**Маковей Сергей Олегович**

**Сигида Максим Павлович**

Студенты, Московский Политехнический университет

10.05.03

Sergej.magistr98@mail.ru

sigida\_mp@1945bb.edu.ru

**Анализ трафика устройств интернета вещей с помощью алгоритмов машинного обучения**

|  |
| --- |
| Аннотация.  В настоящей статье приводится описание интернета вещей, обозначение его актуальности в данной области, выделение проблем, связанные с угрозами сетевых атак, способы решения данных проблем методами классификации, а также выявление лучшего из представленных классификаторов путем сравнения практических расчетов и полученных результатов.  Ключевые слова:  Интернет вещей, классификатор, сетевые аномалии, k ближайших соседей, наивный Байес, дерево решений. |
|  |
| **Makovey Sergey Olegovich**  **Sigida Maxim Pavlovich**  **Students, Moscow Polytechnic University**  **Analysis of Internet of things device traffic using machine learning algorithms**  Abstract  This article describes the Internet of things, indicates its relevance in this area, highlights the problems associated with threats of network attacks, ways to solve these problems by classification methods, as well as identifying the best of the presented classifiers by comparing practical calculations and the results obtained.  Key words:  Internet of Things, classifier, network anomalies, K nearest neighbors, naive Bayes, decision tree. |
|  |

**Тема проекта:**

«Разработка ПО для разметки многомерных данных»

**Название проекта:**

«Анализ трафика устройств интернета вещей с помощью алгоритмов машинного обучения»

**Назначение практического результата:**

Результатом работы является проведение исследований эффективности методов классификации по выявлению аномалий сетевого трафика и выбор наиболее оптимального метода в качестве решения проблем по быстрой классификации сетевого трафика для обнаружения сетевых атак и аномалий. На основе этих методов должны быть разработаны обученные модели, способные классифицировать трафик Интернета вещей. Основным требованием, предъявляемым к этим моделям, является возможность обнаружения в реальном масштабе времени произвольных типов аномалий, в том числе тех, что были ранее неизвестны.

СОДЕРЖАНИЕ

1. ВВЕДЕНИЕ………………………………………………………………….4

2. §1 Обзор классификаторов для выявления сетевых аномалий.……………………………………………………………………...5

3. § 2 Реализация классификации для выявления атак на устройства IoT……………………………………………………………………………...13

* Описание набора данных……………………………………13
* Подготовка данных………………………………………….16
* Алгоритмы классификации………………………………..18

4. § 3 Сравнительный анализ……………………………………………..21

* Метрики оценки алгоритмов классификации…………..21
* Сравнительный анализ алгоритмов классификации…23

5. Заключение……………………………………………………………….25

6. Список используемых источников ………………………………….26

**Введение**

Для начала, определим, что же такое Интернет вещей.

На данный момент, существует примерно 37 различных определений этого термина. Но в общем случае, название происходит с английского Internet of Things (IoT) и обозначает концепцию вычислительной сети физических предметов (вещей, устройств), которые взаимодействуют c другими устройствами или с внешней средой с помощью встроенных технологий. Т.е. Интернет вещей – это полностью автоматизированный цикл работы устройств и систем за счет их подключения к беспроводной сети.

На сегодняшний день Интернет вещей уже стал неотъемлемой частью жизни людей. Созданная сетевая инфраструктура дала возможность человеку решать задачи, которые до этого приходилось решать самостоятельно. Согласно множеству отчётов и прогнозов от аналитических компаний, концепция «Интернета вещей» будет расширяться и дальше.

Однако у любой системы существуют недостатки. Несмотря на довольно высокий рост развития IoT, одной из основных угроз для устройств IoT является реализация сетевых атак, примеры которых будут затронуты в данной работе для проведения исследований эффективности многоклассовой классификации.

В качестве примера данных для проведения необходимых исследований был подобран датасет, содержащий в себе несколько типов сетевых атак [1].

Целью нашей работы является проведение исследований эффективности методов классификации по выявлению аномалий сетевого трафика. Угрозы, производящие к атакам на большие сети, такие как Интернет, требуют эффективных методов их выявления и реагирования. Наибольшую проблему вызывают атаки, имеющие аномальное поведение в характеристиках выбранного пакета атрибутов сетевого трафика. Обнаружение атаки должно быть таким точным, насколько это возможно. Ложные срабатывания могут привести к нежелательным реакциям. Метод обнаружения должен быть эффективным против самых разнообразных атак, существующих на сегодняшний день, а также быть надежным против будущих попыток злоумышленников уклониться от

обнаружения. Разработка эффективных методов выявления недопустимых событий (аномалий) в работе сети, являющихся следствием технических сбоев или несанкционированных воздействий, является актуальной задачей. Основным требованием, предъявляемым к этим методам, является возможность обнаружения в реальном масштабе времени произвольных типов аномалий, в том числе тех, что были ранее неизвестны.

В качестве способа решения данной проблемы в рамках данной работы были использованы классификаторы – систематизированный перечень наименованных объектов, каждому из которых в соответствие дан уникальный код [2]. Классификация объектов производится согласно правилам распределения заданного множества объектов на подмножества (классификационные группировки) в соответствии с установленными признаками их различия или сходства. Существует множество различных видов классификаторов. Но среди всего перечня было принято решение провести исследование на основе базовых классических классификаторов:

1) Метод наивного Байеса;

2) Метод KNN (k ближайших соседей);

3) Метод "дерево решений"

В качестве языка программирования был выбран Python. Это оптимальный выбор для поставленной перед нами задачи. Python один из самых популярных языков для анализа данных и машинного обучения.

**§ 1. Обзор классификаторов для выявления сетевых аномалий.**

Сетевые атаки уже давно стали привычными в современных вычислительных сетях, их быстрое и эффективное выявление - основная задача для любой крупной сети.

С достаточно быстро растущей пропускной способностью каналов, а также более быстрым появлением неизвестных атак и вредоносных программ большинство современных систем обнаружения вторжений не могут справиться.

Одним из методов обнаружения сетевых атак и вредоносного программного обеспечения (ПО), влияющего на работу вычислительной сети, является детектирование аномалий трафика.

Аномалия в общем случае - это отклонение от нормы либо от общей закономерности в передаваемом трафике. Под данным термином подразумеваются различные виды сетевых атак, в том числе и неизвестные[3].

Для осуществления безопасной работы с данными в вычислительной сети было разработано множество способов и методов обнаружения сетевых аномалий.

Существуют следующие основные методы выявления атак:

- анализ сигнатур;

- статистический анализ;

- контроль целостности;

- анализ систем состояния;

- графы сценариев атак;

- экспертные системы;

- методы, основанные на спецификациях;

- нейронные сети;

- имунные сети;

- поведенческая биометрия;

- циклический анализ[3].

Такие методы как контроль целостности, поведенческая биометрия, циклический анализ и методы, основанные на спецификациях, не позволяют выявить осуществление атаки в данный момент. Они являются инструментами, позволяющими выявить лишь следствие сетевой атаки.

Сравнительные характеристики превентивных методов приведены в таблице 1.

|  |  |  |
| --- | --- | --- |
| Методы: | Достоинства: | Недостатки: |
| анализ сигнатур | скорость анализа, простота работы | отсутствие защиты от неизвестных атак |
| статистический анализ | возможность защиты от неизвестных атак | не выявление изменений в деятельности объекта |
| анализ систем состояния | верифицируемость | существование ситуаций с невозможностью выявить атаки |
| графы сценариев атак | верифицируемость, возможность защиты от неизвестных атак | высокая сложность вычислений |
| экспертные системы | возможность разделения причин и решений проблем | немаштабируемость, сложность учета зависимостей |
| нейронные сети | низкая сложность вычислений, возможность защиты от неизвестных атак | неверифмцмруемость, устойчивость только в обученной сети |
| имунные сети | скорость анализа, возможность защиты от неизвестных атак | неверифмцмруемость, высокая сложность вычислений |

Таблица 1 - Сравнительные характеристики

Анализируя каждый представленный метод, можно сказать, что на данный момент наиболее эффективным и точным методом прогнозирования аномалий является метод с использованием статистического анализа. У каждого представленного метода есть свои достоинства и недостатки, выбор конкретной используемой системы и соответствующего метода обнаружения аномалий зависит от бюджета организации и уровня знаний специалиста по безопасности.

Современные системы обнаружения вторжений (СОВ), представляющие собой совокупность методов обнаружения сетевых атак и аномалий, хорошо справляются со своими задачами. Вместе с тем наиболее эффективные СОВ являются сложными аппаратно-программными комплексами. Так, например, сетевые СОВ для обнаружения аномалий и последующей классификации часто используют накопление статистических данных и технологию DPI (Deep Packet Inspection – проверка и фильтрация сетевых пакетов по их содержимому), что требует серьезных вычислительных мощностей для анализа трафика[4,5].

Таким образом, актуальной становится задача быстрой классификации сетевого трафика для обнаружения сетевых атак и аномалий. Задача может быть поставлена либо как задача бинарной классификации (нормальный или аномальный трафик), либо как более сложная задача многоклассовой классификации, когда аномальный трафик в свою очередь классифицируется по заранее выделенным типам атак.

Вместе с тем широкое распространение в выявлении и классификации аномального поведения в информационных системах получили алгоритмы машинного обучения, способные самообучаться и противостоять новому типу угроз. Многие алгоритмы машинного обучения могут работать достаточно быстро при небольших вычислительных мощностях.

В последнее время было проведено много исследований в области классификации трафика и выявления сетевых атак используются современные методы машинного обучения.

Так, ученые из ЮФУ в статьях [6,7] предлагают метод обнаружения низкоинтенсивных (low-rate) атак типа «отказ в обслуживании» (DDoS). Особенностью метода является предварительная кластеризация пакетов с помощью самоорганизующихся карт Кохонена. Выходной вектор самоорганизующейся карты является входным вектором многослойного персептрона, который осуществляет бинарную классификацию - определяет, является ли набор сетевых пакетов нормальным или атакующим. В результате достигнута ошибка распознавания 0,84%. В статье [8] в качестве эффективного инструмента выявления DDoS-атак предложено использовать нейронную сеть. Для обучения и тестирования нейронной сети использовался набор данных «NSL-KDD». Точность классификации составила 97,87%. Отметим, что предлагаемые подходы рассчитаны на выявление только одного класса атак типа «отказ в обслуживании».

Значительное число публикаций посвящено возможностям использования методов глубокого обучения для обнаружения и классификации сетевых атак. В статье [9] приводится обзор современных публикаций по этой теме. В статье [10] рассмотрены

две задачи классификации атак - бинарная классификация и классификация на 4 класса атак. Авторы используют рекуррентные нейронные сети для классификации большого объема данных. В результате для бинарной классификации достигнута точность менее 0,1% ошибок, для классификации по типу атак - 0,5%.

В статье [11] для обнаружения DDoS атак произведено сравнение рекуррентных нейронных сетей, в том числе LSTM сетей с традиционным методом случайного леса. LSTM сети показали самую высокую точность - 98,4% правильного обнаружения атак. В статье [12] используются сети с автокодировщиком со стохастическим алгоритмом определения порога срабатывания. Этот метод позволил увеличить точность обнаружения атак на наборе NSL-KDD до 88,65%. В статье [13] предлагается система для обнаружения и классификации как известных, так и неизвестных аномалий по 4 классам. Экспериментально определена оптимальная архитектура нейронной сети. В статье [14] рассматривается возможность автоматической кластеризации пакетов для системы обнаружения аномалий в корпоративных сетях. Аномальными считаются большие кластеры с высокой плотностью, а также малые или разреженные кластеры. Далее на этих данных обучаются алгоритмы бинарной классификации. На наборе NSL-KDD удается получить точность классификации 88%.

В статье [15] рассматривается классификация атак в беспроводных сетях IEEE 802.11. Атаки классифицируются на 3 класса с помощью многослойного автокодировщика. В работе [16] для классификации вредоносного трафика используются сверточные нейронные сети. Идея состоит в том, что сырые данные трафика преобразуются в изображения, которые распознаются сверточными сетями. При этом точность детектирования атак достигает 99.41%. В работе [17] для детектирования трудно обнаруживаемого типа атак - сканирования портов и поиска уязвимостей используются глубокие сети доверия, комбинирующие подходы обучения с учителем и без учителя. В статье [6] так же представлен метод бинарной классификации атак на основе метода нечеткой кластеризации C-средних. Для повышения точности алгоритма используется частичная ручная разметка небольшой части обучающих данных. Подробное сравнение различных алгоритмов машинного обучения, применяемых в системах информационной безопасности приведено в статье [6]. Авторы рассматривают три задачи - обнаружение вторжений, анализ вредоносных программ и обнаружение спама. Выводы - для каждой задачи лучше применять свои методы, которые требуют непрерывного обучения и тщательной настройки параметров. В статье [7] так же рассматриваются методы кластеризации для обнаружения вторжений на основе метода к-средних.

Отметим, что методы глубокого обучения не обладают высокой производительностью особенно на этапе обучения, также не исследованы методы, которые позволяют проводить корректную классификацию более чем по 4 классам атак.

Все вышеперечисленные работы проводились на уже старых наборах данных KDD’99 и его модификации NSL-KDD (2014). В отличии от них, в настоящей работе объектом исследования является самый новый из доступных на данный момент набор данных сетевого трафика IoT-23 опубликованный в 2020 году и более комплексный анализ алгоритмов классификации.

В нашем исследовании проведено сравнение традиционных методов машинного обучения как для бинарной, так и многоклассовой классификации сетевого трафика по 9 типам атак.

Бинарная классификация

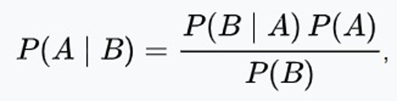
Обучению подвергались следующие классификаторы из библиотеки scikit-learn:

• DecisionTree - алгоритм решающих деревьев;

• KNeighbors - алгоритм k-ближайших соседей;

• GaussianNB - г[ауссовский наивный байесовский классификатор.](http://espressocode.top/naive-bayes-classifiers/" \t "_blank)

1) Теорема Байеса находит вероятность наступления события, учитывая вероятность другого события, которое уже произошло. Теорема Байеса математически сформулирована как следующее уравнение:



В гауссовском наивном байесовском методе предполагается, что непрерывные значения, связанные с каждым признаком, распределены в соответствии с **распределением Гаусса**. Гауссово распределение также называется нормальным распределением. При построении график дает кривую в форме колокола, которая симметрична относительно среднего значения объекта.  
Вероятность появления признаков предполагается Гауссовой:

Параметры  и  оцениваются с использованием максимального правдоподобия.

2) В многомерном пространстве каждая частица p может быть представлена своими координатами. Термин частица в данном контексте заменим термином точка. Точки p1 и p2 будут являться соседями, если расстояние между ними не превышает величину радиуса взаимодействия r. Предлагаемый алгоритм позволяет закодировать множество исходных точек в виде линейного списка, в котором частицы находятся рядом (в порядке следования номеров) со своими ближайшими соседями, так что поиск очередной пары соседей будет осуществляться путем сопоставления очередной точки со следующими (по порядку следования номеров) элементами списка. Заметим, что для точек p1 и p2, являющихся ближайшими соседями, будет выполняться неравенство r. (1)≤2|ρ1–ρ| То есть две соседние точки отдалены от третьей на расстояние, не превышающее радиус взаимодействия r. 2. Тогда p1 и p2 расположатся на окружностях. Очевидно, что при таком кодировании ближайшие соседи будут находиться на одной и той же или на соседних окружностях. Таким образом, для поиска ближайших соседей достаточно выполнить следующие шаги. Листинг 1. Обобщенный алгоритм поиска ближайших соседей.  
1. Упорядочить все множество частиц по возрастанию значений α и ρ  
2. для i←1, …, N,  
3. для j←i+1, …, N,  
4. если ρj–ρi≤ r,  
5. тогда (pi , pj) – ближайшие соседи,  
6. иначе переход на шаг 3.

Упорядочение множества частиц (шаг 1) по значениям ρ гарантирует, что точки на одной окружности будут находиться в упорядоченном множестве рядом, то есть иметь соседние порядковые номера, а упорядочение по значению α гарантирует, что эти же точки в пределах одной окружности будут упорядочены снизу вверх по дуге.

3) Алгоритм дерева принятия решений строит модель интеллектуального анализа данных путем создания ряда разбиений в дереве. Эти разбиения представлены как узлы. Алгоритм добавляет узел к модели каждый раз, когда выясняется, что входной столбец имеет значительную корреляцию с прогнозируемым столбцом. Способ, которым алгоритм определяет разбиение, отличается в зависимости от того, прогнозирует ли он непрерывный столбец или дискретный столбец.

Алгоритм дерева принятия решений использует функцию выбора компонентов для выбора наиболее полезных атрибутов.

Для работы нам нужны данные. Причем к данным выдвигаются конкретные требования:

* “свежесть”
* объемность
* интерпретируемость признаков
* удобный формат
* высокое качество проработки классов
* предобработка человеком

Данные должны быть новыми. Чем новее, тем лучше. Методы злоумышленников меняются очень быстро. Из-за чего черты присущие определенным атакам еще год назад могут измениться или вовсе исчезнуть. Оптимальный выбор, если данные за прошлый год, так как на их сбор и качественную предобработку нужно много времени, более новые данные хорошего качества достать не видится возможным.

Данные должны быть большие как по числу объектов (замеры трафика) так и по числу замеряемых признаков. Нужно будет выделять из данных объекты на обучение, на проверку моделей, на итоговую проверку. Большое количество данных также дает возможность использовать больше методов в обучении (например, кросс валидация) и уменьшает вероятность переобучения моделей. Большое количество признаков в принципе дает больше данных для обучения моделям. Также в данных с большой вероятностью будут сильно коррелирующие и вредные признаки, которые придется выкидывать. Оптимально число объектов должно быть не менее 10 миллионов с 20 или более признаками.

В бизнес задачах не всегда требуется интерпретируемость, главное, чтобы модель улучшала необходимую метрику. Нам же для анализа и исследования нужно понимать на что именно реагирует конкретная модель и чем этот раздражитель является в реальном мире. Это дает возможность лучше понять, что мешает, а что помогает правильной классификации.

Данные должны быть представлены в стандартном удобном формате (.csv, .raw, .xls/.xlsx, .mat). Это обеспечивает удобность и простоту чтения данных, а также независимость от платформы (UNIX/Windows).

В исследуемых данных должно быть выделено как можно больше классов. Это делает задачу классификации более сложной, но повышает простор исследований и ценность самой данной работы. Оптимально классов должно быть не меньше 8.

Лучше если в классификации данных использовали ручные методы специалистов, а не только автоматические методы. Это улучшает качество классификации, а также начинают лучше фиксироваться выбросы и аномалии в данных.

Для проведения дальнейшей работы мы выбрали набор данных [IoT - 23](https://www.stratosphereips.org/datasets-iot23) созданный в лаборатории Avast AIC laboratory. Он подошел по всем выдвинутым к данным требованиям:

* опубликован 22 января 2020 года
* включает в себя более 325 миллионов объектов и более 20 признаков
* признаки понятны и интерпретируемые
* данные представлены в формате легко преобразуемом в .csv
* включает в себя более 10 классов
* была выполнена тщательная предобработка специалистами

**§ 2. Реализация классификации для выявления атак на устройства IoT.**

**Описание набора данных**

Исследование проводилось на наборе данных IoT-23 от "Stratosphere Laboratory" опубликованные 22 января 2020 года (<https://www.stratosphereips.org/datasets-iot23>). Данный набор данных имеет 23 захвата сетевого трафика в диапазоне от 2018 до 2019 года, выполненных на устройствах IoT, 20 вредоносных захватов и 3 захвата для безопасного трафика устройств IoT. Этот сетевой трафик IoT был захвачен в "Stratosphere Laboratory", AIC group, FEL, CTU University, Чешская Республика.

Каждая запись в наборе представляет собой образ сетевого соединения и включает 23 информационных признака – индивидуальное измеряемое свойство или характеристику наблюдаемого явления – и промаркирована как «атака» или «не атака».

В исследуемом наборе атаки делятся на девять основные категории:

**Attack:** эта метка указывает на то, что был некоторый тип атаки от зараженного устройства к другому хосту. Здесь мы обозначаем как атаку на любой поток, который, анализируя его полезную нагрузку и поведение, пытается воспользоваться некоторыми уязвимыми услугами. Например, грубая сила к некоторому логину telnet, инъекция команды в заголовок запроса GET и т. д.

**C&C:** эта метка указывает, что зараженное устройство было подключено к серверу CC. Эта активность была обнаружена при анализе захвата вредоносных программ в сети, потому что соединения с подозрительным сервером являются периодическими, или наше зараженное устройство загружает с него некоторые двоичные файлы, или некоторые IRC-подобные или декодированные заказы приходят и уходят с него.

**DDoS:** эта метка указывает, что зараженное устройство выполняет распределенную атаку отказа в обслуживании. Эти потоки трафика обнаруживаются как часть DDoS-атаки из-за количества потоков, направленных на один и тот же IP-адрес.

**FileDownload:** эта метка указывает, что файл загружается на наше зараженное устройство. Это обнаруживается путем фильтрации соединений с байтами ответа более 3 КБ или 5 КБ, обычно это сочетается с некоторым известным подозрительным портом назначения или IP-адресом назначения, известным как сервер C&C.

**HeartBeat:** эта метка указывает, что пакеты, отправленные по этому соединению, используются для отслеживания зараженного хоста сервером C&C. Это было обнаружено путем фильтрации соединений с байтами ответа ниже 1B и с периодическими аналогичными соединениями, обычно это сочетается с некоторым известным подозрительным портом назначения или IP-адресом назначения, известным как сервер C&C.

**Mirai:** эта метка указывает, что соединения имеют характеристики ботнета Mirai. Эта метка добавляется, когда потоки имеют аналогичные шаблоны, как наиболее распространенные известные атаки Mirai.

**Okiru:** эта метка указывает, что соединения имеют характеристики ботнета Okiru. Это решение о маркировке было сделано с теми же параметрами, что и с Mirai, но с той разницей, что это семейство ботнетов менее распространено.

**PartOfAHorizontalPortScan:** эта метка указывает, что соединения используются для выполнения горизонтального сканирования портов для сбора информации для выполнения дальнейших атак. Для размещения этих меток мы полагаемся на шаблон, в котором соединения совместно используют один и тот же порт, одинаковое количество передаваемых байтов и несколько различных IP-адресов назначения.

**Torii:** эта метка указывает, что соединения имеют характеристики ботнета Torii. Это решение о маркировке было сделано с теми же параметрами, что и с Mirai, но с той разницей, что это семейство ботнетов менее распространено.

Метка указывающая на то, что в соединениях не было обнаружено никаких подозрительных вредоносных действий: **Benign.**

Три наиболее распространенных вредоносных (недоброкачественных потоков) метки являются: PartOfAHorizontalPortScan (213 852 920 потоков), Okiru (60 990 707 потоков) и DDoS (19 472 910 потоков). В то время как три наименее распространенные вредоносные (недоброкачественные потоки) метки: C&C-Mirai (2 потока), Okiru-Attack (3 потока) и PartOfAHorizontalPortScan-Attack (5 потоков).



Рисунок 0 - Распределение целевого признака (%)

**Подготовка данных**

Особенность нашего набора данных заключается в том, что данные в нем уже размечены специалистами, которые, собственно, и собирали данные. Однако, мы не можем провести полноценную исследовательскую работу, не предобработав данные. На это есть несколько причин:

а) Многие алгоритмы машинного обучения способны работать только с числовыми признаками, т.е. им нельзя передавать наши категориальные и порядковые данные. Даже некоторые числовые наши данные не будут обработаны, т.к. они представлены не в числовых типах данных, а в текстовом.

б) Существуют некоторые опечатки в исходных данных, которые могут сильно сказаться на точности работы обученных моделей. Таким образом, в целевом признаке один и тот же класс может быть записан как с прописной, так и со строчной буквы. Для абсолютно любого алгоритма машинного обучения эти буквы будут находиться в разных классах.

в) Очень большая несбалансированность классов. Имеются классы с миллионами записей, а имеются совсем незначительные, с количеством записей меньше десяти. Самую большую проблему представляют как раз вторые. Их нельзя сразу все включать в обучающую выборку, т.к. в таком случае любая модель просто переобучится на данных классах. Но если оставить для проверки модели всего одно значение, то мы не сможем правильно оценить точность работы модели. При сбалансированном разбиении маленьких классов на обучающую и тестовую выборку скорее всего модели плохо обучатся, и вместе с этим мы не сможем поверить, какая именно модель лучше работает с данными классами из-за слишком малой тестовой выборки.

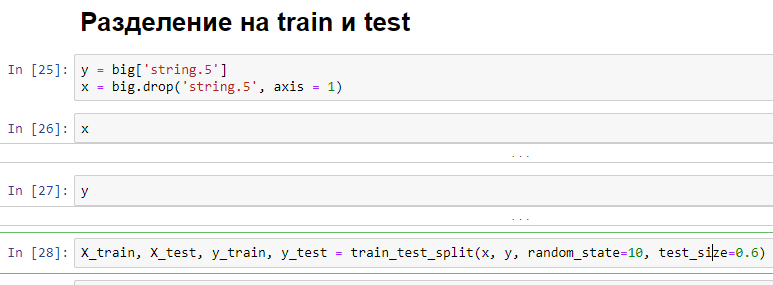
г) Некоторые алгоритмы машинного обучения работают со всеми признаками, как с одним вектором, а поскольку в наших данных значения признаков неоднородны, это может привести к серьезному снижению качества работы обученных моделей.

Для правильной оценки качества работы классификаторов и для отсутствия проблемы предобученных данных (проблема переобучения) данные были разделены на тренировочную и тестовую части.

Модели обучаются только по тренировочному набору данных. Даже при использовании кроссвалидации для определения оптимальных параметров модели, кросвалидация выполняется только на тренировочных данных.

В итоговом наборе данных полностью отсутствуют элементы из тренировочной части. Таким образом обученная нами модель на тренировочных данных точно не знает точного ответа на любой из элементов итоговой выборки и будет выдавать предсказание опираясь на правила и закономерности выработанные в ходе обучения.

В своем исследовании мы разделили исходный набор данных в соотношении 60% на тренировочную выборку и 40% на тестовую выборку. Это дает достаточно равномерное распределение, так большой размер тренировочной выборки снижает вероятность проблемы недообучения, когда модели не хватило данных для обучения. Так же большой размер тестовой выборки повышает точность полученных оценок при масштабировании как на общий больший размер данных, так и большее число объектов каждого класса. Так же было применено перемешивание данных при распределении по выборкам параметром random\_state равным десяти для лучшего разделения каждого класса на тренировочную и тестовую выборки. Программный код создания тренировочной и тестовой выборок представлен на рисунке 1.

  
Рис. 1 - Создание тренировочной и тестовой выборок

**Алгоритмы классификации**

Для классификации рассматриваемого набора данных были использованы следующие алгоритмы машинного обучения

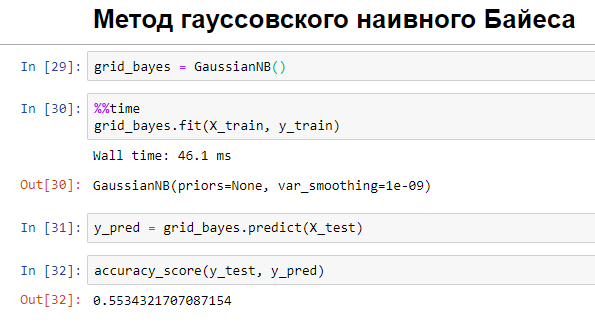
1) Мультиномиальный наивный Байес в гауссовском варианте;

2) Метод К-ближайших соседей (KNN);

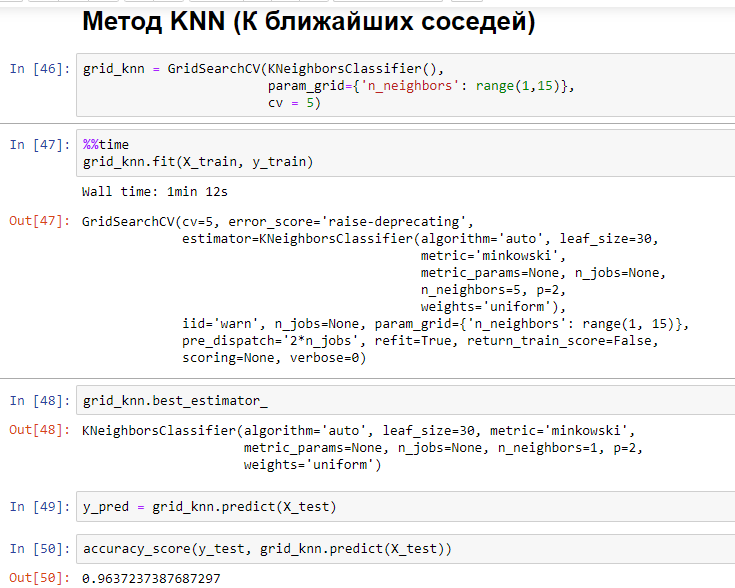
3) Дерево решений.

Все алгоритмы были реализованы с помощью стандартной библиотеки scikit-learn. Так в качестве наивного Байеса рассматривался классификатор GaussianNB(). В качестве метода к-ближайших соседей классификатор KNeighborsClassifier(), а в качестве дерева решений классификатор DecisionTreeClassifier().

# В методе гауссовского наивного Байеса использовались стандартные параметры без кроссвалидации. Данный метод показал наилучший результат по скорости обучения. Программный код данного метода представлен на рисунке 2.

  
Рис. 2 – Метод гауссовского наивного Байеса

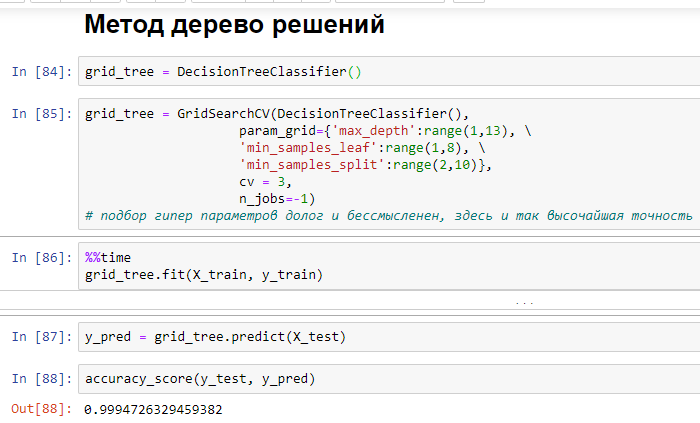
В методе К-ближайших соседей (KNN) с помощью кроссвалидации методом GridSearchCV() с числом валидаций равным 5 было определено и использовано оптимальное количество соседей по которым определяется принадлежность рассматриваемого объекта к определенному классу. Количество соседей дающих наибольшую точность оказалось равно одному. Программный код данного метода представлен на рисунке 3.

  
Рис. 3 – Метод KNN

В методе дерево решений с помощью кроссвалидации методом GridSearchCV() с числом валидаций равным 3 были определены и использованы следующие параметры:

1. Максимальная глубина дерева (max\_depth) равная 8;  
2. Минимальное количество выборок, необходимых для разделения внутреннего множества (min\_samples\_split) равное двум;  
3) Минимальное количество выборок у листа (min\_samples\_leaf) равное одному.

Программный код данного метода представлен на рисунке 4.

  
Рис. 4 – Метод дерево решений

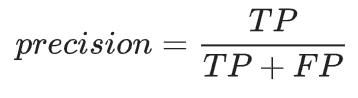
**§ 3 Сравнительный анализ.**

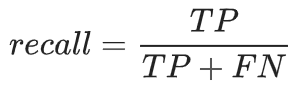
**Метрики оценки алгоритмов классификации**

В задачах машинного обучения наиболее часто используются следующие метрики для оценки эффективности построенных моделей[18]: точность (precision), полнота (recall), F-мера (F-score), ROC-кривые (Receiver Operating Characteristic curve - кривая ошибок), AUC-ROC и AUC-PR(Area Under Curve - площадь под кривой ошибок и площадь под кривой precision-recall).  
После проведения классификации возможно получение четырех видов результатов: TP (True Positive – истинно положительный), TN (True Negative – истинно отрицательный), FP (False Positive – ложно положительный) и FN (False Negative – ложно отрицательный). Эти результаты можно представить в виде матрицы ошибок в таблице 2, где 𝑦𝑦′ – ответ алгоритма на объекте, а 𝑦𝑦 – истинная метка класса на этом объекте.

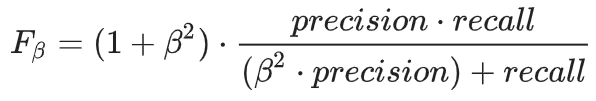
|  |  |  |
| --- | --- | --- |
|  | y = 1 | y = 0 |
| 𝑦′ = 1 | True Positive (TP) | False Positive (FP) |
| 𝑦′ = 0 | False Negative (FN) | True Negative (TN) |

Таблица 2 - Матрица ошибок классификации

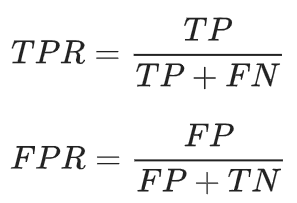
Точность (precision) показывает долю объектов, названных классификатором положительными и при этом действительно являющимися положительными:  


Полнота (recall) показывает долю правильно помеченных положительных объектов среди всех объектов положительного класса:  


Точность чувствительна к распределению данных, в то время как полнота – нет. Полнота не отражает, сколько объектов помечены как положительные неверно, а точность не дает никакой информации о том, сколько положительных объектов помечены неправильно.

F-мера (F-score, Fβ) сочетает в себе вышеупомянутые две метрики – среднее гармоническое точности и полноты:  
 ,  
где β – принимает значения в диапазоне 0 < β < 1, если приоритет отдается точности, и β > 1, если приоритет отдается полноте.

F-мера достигает максимума при полноте и точности, равными единице, и близка к нулю, если один из аргументов близок к нулю.

ROC-кривая или кривая ошибок – график, позволяющий оценить качество классификации, который отображает соотношение между чувствительностью (TPR, True Positive Rate) алгоритма и долей из объектов отрицательного класса, которые алгоритм предсказал неверно (FPR, False Positive Rate) при варьировании порога решающего правила:  


Количественную интерпретацию ROC-кривой дает показатель AUC-ROC. В идеальном случае, когда классификатор не делает ошибок (FPR = 0, TPR = 1), получается площадь под кривой, равная 1; если классификатор «гадает», то AUC-ROC будет стремиться к 0,5, так как классификатор будет выдавать одинаковое количество TP и FP. Площадь под кривой в этом случае показывает качество алгоритма (больше – лучше); кроме этого, важной является крутизна самой кривой (желательно максимизировать TPR, минимизируя FPR), а значит, она в идеале должна стремиться к точке (0,1)

Критерий AUC-ROC устойчив к несбалансированным классам и может быть интерпретирован как вероятность того, что случайно выбранный положительный объект будет ранжирован классификатором выше (будет иметь более высокую вероятность быть положительным), чем случайно выбранный отрицательный объект.

Помимо ROC-кривой существует PR-кривая (от англ. Precision-Recall Curve), показывающая отношение точности (2) от полноты (3). Соответственно, количественный показатель в данном случае такой же – площадь по кривой – AUC-PR (больше – лучше). В идеальном случае кривая должна стремиться к точке (1,1), где классификатор получает только истинно положительные результаты, без отрицательных. PR-анализ также применяется на несбалансированных данных.

**Сравнительный анализ алгоритмов классификации**

На рисунке 5 показаны матрицы ошибок для каждого из рассматриваемых алгоритмов машинного обучения (KNN, Наивный Байес, Дерево решений соответственно). Они показывают количество меток, присвоенных алгоритмом (Predicted label), и их соответствие истинным меткам (True label). Значения в матрице ошибок нормализованы относительно количества записей каждого класса.

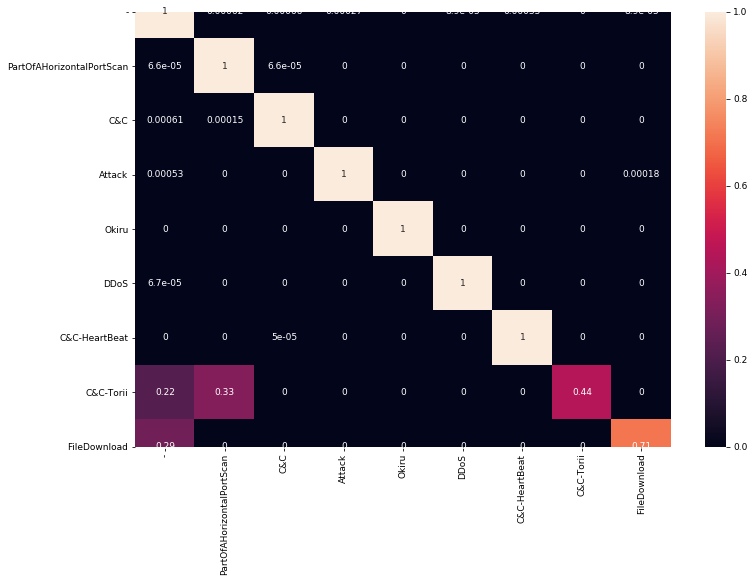
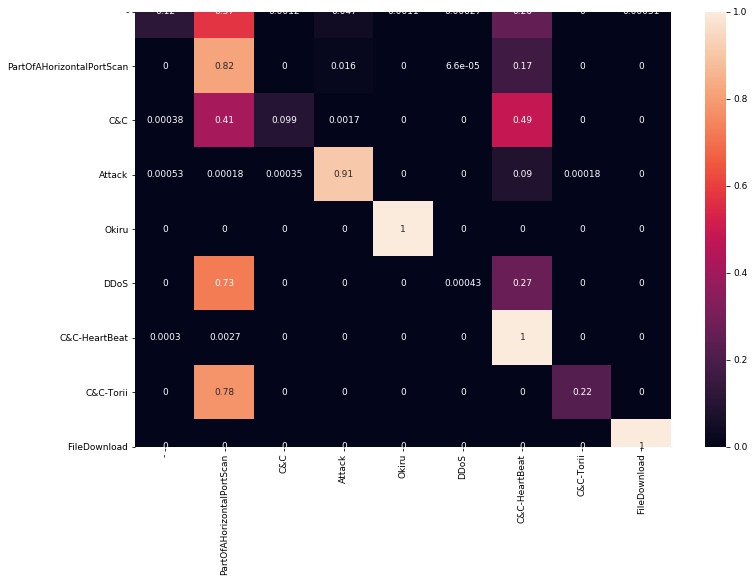
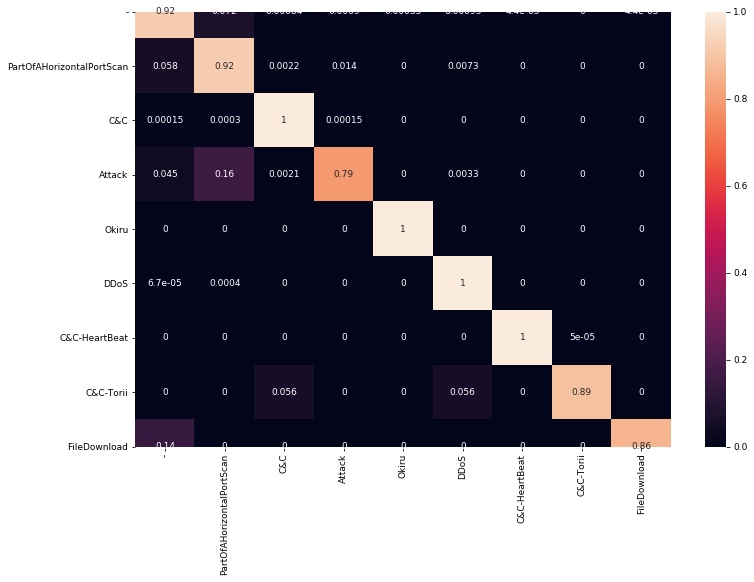


Рисунок 5 – Матрица ошибок для каждого алгоритма

**Заключение.**

Из анализа представленных данных можно сделать вывод, что из-за несбалансированности обучающих данных алгоритмы машинного обучения показали себя достаточно плохо на исследуемом наборе. Для решения этой проблемы требуется сбалансировать классы, например, путем дублирования или перейти к бинарной классификации – «нормальный» vs «не нормальный».

Однако если рассмотреть определение наиболее сбалансированных классов вредоносного трафика (PartOfAHorizontalPortScan, Okiru и DDoS), то лучше всего себя показал алгоритм дерева решений. Алгоритм с методом ближайших соседей так же показал хороший результат, однако все же уступает алгоритму дерева решений. Алгоритм наивного Байеса показал себя хуже всего из-за того, что он слишком часто ошибается, ложно относя трафик к одному из двух классов (PartOfAHorizontalPortScan и C&C -HeartBeat).

Таким образом, для многоклаcсовой классификации вредоносного трафика из исследованных алгоритмов машинного обучения лучше всего подходит алгоритм дерева решений.

**Список используемых источников.**

1. Aposemat\_IoT-23-master (https://www.stratosphereips.org/datasets-iot23) – Набор данных, предоставленный Чешским университетом.

2. Шелухин О.И., Симонян А.Г., Ванюшина А.В. Влияние структуры обучающей выборки на эффективность классификации приложений трафика методами машинного обучения // T-Comm: Телекоммуникации и транспорт. 2017. Т. 11. № 2. С. 25–31.

3. Emel'yanov V.N., Emel'yanova M.M., Zinov'eva E.L., Shamonov M.U. (2016). Analiz metodov obnaruzheniya anomaliy setevogo trafika // V sbornike: Priborostroenie v XXI veke - 2015. Integraciya nauki, obrazovaniya I proizvodstva. Sbornik materialov Xi Mezdunarodnoy naucho-technicheskoy konferencii, pp. 420-425

4. Шелухин О.И. Сетевые аномалии. Обнаружение, локализация, прогнозирование. М.: Горячая линия–Телеком, 2019. 448 с. - Сборник научных трудов

5. Шелухин О.И., Сакалема Д.Ж., Филинова А.С. Обнаружение вторжений в компьютерные сети (сетевые аномалии). М: Горячая линия–Телеком, 2016. 220 c. - Сборник научных трудов

6. Абрамов Е.С., Тарасов Я.В. Применение комбинированного нейросетевого метода для обнаружения низкоинтенсивных DDoS-атак на web-сервисы // Инженерный вестник Дона. 2017. Т. 46, № 3(46). C. 59. URL: https://elibraryru/item.asp?id=30753050

7. Тарасов Я.В. Исследование применения нейронных сетей для обнаружения низкоинтенсивных DDoS-атак прикладного уровня // Вопросы кибербезопасности.  
2017. № 5(24). C. 23-29. DOI: 10.21681/2311-3456-20175-23-29

8. Воробьёва Ю.Н. и др. Нейросетевая модель выявления DDOS-атак // Вестник технологического университета.  
2018. Т. 21, №. 2. С. 94-98. URL: https://elibrary.ru/item.as-p?id=32683897

9. Bodström T., Hämäläinen T. State of the Art Literature Review on Network Anomaly Detection with Deep Learning / [16] O. Galinina, S. Andreev, S. Balandin, Y. Koucheryavy (Eds.) // Internet of Things, Smart Spaces, and Next Generation Networks and Systems. NEW2AN 2018, ruSMART 2018. Lecture Notes in Computer Science. Vol. 11118. Springer, Cham, 2018. Pp. 64-76. DOI: 10.1007/978-3-030-01168-0\_7

10. Yin C., Zhu Y., Fei J., He X. A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks // IEEE Access. 2017. Vol. 5. Pp. 21954-21961. DOI: 10.1109/AC-cess.2017.2762418 References

11. Yuan X, Li C., Li X. DeepDefense: Identifying DDoS Attack via Deep Learning // 2017 IEEE International Conference on [1] Smart Computing (SMARTCOMP). Hong Kong, 2017. Pp. 1-8. DOI: 10.1109/SMARTCOMP.2017.7946998

12. Aygun R.C., Yavuz A.G. Network Anomaly Detection with Stochastically Improved Autoencoder Based Models // 2017 IEEE 4th International Conference on Cyber Security and [2] Cloud Computing (CSCloud). New York, NY, 2017. Pp. 193198. DOI: 10.1109/CSCloud.2017.39

13. Van N., Thinh T., Sach L. An anomaly-based network intrusion detection system using Deep learning // 2017 International Conference on System Science and Engineering (ICSSE). Ho Chi Minh City, 2017. Pp. 210-214. DOI: 10.1109/ ICSSE.2017.8030867

14. Baek S., Kwon D., Kim J., Suh S.C., Kim H., Kim I. Unsupervised Labeling for Supervised Anomaly Detection in Enterprise and Cloud Networks // 2017 IEEE 4th International Conference on Cyber Security and Cloud Computing (CSCloud). New York, NY, 2017. Pp. 205-210. DOI: 10.1109/CSCloud.2017.26

15. Thing V.L.L. IEEE 802.11 Network Anomaly Detection and Attack Classification: A Deep Learning Approach // 2017 IEEE Wireless Communications and Networking Conference (WCNC). San Francisco, CA, 2017. Pp. 1-6. DOI: 10.1109/ WCNC.2017.7925567

16. Wang W., Zhu M., Zeng X., Ye X., Sheng Y. Malware traffic classification using convolutional neural network for representa- [6] tion learning // 2017 International Conference on Information Networking (ICOIN). Da Nang, 2017. Pp. 712-717. DOI: 10.1109/1CO1N.2017.7899588

17. Viet H.N., Van Q.N., Trang L.L.T., Nathan S. Using Deep Learning Model for Network Scanning Detection // Proceedings of the 4th International Conference on Frontiers of Educational Technologies (ICFET '18). ACM, New York, NY, USA, 2018. Pp. 117-121. DOI: 10.1145/3233347.3233379

18. Bishop C.M. Pattern Recognition and Machine Learning. Berlin: Springer, 2006.