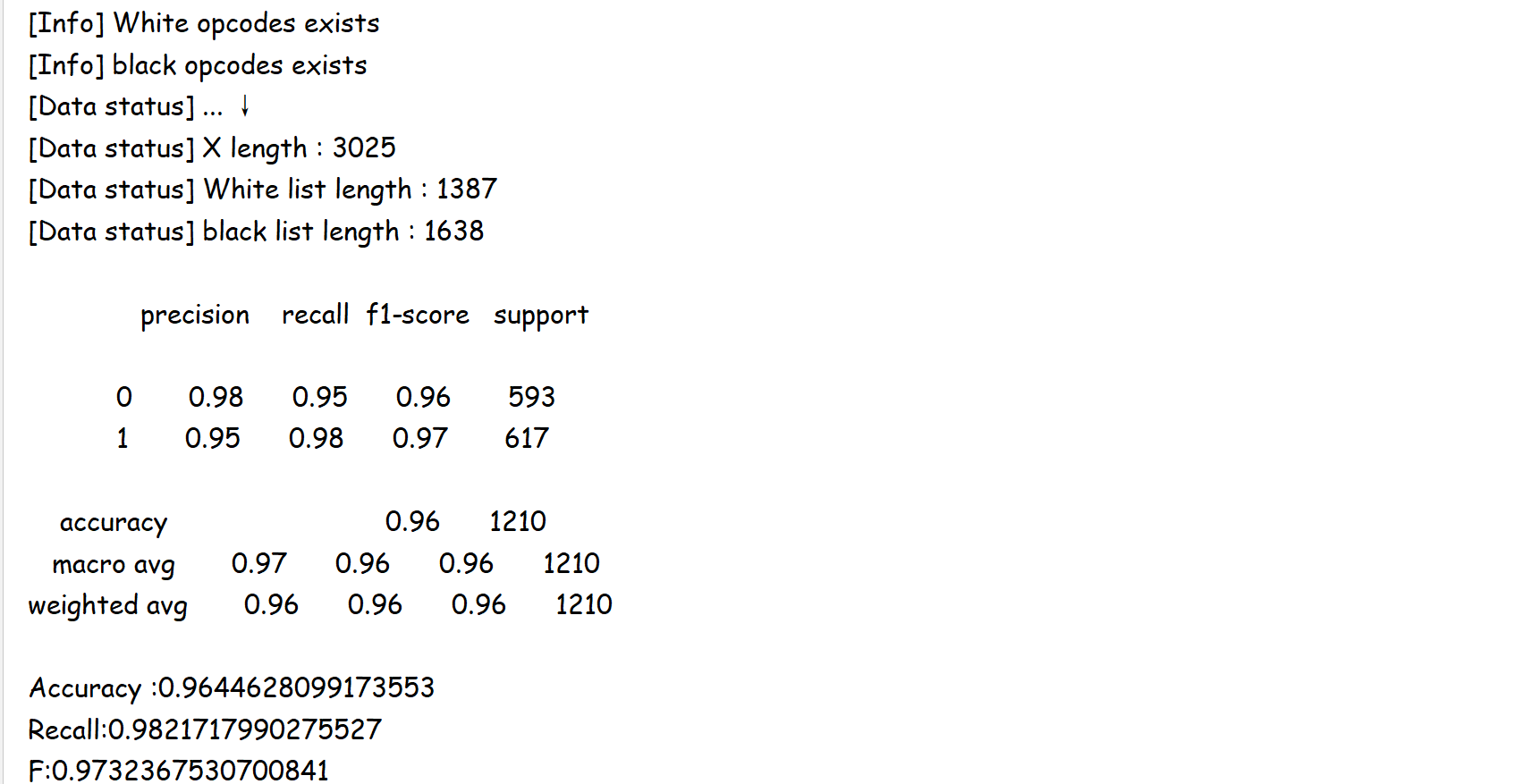
2.随机划分训练集和测试集

3.在训练集上使用决策树进行训练、获取模型数据

4.使用模型数据在测试集上进行预测

5.验证决策树的预测结果

在训练集上训练数据,针对测试集进行检测,图显示了特征提取Opcode特征后,在最大特征数取100的情况下精确率为96.69%，召回率为97.57%，F值为97.13%，如图9所示



2图9 决策树算法

|  |  |  |  |
| --- | --- | --- | --- |
| 训练类型 | 精确率 | 召回率 | F |
| 决策树(第一次) | 0.98 | 0.95 | 0.96 |
| 决策树(第二次) | 0.95 | 0.98 | 0.97 |
| 平均值 | 0.965 | 0.965 | 0.965 |

表4 决策树实现结果

4.4 XGBoost算法

XGBoost(eXtreme Gradient Boosting)是Boosting算法中的一种，是一种提升树模型，将很多树的模型集成起来。其以正则化提升（Regularized Boosting）技术而闻名，通过代价函数里加入正则项，控制模型的复杂度，防止过拟合。可以实现并行处理，相比GBM有了很大的速度提升。Boosting是一种使用多个更简单的模型来拟合数据的学习算法。它所用的这些更加简单、基础的模型也称为基本学习器。其学习的方法是使用参数设置一样或稍有不同的基本学习器来拟合数据。为了得到更加科学的性能，我们选择数据集中的70%的数据样本进行训练,30%的数据样本进行测试,对不同树进行深度测试。

算法处理流程如下：

1.使用VLD提取Webshell与开源CMS中的Opcode

2.设置70%样本数据为训练集,30%样本数据

3.在训练集上使用决策树进行训练、获取模型数据

4.使用模型数据在测试集上进行预测

5.验证决策树的预测结果.

在训练集上训练数据,针对测试集进行检测,图显示了特征提取Opcode特征后,在最大特征数取100的情况下精确率为92.81%,召回率为90.93%，F值为90.98%,如图9所示

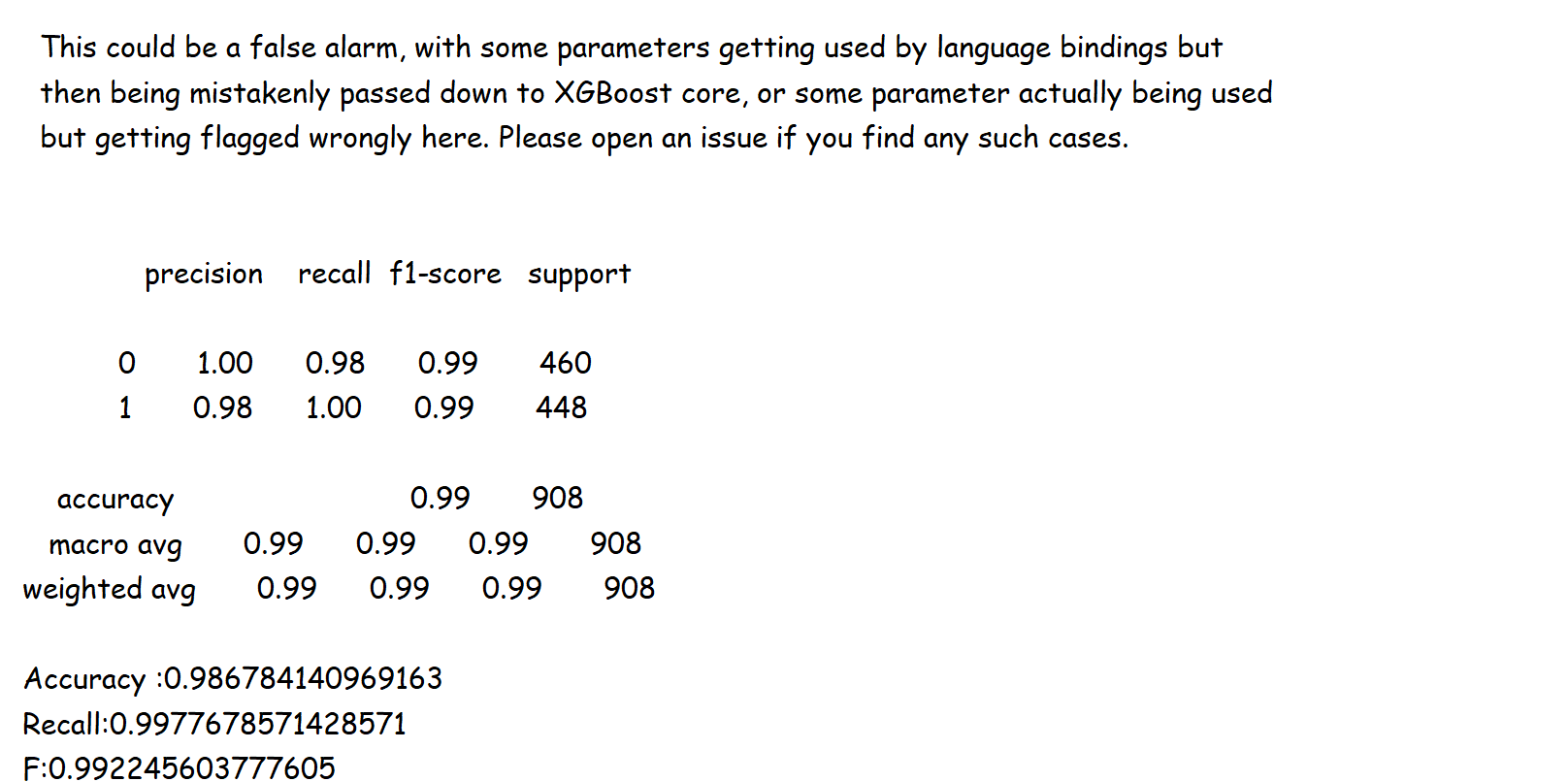


图9 XGBoost算法

|  |  |  |  |
| --- | --- | --- | --- |
| 训练类型 | 精确率 | 召回率 | F |
| XGBoost(第一次) | 1.00 | 0.98 | 0.99 |
| XGBoost(第二次) | 0.98 | 1.00 | 0.99 |
| 平均值 | 0.99 | 0.99 | 0.99 |

表5 XGBoost实现结果

5 性能比较

5.1实验环境

本文实验采用主机为Kali-Linux 20.04系统,处理器为AMD Ryzen 7 5800H,内存为16GB,实验使用Python语言实现,实验环境的具体如表6所示

|  |  |
| --- | --- |
| 配置 | 描述 |
| Kali-Linux 20.04 | 操作系统 |
| Python 3.6 | 编译器环境 |
| PHP 7.3 | 实验环境 |
| Numpy 1.16.3 | 扩展程序库 |
| Scikit-learn | 机器学习库 |

表6 实验环境

5.2 性能评测指标

为了进一步对3种机器学习算法在Webshell中的性能比较,使用精确率,召回率,F值3个指标进行相应的性能评估,评估使用的参数的计算方式如下,其中TP为真正例,TN为真负例，FP为假正例，FN为假负例。

精确率是判断检测中真正例在所有例中的占比情况,体现为检测所有正例中的检测情况,计算公式如下.

召回率是正确判断出的正例占数据集中所有正例的比例,也叫查全率,相应计算公式如下

5.3 PHP类型Webshell文件的检测

本文从精确率、召回率、F值3个方面对使用KNN、随机森林、XGBoost算法的模型进行比较，算法训练过程中内部参数均调至理想水平。PHP类型文件使用JieBa和VLD进行特征提取,提取到的各模型性能指标情况如表7所示

|  |  |  |  |
| --- | --- | --- | --- |
| 训练类型 | 精确率 | 召回率 | F |
| KNN | 0.97 | 0.97 | 0.97 |
| 决策树 | 0.965 | 0.965 | 0.965 |
| XGBoost | 0.99 | 0.99 | 0.99 |

5.4 反思与总结

由于考虑方面的局限性,未考虑到加密、流量、混淆等操作,所以本方案具有一定的局限性,由于公开数据中的样本数量有限,导致算法容易出现过拟合现象,整理搜集更加全面的数据和解决Webshell加密导致的影响,会进一步解决一场样本,提升检测性能。从实验结果来看XGboost算法较KNN和决策树在Webshell危险函数检测中具有更优秀的性能表现,这是因为XGBoost在吸收随机森林的优点的同时,也做到了防止过拟合和进一步运算,具有更优秀的数据集的适应力。

参考文献:

[1]李帅刚,王全民. 基于Bi-GRU的Webshell检测[J]. 计算机系统应用,2021,30(07):259-264.

[2]王世通. 基于HTTP协议的WebShell检测研究[D].北京邮电大学,2021.

[3]韩彤. 基于深度学习的WebShell检测方法的研究与实现[D].北京邮电大学,2021.

[4]周子恒,李琳,赵叙,李凯. 基于深度学习的Webshell恶意代码检测方法研究[J]. 电脑知识与技术,2021,17(07):180-183.

[5]马翔宇,李速. 基于深度学习的Webshell检测的技术研究[A]. 公安部第一研究所.2020中国网络安全等级保护和关键信息基础设施保护大会论文集[C].公安部第一研究所:《计算机工程与应用》编辑部,2020:61-68.

[6]周琳娜,刘旭东,刘冰妍. 基于PHP文件上传漏洞的攻击与防御研究[J]. 信息通信技术,2020,14(06):32-38.

[7]陈艺夫,王平. PHP webshell免杀技术实践与探索[J]. 通信技术,2020,53(12):3078-3086.

[8]岳子涵,薛质,沈兴文,吴毅良. 基于语义分析的PHP Webshell检测方法研究[J]. 通信技术,2020,53(12):3051-3055.

[9]端木怡婷. Webshell检测方法研究综述[J]. 软件,2020,41(11):67-69.

[10]赵瑞杰,施勇,张涵,龙军,薛质. 基于TF-IDF的Webshell文件检测[J]. 计算机科学,2020,47(S2):363-367.