# АННОТАЦИЯ

Целью данного проекта является использование возможностей методов ИИ для реализации защиты от сетевых атак. В работе будут рассмотрены основные концепции и методы, применяемые для обнаружения и предотвращения атак, включая алгоритмы машинного обучения.

Для достижения этой цели необходимо выполнить ряд задач:

* Разобрать законодательство РФ в сфере ИБ
* Рассмотреть меры и технологии, направленных на защиту сетей
* Оценить потенциал применения ИИ для защиты от сетевых угроз
* Разобрать рекомендации по защите систем с использованием ИИ
* Проанализировать основные методы машинного обучения
* Использовать один из методов МО для защиты от сетевых атак

**Оглавление**

[АННОТАЦИЯ 1](#_Toc168535358)

[СОКРАЩЕНИЯ 3](#_Toc168535359)

[ВВЕДЕНИЕ 4](#_Toc168535360)

[1. ВЕДЕНИЕ В СЕТЕВУЮ БЕЗОПАСНОСТЬ 5](#_Toc168535361)

[1.1. Законодательство в сфере ИБ 5](#_Toc168535362)

[1.2. Сетевая безопасность 7](#_Toc168535363)

[1.3. Методы защиты 11](#_Toc168535364)

[Вывод по первой главе: 15](#_Toc168535365)

[2. ЗАЩИТА ИНФОРМАЦИОННЫХ СИСТЕМ ОТ СЕТЕВЫХ УГРОЗ 16](#_Toc168535366)

[2.1. Использование систем ИИ в ИБ 16](#_Toc168535367)

[2.2. Инструменты распознавания сетевых атак 19](#_Toc168535368)

[2.3. Архитектура и особенности нейронных сетей 22](#_Toc168535369)

[2.4. Машинное обучение 29](#_Toc168535370)

[2.5. Угрозы системам с использованием ИИ 33](#_Toc168535371)

[Вывод по второй главе: 35](#_Toc168535372)

[3. ПРИМЕНЕНИЕ МЕТОДА МАШИННОГО ОБУЧЕНИЯ ДЛЯ КЛАССИФИКАЦИИ СЕТЕВЫХ АТАК 36](#_Toc168535373)

[3.1. Метод машинного обучения «случайный лес»: основные принципы и применение 36](#_Toc168535374)

[3.2. Существующие аналоги случайного леса 39](#_Toc168535375)

[3.3. Алгоритм и математическое описание выбранного метода 40](#_Toc168535376)

[3.4. Вывод по третьей главе 48](#_Toc168535377)

[ЗАКЛЮЧЕНИЕ 49](#_Toc168535378)

[Список использованных источников 50](#_Toc168535379)

[Приложение 52](#_Toc168535380)

# СОКРАЩЕНИЯ

В данной работе используются следующие сокращения:

МО – Машинное обучение;

АИС - Автоматизированная информационная система;

ВПО - Вредоносное программное обеспечение;

ЗИ - Защита информации;

ИБ - Информационная безопасность;

ИИ - Искусственный интеллект;

ИС - Информационная система;

МЭ -Межсетевой экран;

КИИ - Критическая информационная инфраструктура;

ПДн - Персональные данные;

ПО - Программное обеспечение;

РФ - Российская Федерация;

ФЗ - Федеральный закон;

ЭВМ - Электронная вычислительная машина.

# ВВЕДЕНИЕ

В век стремительного развития информационных технологий и глобальной цифровизации защита данных и инфраструктуры от сетевых атак является критически важной задачей для любой организации. С каждым годом количество и сложность атак возрастает, что требует разработки и внедрения эффективных методов защиты. Традиционные подходы, основанные на статических правилах, становятся все менее эффективными перед лицом новых, более опасных угроз.

В этом контексте методы искусственного интеллекта (ИИ) представляются многообещающим решением. ИИ может анализировать большие объемы данных, выявлять аномалии в реальном времени и адаптироваться к новым видам атак. Применение методов машинного обучения и глубокого обучения позволяет системам защиты не только реагировать на известные угрозы, но и предсказывать появление новых, что значительно повышает уровень безопасности.

Использование ИИ в реализации защиты от сетевых атак позволяет автоматизировать процессы мониторинга и реагирования, снижая таким образом человеческий фактор и ускоряя время реакции на инциденты.

# ВЕДЕНИЕ В СЕТЕВУЮ БЕЗОПАСНОСТЬ

# Законодательство в сфере ИБ

В России существует множество законов и нормативных актов, которые регулируют сферу информационной безопасности (ИБ) и направлены на защиту информации, систем и пользователей от различных видов угроз. В основе законодательства Российской Федерации лежит Конституция РФ. Она включает в себя нормы, которые устанавливают правовые основы информационной безопасности: ключевые аспекты правового положения участников информационных отношений, принципы информационной безопасности (законность, соблюдение прав, баланс интересов личности, общества и государства), а также конституционный статус государственных органов, ответственных за обеспечение информационной безопасности. Основные законы и нормативные акты:

1. Федеральный закон «О персональных данных» (ФЗ-152):

* Определяет правила сбора, хранения, обработки и защиты персональных данных.
* Обязует операторов персональных данных предпринимать необходимые меры для их защиты.
* Требует уведомления Роскомнадзора о намерении обрабатывать персональные данные.
  + 1. Федеральный закон «О безопасности критической информационной инфраструктуры Российской Федерации» (ФЗ-187):
* Устанавливает меры защиты критической информационной инфраструктуры (КИИ).
* Определяет обязанности субъектов КИИ по обеспечению безопасности своих информационных систем.
* Создает правовые и организационные основы для противодействия компьютерным инцидентам.
  + 1. Федеральный закон «О внедрения технологий искусственного интеллекта» (123-ФЗ)
* регламентирует условия для разработки и внедрения технологий искусственного интеллекта, а также возможность последующего использования результатов его применения.
  + 1. Федеральный закон «О безопасности» (ФЗ-390):
* Обозначает основные направления государственной политики в сфере безопасности.
* Включает аспекты информационной безопасности как составной части национальной безопасности.
  + 1. Указ Президента РФ от 17.03.2008 №351 «О мерах по обеспечению ИБ РФ при использовании информационно-телекоммуникационных сетей международного информационного обмена»;
    2. Указ Президента РФ от 05.12.2016 №646 «Об утверждении Доктрины ИБ РФ»;
* Доктрина информационной безопасности — это система официальных взглядов и принципов, определяющих основные направления обеспечения национальной безопасности Российской Федерации в информационной сфере.
* Доктрина служит основой для формирования государственной политики и развития общественных отношений в области использования информационных технологий, а также для выработки мер по совершенствованию системы обеспечения информационной безопасности.
  + 1. Указ Президента РФ от 09.05.2017 №203 «О стратегии развития информационного общества в РФ на 2017-2030 годы»
    2. Указ Президента РФ от 01.05.2022 №250 «О дополнительных мерах по обеспечению ИБ РФ»
    3. ГОСТ Р 57580 и ГОСТ Р ИСО/МЭК 27001:
* Регламентируют стандарты управления информационной безопасностью.
* Описывают требования по оценке рисков ИБ и внедрению мер защиты.
  + 1. Приказы ФСТЭК:
* Приказ № 239 от 25 декабря 2017 года «Об утверждении требований по обеспечению безопасности значимых объектов критической информационной инфраструктуры Российской Федерации».
* Приказ № 17 от 2 февраля 2021 года «Об утверждении Требований к созданию систем безопасности персональных данных в информационных системах».
* Приказ № 246 от 29 сентября 2022 года «О внесении изменений в Требования по обеспечению целостности, устойчивости функционирования и безопасности информационных систем и информационно-телекоммуникационных сетей».

# Сетевая безопасность

Сетевая безопасность — это комплекс мер и технологий, направленных на защиту компьютерных сетей от несанкционированного доступа, вредоносных программ и других угроз. Она включает в себя защиту данных, передаваемых по сети, а также обеспечение безопасности сетевых устройств и инфраструктуры.

Существуют множество угроз сетевой безопасности, такие как:

* Вредоносное ПО: это любая программа, созданная со злым умыслом или злыми намерениями. Такое ПО предназначено для нанесения ущерба компьютерам, серверам и другим устройствам, а также кражи личных данных пользователей.
* Фишинг и социальная инженерия: это вид кибератаки, целью которой является получение конфиденциальной информации, такой как логины, пароли и данные банковских карт. Фишинговые атаки обычно осуществляются через электронные письма, веб-сайты или текстовые сообщения, которые выглядят как официальные запросы от банков, платёжных систем или других организаций. Фишинговые электронные письма могут содержать ссылки на поддельные веб-страницы, где злоумышленники пытаются получить доступ к личным данным пользователей. Они также могут включать в себя вложения с вредоносными программами, которые автоматически запускаются при открытии файла.
* DoS-атаки (Denial of Service): это атаки на компьютерную систему или сеть, целью которых является сделать их недоступными для легитимных пользователей. DoS-атаки направлены на перегрузку или полное отключение системы, что приводит к невозможности предоставления услуг или доступа к информации. DoS-атаки могут быть направлены на различные компоненты компьютерной системы или сети, такие как серверы, маршрутизаторы, коммутаторы и другие устройства. Они могут осуществляться с помощью различных инструментов и методов, таких как скрипты, программы, боты и т. д.
* DDoS-атаки (Distributed Denial of Service): более мощные версии DoS-атак, в которых используются распределённые ресурсы для перегрузки целевой системы.
* Атаки на веб-приложения: это попытки злоумышленников получить несанкционированный доступ к информации, хранящейся на сервере, или нарушить работу веб-сайта. Существует множество видов атак на веб-приложения, которые могут быть направлены на различные компоненты системы: SQL-инъекции, межсайтовый скриптинг, небезопасные прямые ссылки на объекты, недостаточная аутентификация, выполнение кода.
* Перехват данных: прослушивание сетевого трафика с целью получения конфиденциальной информации.
* Несанкционированный доступ: попытка получить доступ к ресурсам сети без разрешения. Последствия несанкционированного доступа могут быть серьёзными, включая кражу данных, изменение или удаление информации, а также нарушение работы системы.
* Компрометация учётных данных: это ситуация, когда конфиденциальные данные (логины, пароли, номера кредитных карт и т. д.) становятся известны третьим лицам без разрешения владельца. Это может произойти в результате различных видов кибератак, таких как фишинг, социальная инженерия, взлом и другие. Компрометация учётных данных может привести к серьёзным последствиям для безопасности, включая кражу личных данных, мошенничество с кредитными картами, несанкционированный доступ к конфиденциальной информации и даже промышленный шпионаж.
* Небезопасные настройки сети: это ошибки в конфигурации сетевых устройств и программного обеспечения, которые могут быть использованы злоумышленниками для несанкционированного доступа к информации или ресурсам. К таким ошибкам относятся: открытые порты на межсетевом экране, слабые пароли на учётных записях администраторов, незаблокированные протоколы и службы, которые не используются в работе, но могут быть использованы для атак, небезопасные настройки маршрутизаторов и коммутаторов, позволяющие злоумышленникам перенаправлять трафик или получать доступ к конфиденциальной информации.
* Человеческий фактор: ошибки пользователей, приводящие к уязвимостям в системе безопасности.

Уязвимости:

1. Необновленное программное обеспечение (устаревшие версии ПО могут содержать известные уязвимости):

2. Отсутствие шифрования (передача данных в незашифрованном виде делает их легкой мишенью для перехвата)

3. Слабые пароли (Простые пароли легко могут быть взломаны с помощью методов подбора)

4. Необнаруженные бэкдоры («Черный ход» в системе, оставленный намеренно или случайно, может быть использован злоумышленниками.

Примеры:

1. DDoS атаки на российские организации:

В последние годы было несколько заметных случаев DDoS-атак (распределённых атак отказа в обслуживании) на российские банки, финансовые учреждения и государственные сайты. Эти атаки приводили к временной недоступности онлайн-сервисов.

2. Атака на «Энергомашбанка» (2016):

В октябре 2016 года был взломан сервер «Энергомашбанка». В результате атаки злоумышленники получили доступ к информации о транзакциях и личным данным клиентов.

3. Взлом «Яндекс.Такси» (2020):

В июле 2020 года была проведена атака на сервис «Яндекс.Такси». Злоумышленники использовали уязвимость для массового заказа такси в одном районе Москвы, создавая заторы и неудобства для пользователей.

4. Кибератака на Роснефть (2017):

Одна из крупнейших кибератак, известная как NotPetya, затронула в том числе и российскую нефтяную компанию «Роснефть». Вирус шифровал данные на компьютерах компании, что привело к приостановке ряда операционных процессов.

5. Атака на российские больницы (2020):

Во время пандемии COVID-19 были зафиксированы случаи кибератак на медицинские учреждения. Злоумышленники пытались получить доступ к данным пациентов и финансовым ресурсам больниц.

6. Фишинг и социальная инженерия:

В России, как и во многих других странах, широко распространены фишинговые атаки. Злоумышленники отправляют поддельные электронные письма и сообщения, пытаясь убедить жертв разгласить свои личные или финансовые данные.

7. Атака на Минфин РФ (2021):

В 2021 году Министерство финансов России сообщило о хакерской атаке, в результате которой злоумышленники попытались получить доступ к внутренним документам и переписке министерства.

# Методы защиты

Сетевые атаки становятся всё более сложными и опасными. Защита от них требует комплексного подхода, включающего различные методы и технологии. Такие как:

1. Многоуровневая защита:

* Защита периметра: *Межсетевые экраны (фаерволы)* — это устройства, которые контролируют и фильтруют сетевой трафик между внутренней сетью организации и Интернетом. Они могут быть аппаратными, программными или виртуальными. Межсетевые экраны защищают от несанкционированного доступа к внутренним ресурсам компании, а также ограничивают доступ к определённым сайтам и приложениям. *Системы обнаружения и предотвращения вторжений (IDS/IPS)* — это технологии, предназначенные для обнаружения и блокирования сетевых атак. IDS (система обнаружения вторжений) отслеживает сетевой трафик и выявляет подозрительную активность. IPS (система предотвращения вторжений) не только обнаруживает, но и автоматически блокирует атаки. Системы IDS/IPS используются для защиты от вирусов, червей, троянов, DoS-атак и других угроз..
* Внутренняя защита: Сегментация сети — это разделение компьютерной сети на отдельные сегменты или подсети с целью повышения безопасности, производительности и управляемости. Сегментация позволяет изолировать критически важные ресурсы и данные от несанкционированного доступа, а также оптимизировать трафик между сегментами. Это особенно важно для крупных организаций с большим количеством пользователей и устройств.

1. Использование ИИ и машинного обучения (МО):

* Сбор данных. Для обучения моделей ИИ необходимо собрать данные о прошлых атаках, включая информацию о методах атак, используемых инструментах и уязвимостях. Эти данные могут быть получены из различных источников, таких как журналы событий, системы обнаружения вторжений (IDS) и системы управления событиями безопасности (SIEM).
* Обучение моделей. На основе собранных данных можно обучить модели ИИ, которые будут способны обнаруживать аномалии в поведении пользователей и систем. Например, модель может научиться распознавать необычные запросы к базе данных или подозрительные сетевые соединения.
* Тестирование и оптимизация. После обучения модели необходимо провести её тестирование на новых данных, чтобы убедиться в её эффективности. Если модель не справляется с обнаружением атак, необходимо её доработать или выбрать другую модель.
* Внедрение в систему. После успешного тестирования модель ИИ может быть внедрена в систему безопасности для постоянного мониторинга и обнаружения атак.
* Интеграция с другими системами. ИИ можно интегрировать с существующими системами безопасности, такими как IDS и SIEM, для более точного обнаружения атак и повышения общей эффективности защиты.
* Мониторинг и анализ результатов. После внедрения модели ИИ важно отслеживать её работу и анализировать результаты. Это позволит выявить слабые места в системе безопасности и улучшить её защиту.
* Постоянное обновление. Технологии ИИ постоянно развиваются, поэтому необходимо следить за новыми методами и алгоритмами, чтобы обеспечить максимальную эффективность защиты.

1. Аутентификация и контроль доступа:

* Парольная аутентификация. Самый распространённый способ аутентификации, при котором пользователь вводит пароль для доступа к системе или ресурсу. Парольная аутентификация может быть усилена использованием многофакторной аутентификации (MFA), которая требует от пользователя предоставления нескольких видов доказательств своей личности, таких как пароль, токен или биометрические данные.
* Биометрическая аутентификация. Этот метод использует уникальные характеристики человека, такие как отпечатки пальцев, распознавание лица или голоса, для аутентификации. Биометрическая аутентификация обеспечивает высокий уровень безопасности, но требует специального оборудования и может вызывать опасения у некоторых пользователей относительно конфиденциальности их личных данных.
* Сертификаты и ключи. В этом методе используются цифровые сертификаты и ключи для аутентификации пользователей и устройств. Сертификаты выдаются доверенными центрами сертификации (ЦС) и содержат информацию о владельце сертификата, а также открытый ключ, который используется для шифрования данных. Ключи могут быть симметричными (один и тот же ключ для шифрования и дешифрования) или асимметричными (открытый и закрытый ключи).
* Контроль доступа на основе ролей (RBAC). Этот метод позволяет определять права доступа пользователей на основе их роли в организации. Например, администратор может иметь полный доступ ко всем ресурсам, а обычный пользователь — только ограниченный доступ к определённым функциям.
* Управление доступом на основе атрибутов (ABAC). В этом методе права доступа определяются на основе атрибутов пользователя, устройства или запроса. Атрибуты могут включать в себя роль пользователя, местоположение, время суток и другие параметры.

1. Шифрование данных:

* Симметричное шифрование. В этом методе используется один и тот же ключ для шифрования и дешифрования данных. Это самый простой и быстрый способ шифрования, но он имеет ряд недостатков, таких как необходимость безопасного хранения ключа и сложность управления ключами.
* Асимметричное шифрование (криптосистема с открытым ключом). В этом методе используются два ключа: открытый ключ, который может быть известен всем, и закрытый ключ, который известен только получателю. Открытый ключ используется для шифрования данных, а закрытый ключ — для их дешифрования. Этот метод обеспечивает более высокий уровень безопасности, но требует больше вычислительных ресурсов.
* Хеширование. Хеширование — это одностороннее преобразование данных в фиксированный размер хеш-значения. Хеш-значение используется для проверки целостности данных и не может быть использовано для восстановления исходных данных.5. Мониторинг и анализ:

1. Обучение и осведомлённость сотрудников:

* Кибербезопасность тренинги: регулярное обучение персонала по вопросам кибербезопасности и новым методам атак.
* Фишинг-тесты: симуляции фишинг-атак для повышения осведомленности сотрудников.

1. Инцидент-менеджмент:

* Планы реагирования на инциденты: разработка чёткого плана действий в случае атаки, включая коммуникационные стратегии и процедуры восстановления.
* Резервное копирование: регулярное создание резервных копий данных для снижения последствий потенциального ущерба.

# Вывод по первой главе:

В современном цифровом мире сетевые атаки представляют собой серьезную угрозу для безопасности информации и стабильности функционирования информационных систем. Поэтому важно использовать различные методы защиты, чтобы эффективно противостоять этим атакам.

# ЗАЩИТА ИНФОРМАЦИОННЫХ СИСТЕМ ОТ СЕТЕВЫХ УГРОЗ

# 2.1. Использование систем ИИ в ИБ

Искусственный интеллект (ИИ) – эффективное решение неформализованных задач.

История создания искусственного интеллекта

Первые попытки создания систем, которые могли бы имитировать интеллектуальные способности человека, относятся к середине XX века. В 1950 году британский математик Алан Тьюринг опубликовал статью «Вычислительные машины и интеллект», в которой предложил тест для определения способности машины мыслить как человек. Этот тест впоследствии получил название «тест Тьюринга».

В 1956 году на семинаре в Дартмутском университете был предложен термин «искусственный интеллект» (ИИ). Семинар собрал группу учёных, среди которых были Марвин Мински, Джон Маккарти, Клод Шеннон и другие. Они определили искусственный интеллект как область исследований, которая занимается разработкой компьютерных программ, способных выполнять задачи, требующие интеллектуальных усилий человека.

Ранние системы искусственного интеллекта были основаны на правилах и использовали логические рассуждения для решения задач. Эти системы были ограничены в своих возможностях и не могли адаптироваться к новым ситуациям.

В конце 1960-х годов появились первые экспертные системы, которые использовали знания экспертов для принятия решений. Экспертные системы были более успешными, чем системы, основанные на правилах, но они также имели ограничения. Они были эффективны только в тех областях, где было достаточно знаний экспертов.

С развитием вычислительной техники и появлением новых алгоритмов и методов машинного обучения, искусственный интеллект стал более мощным и универсальным. В настоящее время искусственный интеллект используется в различных областях, таких как медицина, финансы, транспорт, производство и других.

Основные этапы развития искусственного интеллекта:

* 1943–1955: создание первых вычислительных машин.
* 1956: появление термина «искусственный интеллект».
* Конец 1950-х — начало 1970-х: разработка первых систем искусственного интеллекта.
* Середина 1980-х: появление экспертных систем.
* Начало 2000-х: развитие машинного обучения и нейронных сетей.
* Настоящее время: использование искусственного интеллекта в различных областях.

Развитие искусственного интеллекта продолжается, и его возможности постоянно расширяются. Использование систем ИИ в информационной безопасности (ИБ) становится все более важным и распространенным. Основные области применения ИИ в ИБ включают:

1. Обнаружение угроз: Системы ИИ могут анализировать огромные объемы данных и выявлять аномалии, которые могут свидетельствовать о возникновении угрозы. Алгоритмы машинного обучения и глубокого обучения помогают различать нормальное поведение и подозрительные действия в сети.
2. Ответ на инциденты: ИИ-системы могут автоматически реагировать на определенные угрозы, не дожидаясь вмешательства человека. Например, они могут блокировать подозрительные IP-адреса или изолировать зараженные устройства.
3. Анализ вредоносного ПО: ИИ может помогать в анализе нового вредоносного ПО, определять его поведение и предлагать меры для его нейтрализации.
4. Предотвращение атак: ИИ может предсказывать возможные атаки, анализируя данные о предыдущих инцидентах и текущей активности в сети.
5. Соответствие стандартам и требованиям: ИИ может помочь в автоматизации процессов соответствия требованиям безопасности, таких как соблюдение регуляторных норм и внутренних политик безопасности.

Ключевые преимущества использования ИИ в сфере ИБ включают:

* Скорость и эффективность: Системы ИИ способны обрабатывать и анализировать большие объемы данных в реальном времени, что значительно быстрее и эффективнее, чем ручной анализ.
* Автоматизация: Многие рутинные задачи, такие как мониторинг сетевого трафика и обновление политик безопасности, могут быть автоматизированы с помощью ИИ, что освобождает человеческие ресурсы для более сложных задач.
* Адаптивность: ИИ-системы могут обучаться и адаптироваться к новым типам угроз, что делает их более устойчивыми к изменяющимся условиям и атакам.

Однако, несмотря на многочисленные преимущества, есть и недостатки:

* Ложные срабатывания: Некоторые ИИ-системы могут генерировать большое количество ложных тревог, что может перегружать команды ИБ и снижать доверие к системе.
* Зависимость от данных: Качество работы ИИ сильно зависит от качества и количества данных, используемых для обучения.
* Этичные и правовые вопросы: Использование ИИ может вызвать вопросы о конфиденциальности данных и соблюдении прав пользователя.

Интеграция ИИ в системы информационной безопасности продолжает развиваться, предлагая как значительные преимущества, так и новые проблемы, которые необходимо решать.

# 2.2. Инструменты распознавания сетевых атак

Инструменты распознавания сетевых атак – это специализированные программные и аппаратные решения, предназначенные для обнаружения и анализа подозрительной активности в сети. Они помогают предотвратить несанкционированный доступ к информации, нарушение работы системы и другие угрозы безопасности.

Существует множество инструментов для распознавания и предотвращения сетевых атак. Эти инструменты можно разделить на следующие категории:

1. Системы обнаружения вторжений (СОВ)(IDS) и предотвращения вторжений (IPS). СОВ — это программные или программно-аппаратные средства, предназначенные для выявления фактов неавторизованного доступа в компьютерную систему или сеть. СОВ анализируют сетевой трафик и поведение пользователей на предмет подозрительной активности, которая может свидетельствовать о попытке вторжения. Системы обнаружения вторжений могут быть реализованы как сетевые или хостовые. Сетевые СОВ анализируют трафик, проходящий через определённую точку сети, например, маршрутизатор или коммутатор. Они могут обнаруживать такие виды атак, как сканирование портов, попытки несанкционированного доступа к ресурсам и распространение вредоносного кода. Хостовые СОВ устанавливаются на отдельных компьютерах и отслеживают события, происходящие на этих компьютерах. Хостовые СОВ могут обнаруживать атаки, направленные на конкретный компьютер, такие как попытки несанкционированного изменения файлов или реестра, а также использование уязвимостей в программном обеспечении. Система предотвращения вторжений (IPS) — это технология сетевой безопасности, предназначенная для обнаружения и блокирования несанкционированных действий в компьютерной сети. Системы IPS работают в режиме реального времени, анализируя сетевой трафик на предмет подозрительной активности. Они могут автоматически блокировать атаки, такие как DDoS-атаки, попытки несанкционированного доступа к ресурсам или распространение вредоносного кода. Примеры таких систем:

* Suricata: это система IDS/IPS с открытым исходным кодом, которая поддерживает анализ сетевых протоколов, а также обнаружение аномалий.
* Snort: один из самых популярных IDS с открытым исходным кодом, широко используемый во всем мире.
* Stitch: Российская система IDS/IPS для защиты корпоративных сетей.

1. Межсетевые экраны (Firewall). МЭ — это устройство или программа, которые контролируют и фильтруют сетевой трафик между локальной сетью и Интернетом. МЭ защищает сеть от несанкционированного доступа, вредоносных программ и других угроз безопасности. Межсетевые экраны могут быть реализованы как аппаратные устройства, программные приложения или и то, и другое вместе. Аппаратные межсетевые экраны обычно используются в корпоративных сетях, где требуется высокая производительность и надёжность. Программные межсетевые экраны часто используются в домашних и небольших офисных сетях. Правильно настроенный межсетевой экран может стать эффективным средством защиты вашей сети от различных угроз безопасности. Примеры межсетевых экранов:

* Kaspersky Endpoint Security: программное обеспечение от «Лаборатории Касперского», включающее в себя функции межсетевого экрана.
* Fortinet FortiGate: многофункциональный межсетевой экран (NGFW), используемый для защиты сетей среднего и крупного бизнеса.
* Positive Technologies PT Firewall: разработка Positive Technologies для защиты сетевых границ.

1. Анализ сетевого трафика и мониторинг:

* PT NAD (Network Attack Discovery): Российское решение для анализа сетевого трафика и детектирования атак.
* SolarWinds Network Performance Monitor: инструмент для мониторинга и анализа сетевого трафика.
* Zabbix: система мониторинга с открытым исходным кодом, используемая для контроля сетевых и серверных ресурсов.

1. Антивирусные решения и средства защиты конечных точек. Антивирусы — это программное обеспечение, предназначенное для обнаружения, предотвращения и удаления вредоносных программ с компьютеров, серверов и других устройств. Средства защиты конечных точек могут включать в себя функции шифрования данных, контроля доступа, управления устройствами и другие меры безопасности. Они помогают предотвратить несанкционированный доступ к информации, кражу данных и другие киберугрозы. Примеры:

* Kaspersky Endpoint Security: Комплексное решение для защиты от вирусов и других угроз.
* Dr.Web Enterprise Security Suite: Российская антивирусная программа, которая обеспечивает защиту всех узлов сети.

1. Системы управления событиями и информацией безопасности (SIEM). SIEM — это класс решений для обеспечения информационной безопасности, которые предназначены для сбора, анализа и корреляции событий безопасности в режиме реального времени:

* Solar JSOC: Российская платформа класса SIEM для сбора, анализа и корреляции событий безопасности.
* MaxPatrol SIEM: Разработка компании Positive Technologies для мониторинга сетевой безопасности.

Основной задачей этих инструментов является обнаружение и предотвращение сетевых атак, таких как DDoS атаки, фишинг, SQL-инъекции и другие. Устройства и приложения способны интегрироваться в общую систему корпоративной сети и обеспечивать высокий уровень защиты информации. Большинство из них поддерживает русскоязычный интерфейс и имеет локализацию, что облегчает их использование для отечественных специалистов.

# 2.3. Архитектура и особенности нейронных сетей

Нейронные сети — это вычислительные модели, вдохновленные биологическими нервными системами. Основой нейронных сетей являются искусственные нейроны, которые имитируют функции биологических нейронов. Рассмотрим основные аспекты их архитектуры и особенностей:

История создания нейронных сетей:

Идея создания искусственных нейронных сетей (ИНС) возникла в 1940-х годах, когда учёные начали исследовать принципы работы мозга и нервной системы. Первые попытки создания ИНС были предприняты в середине XX века, но они не были успешными из-за отсутствия необходимых вычислительных мощностей. В 1958 году американский нейрофизиолог Фрэнк Розенблатт разработал первую модель нейронной сети, которая получила название «персептрон». Персептрон был способен распознавать простые образы, такие как буквы или цифры. Однако персептрон имел ряд ограничений, которые не позволяли ему решать более сложные задачи. В конце 1960-х годов интерес к нейронным сетям снизился из-за их ограниченной способности к обучению и решению сложных задач. Однако в начале 1980-х годов произошёл новый всплеск интереса к нейронным сетям благодаря развитию новых алгоритмов обучения и появлению более мощных компьютеров.

В настоящее время нейронные сети являются одним из самых перспективных направлений искусственного интеллекта. Они используются для решения широкого спектра задач, таких как распознавание образов, обработка естественного языка, прогнозирование временных рядов и других.

Развитие нейронных сетей продолжается, и их возможности постоянно расширяются. Нейронные сети становятся всё более важным инструментом для решения сложных задач и повышения эффективности работы.

1. Структура слоев: Рисунок 1.

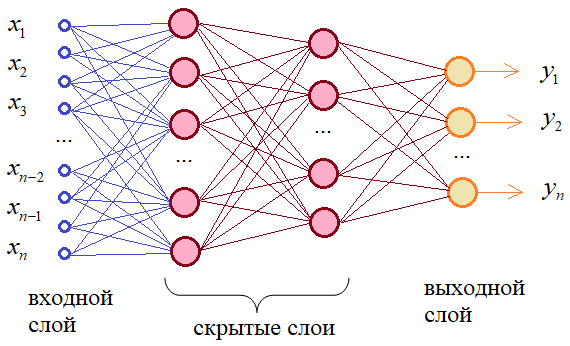


Рисунок 1 - Структура слоев нейронной сети

* Входной слой: принимает данные для обработки, например, изображения, текст или числовые данные.
* Скрытые слои: представляют собой промежуточные уровни нейронной сети, которые выполняют обработку входных данных. Количество скрытых слоёв и количество нейронов в каждом слое также зависит от задачи и алгоритма.
* Выходной слой: Выдает окончательный результат, в зависимости от задачи (классы, регрессия или другие виды прогнозов).
* Веса и смещения: представляют собой параметры нейронной сети, которые определяют силу связи между нейронами. Веса определяют, насколько сильно сигнал от одного нейрона влияет на другой нейрон, а смещения определяют смещение выходного сигнала нейрона.
* Функция активации определяет, как выходной сигнал нейрона преобразуется перед передачей на следующий слой.

1. Существует множество различных архитектур нейронных сетей, каждая из которых имеет свои особенности и предназначена для решения определённых задач. Вот некоторые из наиболее распространённых архитектур:

* Однослойные перцептроны (SMLP) — это простейшие нейронные сети, состоящие из одного слоя нейронов. Они могут использоваться для решения простых задач, таких как линейная регрессия или классификация двух классов.
* Многослойные перцептроны (MLP) — это более сложные нейронные сети, состоящие из нескольких слоёв нейронов. MLP могут решать более сложные задачи, такие как классификация изображений или прогнозирование временных рядов.
* Свёрточные нейронные сети (CNN) — это тип нейронных сетей, который используется для обработки изображений. CNN состоят из свёрточных слоёв, которые выполняют операцию свёртки над входными данными. Свёрточные слои позволяют CNN извлекать признаки из изображений, такие как края, углы и формы.
* Рекуррентные нейронные сети (RNN) — это тип нейронных сетей, который может обрабатывать последовательности данных, такие как текст, речь или временные ряды. RNN имеют обратную связь, которая позволяет им запоминать информацию о предыдущих элементах последовательности. Это делает RNN полезными для задач, где важна контекстная информация, таких как машинный перевод или генерация текста.
* Генеративно-состязательные сети (GAN) — это тип нейронных сетей, состоящий из двух подсетей: генератора и дискриминатора. Генератор создаёт новые данные, а дискриминатор пытается отличить эти данные от реальных данных. GAN используются для генерации реалистичных изображений, видео и звука.
* Автоэнкодеры (AE) — это тип нейронной сети, который состоит из кодировщика и декодировщика. Кодировщик сжимает входные данные в низкоразмерное представление, а декодировщик восстанавливает исходные данные из этого представления. AE используются для сжатия данных, извлечения признаков и обнаружения аномалий.
* Трансформеры — это относительно новый тип нейронных сетей, основанный на механизме внимания. Трансформеры используются для обработки последовательностей данных, таких как текст и речь. Они особенно эффективны для задач машинного перевода и генерации - Долгая-краткосрочная память (Long Short-Term Memory, LSTM) и GRU: Усовершенствованные версии RNN, решают проблему затухания градиента.

1. Функции активации в нейронных сетях — это математические функции, которые определяют выходной сигнал нейрона в зависимости от суммы взвешенных входных сигналов. Они играют ключевую роль в процессе обучения и функционирования нейронной сети. Функции активации представлены на Рисунке 2.

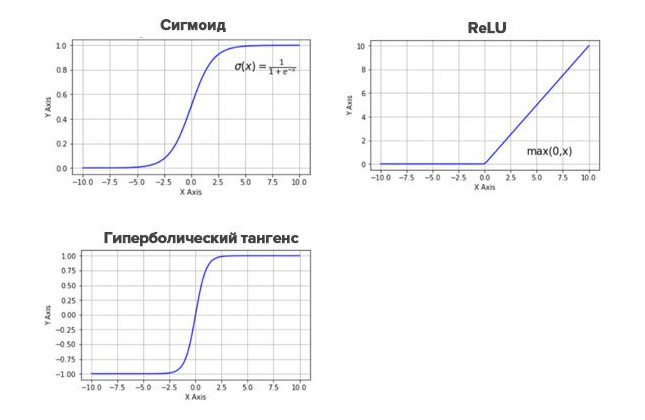


Рисунок 2 - Функции активации нейронных сетей

- Sigmoid и TanH: Применяются для нелинейных преобразований, сглаживают выход.

- ReLU (Rectified Linear Unit): очень популярна из-за своей простоты и эффективности.

- Softmax: Используется для многоклассовой классификации.

4. Алгоритмы обучения нейронных сетей — это методы, которые позволяют настроить параметры модели таким образом, чтобы она могла эффективно решать поставленные задачи. Существует несколько основных алгоритмов обучения:

* Метод обратного распространения ошибки (Backpropagation) — это наиболее распространённый алгоритм обучения многослойных нейронных сетей. Он основан на градиентном спуске и позволяет оптимизировать веса и смещения для минимизации ошибки между предсказанными и реальными значениями.
* Стохастический градиентный спуск (Stochastic Gradient Descent) — это вариация метода обратного распространения, которая использует случайные выборки данных для обновления весов и смещений. Это позволяет ускорить процесс обучения и избежать застревания в локальных минимумах.
* Метод моментов (Momentum) — этот метод добавляет инерцию к процессу обучения, что позволяет модели быстрее адаптироваться к изменениям в данных. Инерция определяется коэффициентом момента, который определяет, насколько сильно модель будет учитывать предыдущие направления движения.
* Адаптивный градиентный спуск (Adaptive Gradient Descent) — этот алгоритм использует различные скорости обучения для разных параметров модели. Это позволяет более точно настроить веса и смещения и ускорить процесс обучения.
* Adam (Adaptive Moment Estimation) — это адаптивный алгоритм, который сочетает в себе идеи стохастического градиентного спуска и метода моментов. Adam автоматически регулирует скорость обучения для каждого параметра модели, что делает его эффективным для широкого спектра задач.
* RMSProp (Root Mean Square Propagation) — этот алгоритм также адаптирует скорость обучения для каждого параметра, но использует среднеквадратичное отклонение вместо среднего значения градиента. RMSProp может быть более эффективным при наличии больших разбросов в значениях градиентов.
* AdaGrad (Adaptive Gradient) — этот алгоритм адаптирует скорость обучения на основе истории градиентов для каждого параметра. AdaGrad может помочь избежать проблемы затухания градиентов, когда некоторые параметры становятся слишком большими или маленькими.

5. Особенности и проблемы нейронных сетей:

* Одна из основных проблем использования нейронных сетей заключается в том, что они представляют собой «чёрный ящик», и их результаты трудно интерпретировать. Это может затруднить понимание того, как именно сеть пришла к определённому решению, и может вызвать недоверие к её результатам.
* Для того чтобы нейронная сеть могла эффективно решать задачи, ей необходимо большое количество обучающих данных. Однако сбор и подготовка таких данных могут быть дорогостоящими и трудоёмкими процессами.
* Если нейронная сеть слишком хорошо адаптируется к обучающим данным, она может начать запоминать их вместо того, чтобы обобщать. Это называется переобучением, и оно может привести к тому, что сеть будет плохо работать на новых данных.
* Нейронные сети принимают решения на основе сложных математических вычислений, которые трудно понять человеку. Это может вызывать опасения по поводу того, насколько эти решения справедливы и предсказуемы.
* Качество входных данных имеет решающее значение для производительности нейронной сети. Если данные содержат ошибки или предвзятость, это может повлиять на результаты работы сети.
* Нейронные сети имеют множество параметров, которые необходимо настроить для достижения оптимальной производительности. Этот процесс может быть сложным и требовать глубоких знаний в области машинного обучения.
* Использование нейронных сетей может вызывать этические вопросы, связанные с конфиденциальностью данных, предвзятостью и ответственностью за результаты.
* Обучение и использование нейронных сетей требует значительных вычислительных ресурсов, таких как мощные процессоры и графические процессоры. Это может ограничивать их доступность для небольших организаций или проектов.

Нейронные сети применяются в различных областях: от распознавания речи и изображений до медицинских диагнозов и автономных систем. Постоянное развитие архитектур и методов обучения ускоряет их повсеместное внедрение.

# 2.4. Машинное обучение

Машинное обучение — это область искусственного интеллекта, которая позволяет компьютерам учиться и адаптироваться на основе данных без явного программирования. История машинного обучения началась в 1950-х годах с появлением первых компьютеров и попыток создать программы, способные к обучению. Первые исследования в области машинного обучения были связаны с разработкой алгоритмов для решения конкретных задач, таких как распознавание образов и прогнозирование временных рядов. В 1960-е годы появились первые алгоритмы машинного обучения, такие как перцептрон Розенблатта и метод опорных векторов (SVM). Эти алгоритмы были основаны на статистических методах и теории информации. В 1970-е и 1980-е годы машинное обучение переживало застой из-за отсутствия доступных данных и вычислительных ресурсов. Однако в конце 1980-х годов с развитием персональных компьютеров и появлением больших объёмов данных начался новый этап развития машинного обучения. Были разработаны новые алгоритмы, такие как нейронные сети и деревья решений, которые стали основой для современных методов машинного обучения. В 2000-е годы с развитием интернета и социальных сетей появились новые источники данных, что привело к быстрому росту машинного обучения. Появились новые методы, такие как глубокое обучение и обучение с подкреплением, которые позволили машинам решать более сложные задачи, такие как распознавание речи и изображений.

Сегодня машинное обучение является одной из самых быстрорастущих областей искусственного интеллекта. Оно используется во многих областях, таких как медицина, финансы, транспорт и производство. Машинное обучение продолжает развиваться, и мы можем ожидать появления новых методов и приложений в будущем.

В сфере ИБ машинное обучение (МО) играет все более важную роль. Вот несколько ключевых направлений, где МО активно применяется:

1. Обнаружение аномалий: Используются алгоритмы, которые анализируют нормальное поведение систем и пользователей. Когда появляется аномалия, система может быстрее обнаружить потенциальную угрозу, например, взлом.

2. Обнаружение вредоносного ПО: Машинное обучение позволяет создавать более эффективные средства обнаружения вредоносного ПО, изучая большие объемы данных о предыдущих атаках.

3. Фильтрация спама и фишинга: Алгоритмы МО могут анализировать структуру и содержание сообщений, чтобы распознать и блокировать спам и фишинговые атаки.

4. Анализ сетевого трафика: МО может следить за потоками данных в сети, обнаруживая подозрительные активности, характерные для DDoS-атак или других видов вторжений.

5. Аутентификация и биометрия: Использование МО для анализа биометрических данных (отпечатки пальцев, голос, лицо) позволяет повысить надежность систем аутентификации.

6. Прогнозирование атак: МО может анализировать тенденции и предсказывать потенциальные угрозы, основываясь на исторических данных и текущих аномалиях.

Методы машинного обучения:

* Обучение с учителем (supervised learning) — метод, при котором модель обучается на размеченных данных, где каждому примеру соответствует правильный ответ. Модель пытается найти закономерности в данных и использовать их для предсказания ответов на новые примеры. (Метод опорных векторов, решающие деревья и случайный лес, нейронные сети)
* Обучение без учителя (unsupervised learning) — метод, при котором модель учится находить структуры и закономерности в неразмеченных данных. Модель не имеет правильных ответов, и она должна самостоятельно определить, какие группы или кластеры существуют в данных. (кластеризация, понижение размерности, ассоциативный анализ)
* Обучение с подкреплением (reinforcement learning) — метод, основанный на взаимодействии агента с окружающей средой. Агент получает вознаграждение за правильные действия и наказание за неправильные. Цель агента — максимизировать общее вознаграждение. Игра в шахматы, управление роботом и т. д.
* Ансамблевые методы — методы, которые объединяют несколько моделей для повышения точности и надёжности. Ансамбли могут включать модели разных типов или одну и ту же модель, обученную на разных подмножествах данных. (бэггинг, бустинг и стекинг.)

Преимущества использования МО в ИБ:

- Скорость и масштаб: МО может обрабатывать большие объемы данных быстрее, чем человек.

- Адаптивность: способность адаптироваться к новым угрозам и не только реагировать на известные атаки.

- Снижение ложных срабатываний: с течением времени алгоритмы могут учиться уменьшать количество ложных тревог.

Проблемы

- Обучение на данных: требуются большие объемы качественных данных для обучения моделей.

- Сложность и стоимость: разработка и внедрение МО в ИБ могут быть дорогостоящими.

- Уязвимость самих моделей: очистка данных и безопасность самих алгоритмов важны, так как они также могут быть целями для атак.

Примеры популярных технологий

* Обнаружение и предотвращение вторжений (Intrusion Detection and Prevention Systems, IDS/IPS) системы обнаружения и предотвращения вторжений используют алгоритмы машинного обучения для анализа сетевого трафика и выявления аномалий в поведении пользователей и устройств. Это позволяет обнаруживать и блокировать несанкционированный доступ к сети, а также предотвращать распространение вредоносного ПО.
* Анализ поведения пользователей (User Behavior Analytics, UBA) анализ поведения пользователей использует алгоритмы машинного обучения для мониторинга действий пользователей в сети и выявления подозрительного поведения. Например, система может обнаружить аномальное использование ресурсов, необычные запросы к базам данных или попытки доступа к конфиденциальной информации.
* Защита от фишинга и мошенничества (Anti-Phishing and Fraud Detection) алгоритмы машинного обучения используются для анализа электронных писем и веб-страниц на предмет признаков фишинговых атак и мошеннических схем. Система может автоматически блокировать подозрительные сообщения и предупреждать пользователей о возможных рисках.
* Распознавание образов и обнаружение аномалий (Image Recognition and Anomaly Detection) машинное обучение используется для распознавания образов на изображениях и видеозаписях, что позволяет обнаруживать аномалии, такие как несанкционированное проникновение в охраняемую зону или изменение содержимого файлов.
* Автоматическое обновление и защита программного обеспечения (Software Update and Protection) с помощью машинного обучения можно автоматически обновлять программное обеспечение на устройствах с целью устранения уязвимостей и защиты от кибератак. Алгоритмы могут анализировать информацию о безопасности и определять, какие обновления необходимы.
* Предотвращение утечки данных (Data Leakage Prevention) технологии машинного обучения позволяют обнаруживать и предотвращать утечку данных из корпоративных систем. Алгоритмы анализируют сетевой трафик, чтобы выявить попытки несанкционированного доступа к конфиденциальным данным.
* Анализ сетевого трафика (Network Traffic Analysis) используя машинное обучение, можно анализировать сетевой трафик для выявления аномалий и подозрительной активности. Это помогает предотвратить несанкционированный доступ и кибератаки.

Использование машинного обучения в ИБ становится неотъемлемой частью современного подхода к защите данных и систем, предлагая адаптивные и эффективные методы борьбы с киберугрозами.

# 2.5. Угрозы системам с использованием ИИ

Системы с использованием ИИ могут представлять различные угрозы, которые необходимо учитывать при их разработке и внедрении. Вот некоторые из основных угроз:

1. Кибератаки:

- Мошеннические алгоритмы: Хакеры могут взломать и изменить алгоритмы ИИ, что приведет к неправильным или опасным действиям системы.

- Фишинговые атаки: ИИ может использоваться для создания убедительных фишинговых сообщений, сложно отличимых от настоящих.

1. Этические и социальные вопросы:

- Предвзятость данных: ИИ может унаследовать предвзятость данных, на которых он обучался, что ведет к дискриминационным решениям.

- Отсутствие прозрачности: Комплексные алгоритмы «черного ящика» затрудняют понимание и контроль решений ИИ.

1. Ошибки и неисправности:

- Ложные результаты: ИИ может делать ошибки, особенно если условия изменяются от тех, на которых он обучался.

- Зависимость от ИИ: Сильная зависимость от систем ИИ может привести к катастрофическим последствиям в случае их отказа.

1. Угроза приватности:

- Сбор данных: ИИ-системы часто требуют больших объемов данных, что может приводить к утечкам личной информации.

- Анализ поведения: ИИ может анализировать и предсказывать поведение людей, что вызывает опасения по поводу приватности.

1. Экономические последствия:

- Угроза рабочим местам: Автоматизация процессов с помощью ИИ может привести к потере рабочих мест в некоторых отраслях.

- Монополизация: Компании, лидирующие в развитии ИИ, получают конкурентные преимущества, что может усилить экономическое неравенство.

Чтобы уменьшить риски, необходимо:

- Разрабатывать и внедрять строгие стандарты безопасности и этики.

- Проводить регулярный аудит и мониторинг систем ИИ.

- Обеспечивать прозрачность алгоритмов и процессов принятия решений.

- Обучать персонал по вопросам работы с ИИ и потенциальных рисков.

Таким образом, адекватное управление и регулирование использования ИИ позволяют минимизировать потенциальные угрозы и использовать технологии во благо общества.

# Вывод по второй главе:

Были рассмотрены современные подходы к использованию методов искусственного интеллекта в защите от сетевых атак. Основное внимание уделено способности ИИ адаптироваться и эффективно реагировать на новые типы угроз, что является значительным преимуществом по сравнению с традиционными методами кибербезопасности. Методы машинного обучения благодаря своим обучающим механизмам позволяют выявлять сложные и скрытые паттерны сетевого трафика, что существенно повышает точность обнаружения сетевых атак.

В заключение можно сказать, что интеграция методов искусственного интеллекта в системы кибербезопасности предоставляет значительные преимущества в защите от сетевых атак. Тем не менее, для полного раскрытия их потенциала необходимо учитывать ряд технических и операционных аспектов, требующих тщательной проработки и постоянного контроля.

# ПРИМЕНЕНИЕ МЕТОДА МАШИННОГО ОБУЧЕНИЯ ДЛЯ КЛАССИФИКАЦИИ СЕТЕВЫХ АТАК

# 3.1. Метод машинного обучения «случайный лес»: основные принципы и применение

В 1995 году Лео Брейман предложил идею использования ансамбля деревьев решений для повышения точности прогнозирования. Он разработал алгоритм случайного леса, который объединял результаты нескольких деревьев решений. В начале 2000-х годов исследователи начали активно использовать этот метод для решения различных задач машинного обучения. Они обнаружили, что случайный лес может быть эффективным инструментом для классификации, регрессии и других задач. С развитием вычислительных мощностей и появлением новых алгоритмов, случайный лес стал ещё более популярным. Сегодня он используется во многих областях, таких как медицина, финансы, маркетинг и другие. Случайный лес является одним из самых популярных методов машинного обучения благодаря своей универсальности и эффективности. Он продолжает развиваться и адаптироваться к новым задачам и условиям.

Суть метода случайного леса

1. Создание множества деревьев:

- Генерируются несколько независимых деревьев решений из подвыборок исходных данных.

- Каждое дерево строится на базе случайно выбранных подмножеств признаков и выборок.

2. Обучение деревьев:

- Каждое дерево обучается на своей подвыборке данных.

- Во время построения узлов деревьев также производится выборка случайного подмножества признаков.

3. Агрегация результатов:

- Для задачи классификации итоговое решение принимается голосованием деревьев.

- Для задачи регрессии результатом является среднее арифметическое предсказаний деревьев.

Демонстрация примера использования метода – случайный лес:

Рассмотрим датасет с числами: пять нулей и две единицы (Рисунок 3). Требуется разбить классы по признакам. В данном случае признаками являются цвет (синий и красный) и знак подчеркивания. Теперь можно разбить исходный датасет сначала по цвету, а затем по подчеркиванию. Причем делается это в виде логического сравнения. Стоит заметить, что следующее разбиение происходит из предыдущего.

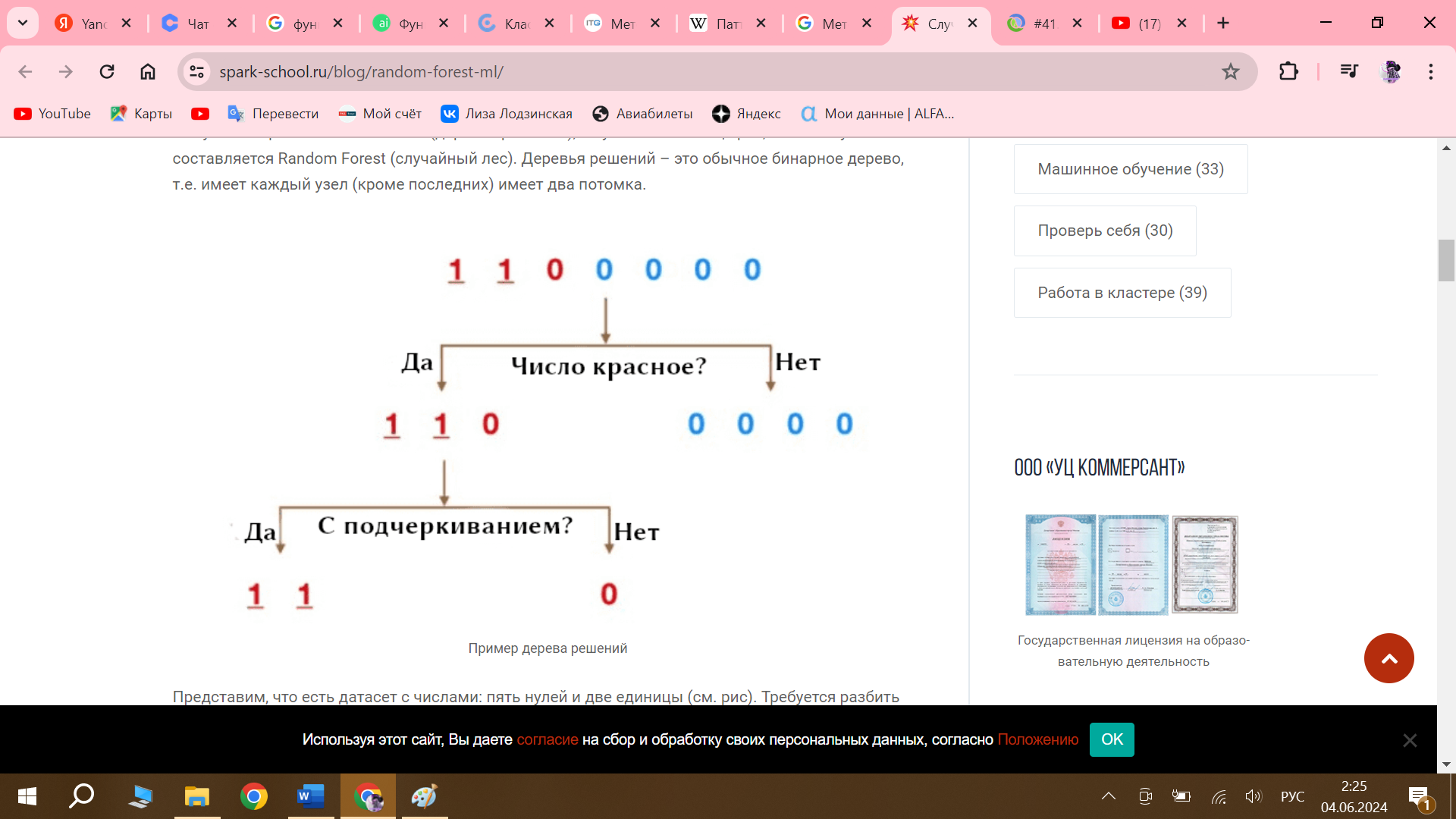


Рисунок 3 – пример дерева решений

Смоделируем случайный лес из множества различных деревьев решений. Каждое дерево предсказывает значения класса на основании своего разбиения, и выбирается то предсказание, которое получило наибольшее количество голосов (Рисунок 4).

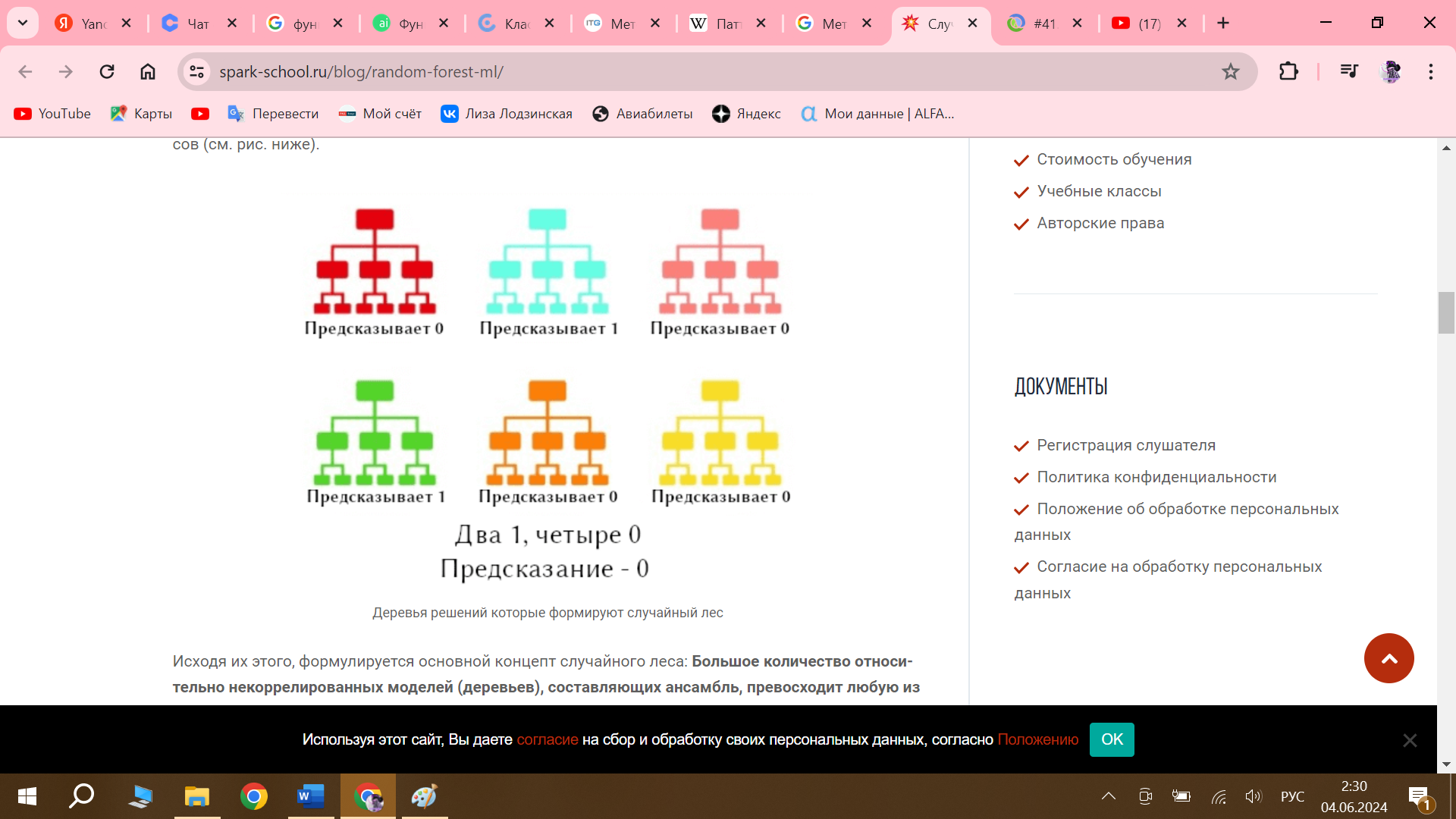


Рисунок 4 - деревья решений, которые формируют случайный лес

Исходя их этого, на выходе достигаем результата, который набрал большее количество голосов. (предсказание – 0)

Преимущества метода смешанного леса

1. Устойчивость к переобучению:

- За счет использования множества деревьев и случайных подвыборок, модель менее склонна к переобучению по сравнению с отдельным деревом решений.

2. Высокая точность:

- Комбинирование множества деревьев позволяет снизить общее количество ошибок.

3. Обработка пропущенных данных:

- Смешанный лес легко справляется с пропущенными значениями в датасете.

4. Малая необходимость в настройке гиперпараметров:

- Обычно достаточно настроить количество деревьев и глубину деревьев, в то время как остальные параметры могут использовать значения по умолчанию.

Недостатки метода смешанного леса

1. Высокие вычислительные затраты:

- Создание множества деревьев требует значительных вычислительных ресурсов, особенно при большой глубине деревьев и больших объемах данных.

2. Сложность интерпретации:

- Несмотря на высокую точность, полученная модель сложна для интерпретации из-за большого количества используемых деревьев.

Метод смешанного леса зарекомендовал себя как мощное средство для построения предсказательных моделей, особенно в случаях, когда точность важнее интерпретируемости.

# 3.2. Существующие аналоги случайного леса

Существует множество аналогичных случайному лесу методов. Вот некоторые из них:

1. Boosting:

- Улучшение моделей через последовательное обучение слабых моделей, каждая из которых исправляет ошибки предыдущей. Например (AdaBoost, Gradient Boosting Machines (GBM), XGBoost).

1. Extra Trees (Extremely Randomized Trees):

- Похож на случайный лес, но еще более случайный. Множество деревьев обучаются на различных подвыборках данных, при этом точки разделения выбираются случайно.

1. Stacking (Stacked Generalization):

- Комбинирование предсказаний нескольких моделей, например (логистической регрессии, деревьев решений и нейронных сетей) с использованием мета-модели, которая обучается на этих предсказаниях для получения финального результата.

1. Voting Classifier/Regressor:

- Комбинирование предсказаний нескольких базовых моделей при помощи голосования (majority vote) в классификации или усреднения в регрессии.

1. Gradient Boosted Decision Trees (GBDT):

- Метод, аналогичный бустингу, с применением градиентного спуска. Один из наиболее мощных методов, помогает создать сильно одну модель из многих слабых моделей. XGBoost, LightGBM и CatBoost - популярные реализации.

1. Heterogeneous Ensemble:

- Комбинирование моделей разного типа (например, решающие деревья, логистическая регрессия, SVM) для повышения производительности. Обычная техника - использование мета-модели для объединения предсказаний.

1. Bagging of XGBoost:

- Комбинирование алгоритмов бустинга с бутстрепированием для повышения устойчивости модели.

Все эти методы направлены на повышение точности и устойчивости моделей машинного обучения путем комбинирования предсказаний нескольких моделей. Выбор метода зависит от особенностей данных и задачи.

# 3.3. Алгоритм и математическое описание выбранного метода

Для задачи классификации сетевых атак был выбран метод случайного леса, поскольку он обладает высокой точностью, что крайне важно для нашего решения. Алгоритм метода представлен на рисунке 5.

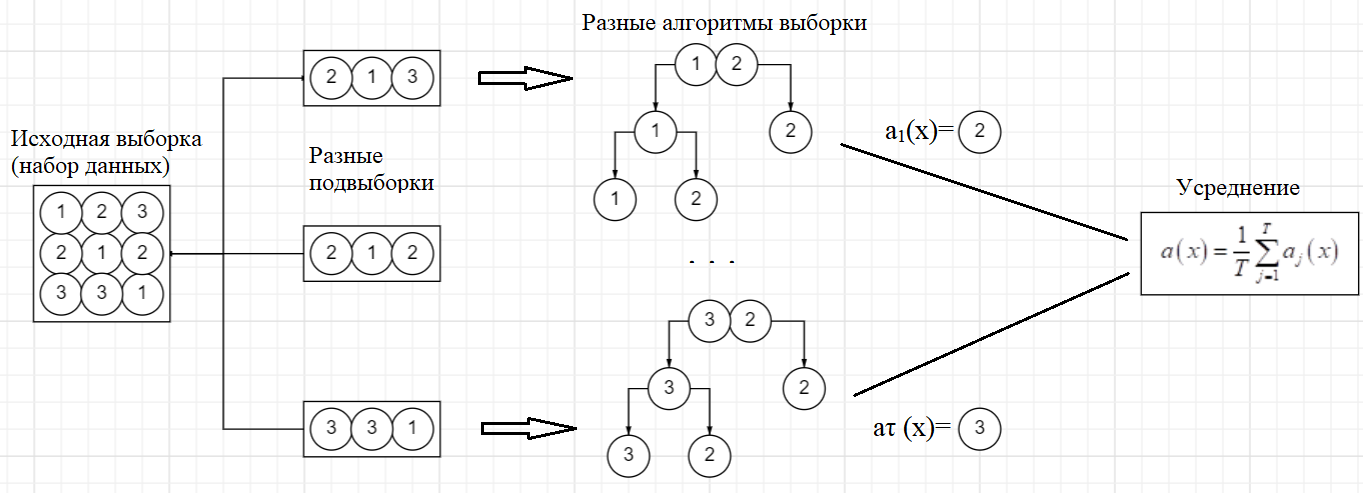


Рисунок 5 – Алгоритм метода Случайный лес

Для начала необходимо выбрать набор данных для обучения системы обнаружения компьютерных атак. Для обучения был выбран один из доступных наборов данных– CICIDS. Разработчик – Canadian Institute for Cybersecurity. Набор данных подготовлен по результатам анализа сетевого трафика в изолированной среде, в которой моделировались действия 25 легальных пользователей, а также вредоносные действия нарушителей. Для создания качественного классификатора необходимы достоверные данные. Наш датасет разбивается на разные подвыборки. На основе наших подвыборок необходимо сформировать несколько разных и, в общем случае, независимых алгоритмов обработки входного вектора a(x). Далее нужно усреднить ответы каждого алгоритма для формирования общего решения по формуле (3.1).

(3.1)

Программа визуализирована в среде Jupyter Notebook на языке Python с использование сторонних библиотек. (Приложение А.)

Были использованы библиотеки:

1. Библиотека Pandas — это библиотека для научных данных на Python, которая предоставляет инструменты и структуры данных для эффективного манипулирования и анализа табличных данных.

Основные возможности библиотеки:

чтение и запись данных в различные форматы (CSV, Excel, SQL и другие);

обработка и анализ данных с использованием различных методов;

построение графиков и визуализация данных;

работа с временными рядами и финансовыми данными.

Библиотека Pandas широко используется в области машинного обучения, статистики и анализа данных. Она позволяет быстро и эффективно обрабатывать большие объёмы данных, что делает её незаменимым инструментом для исследователей и разработчиков.

1. NumPy — это библиотека для научных вычислений на Python, которая предоставляет мощные инструменты для работы с многомерными массивами и матрицами. NumPy позволяет эффективно манипулировать числовыми данными и выполнять сложные математические операции.

Основные возможности библиотеки:

* создание и манипулирование многомерными массивами;
* поддержка векторных и матричных операций;
* интеграция с другими библиотеками, такими как SciPy и Matplotlib;
* оптимизация производительности при работе с большими объёмами данных.

NumPy широко используется в области машинного обучения, обработки сигналов, численного анализа и других областях, где требуется интенсивная работа с числовыми данными. Библиотека позволяет ускорить разработку и повысить эффективность кода, что делает её незаменимым инструментом для исследователей и разработчиков.

1. Scikit-learn (sklearn) — это библиотека машинного обучения на Python, которая предоставляет широкий спектр алгоритмов и инструментов для анализа данных. Она является одной из самых популярных библиотек машинного обучения и широко используется в научных исследованиях и разработке.

Основные возможности библиотеки:

* классификация;
* регрессия;
* кластеризация;
* уменьшение размерности;
* выбор модели;
* предварительная обработка данных.

1. Matplotlib — это библиотека для визуализации данных на Python, которая позволяет создавать графики, диаграммы и другие визуальные представления числовых данных. Она является одной из самых популярных библиотек для визуализации в Python и широко используется в научных исследованиях, анализе данных и других областях.

Основные возможности библиотеки:

* создание различных типов графиков, таких как линейные графики, гистограммы, круговые диаграммы и т. д.;
* настройка внешнего вида графиков с помощью параметров;
* сохранение графиков в различных форматах, таких как PNG, JPEG, SVG и др.;
* интеграция с другими библиотеками, такими как NumPy и Pandas.

1. Seaborn — это библиотека для визуализации данных на Python, которая построена на основе Matplotlib. Она предоставляет простой и удобный интерфейс для создания статистических графиков и диаграмм, что делает её популярной среди исследователей и аналитиков данных.

Основные возможности библиотеки:

* создание различных типов графиков, таких как линейные графики, гистограммы, круговые диаграммы и т. д.;
* настройка внешнего вида графиков с помощью параметров;
* интеграция с другими библиотеками, такими как Pandas и Matplotlib;
* использование цветовых схем и стилей для улучшения читаемости графиков.

После установки библиотек необходимо загрузить наш набор данных (Датасет). (Рисунок 6)

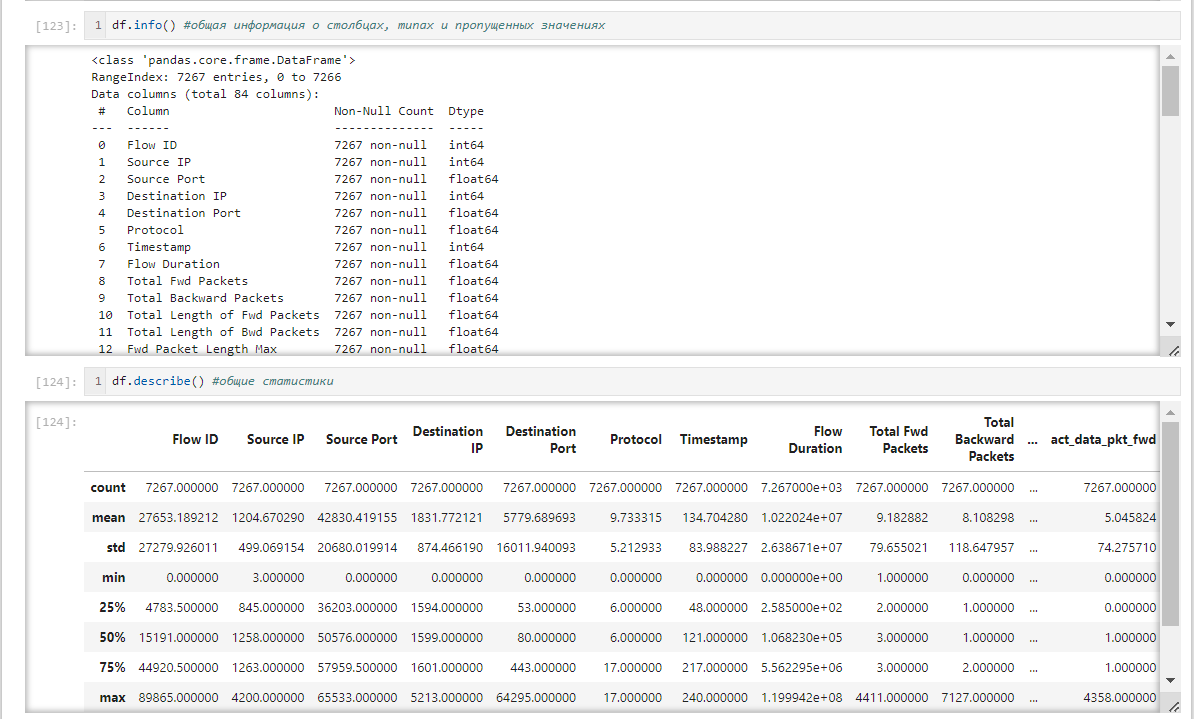


Рисунок – 6 загрузка набора данных

В нашей выборке существует 4 варианта исхода событий(BENIGN – атаки нет, WEB attack – Brute Force, WEB attack –XSS, WEB attack – Sql Injection)

Определим распределение количества атак: (Рисунок 7)

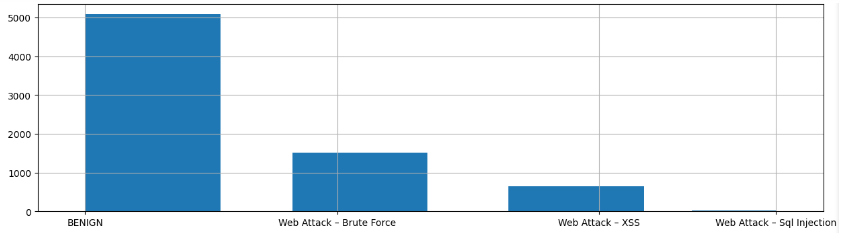


Рисунок – 7 распределение количества атак

Чтобы выявить есть атака или нет заменим значения атаки на (1-есть атака 2-нет атаки). Далее разделим наш набор на тренировочный и тестовый(70% и 30%) и используем метод смешанного леса с использованием 50 деревьев

Метод случайного леса позволяет определить значимость признаков.: (Рисунок 8)

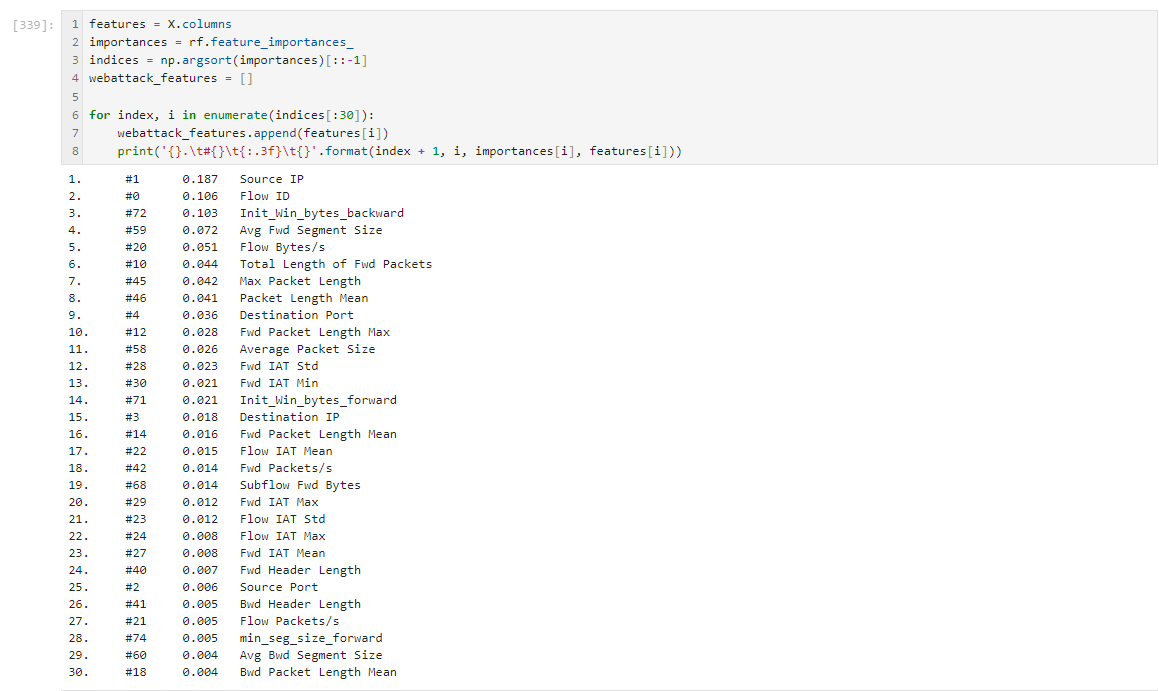


Рисунок 8 – значимость признаков

Продемонстрируем диаграмму для 20 наиболее значимых признаков: (Рисунок 9)

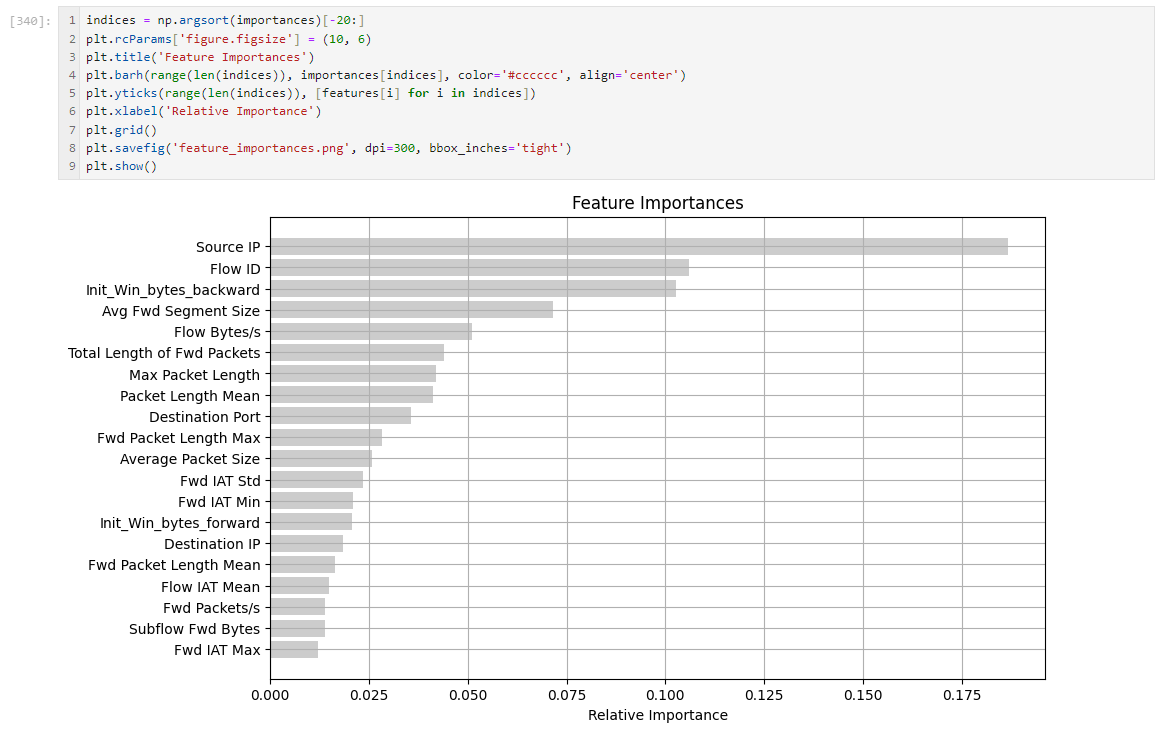


Рисунок 9 – диаграмма для 20 наиболее значимых признаков

Смоделируем корреляционную матрицу с линейными коэффициентами корреляции (коэффициентами корреляции Пирсона), рассчитанными для всех пар двадцати наиболее значимых признаков. (Рисунок 10)

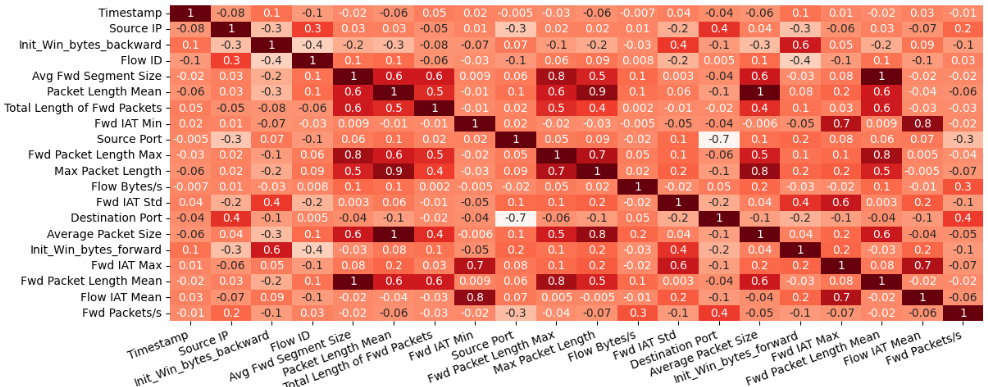


Рисунок 10 - корреляционная матрица для всех пар двадцати наиболее значимых признаков

Из всех пар признаков выделим 10 наиболее важных и также построим корреляционную матрицу. (Рисунок 11)

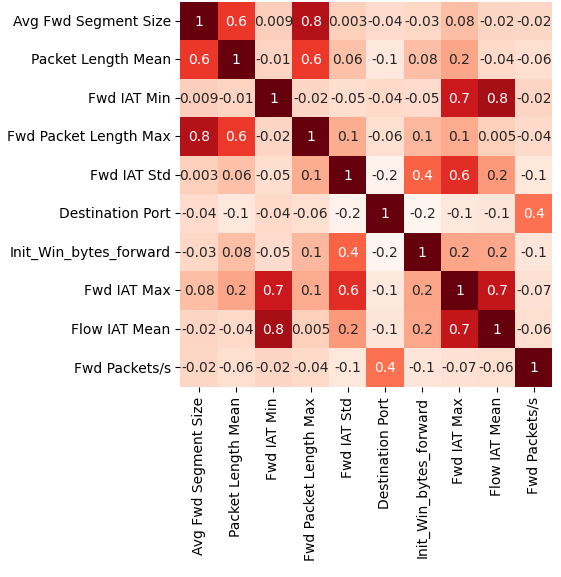


Рисунок 11 – 10 наиболее важных пар признаков

И уже на этой выборке можно построить случайный лес. (Приложение А)

Для того чтобы протестировать наш метод используем 30 % наших данных, оставленных заранее. (Приложение А)

Как можно заметить наш алгоритм справился со своей задачей с высокой точностью и определил все возможные атаки на сеть. (Рисунок 12, 13)

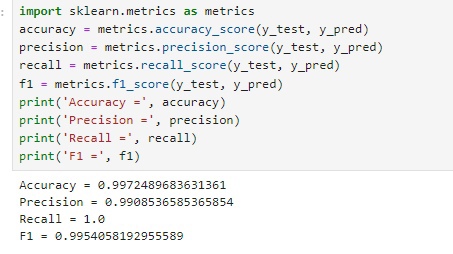


Рисунок 12 – определение точности классификации данных

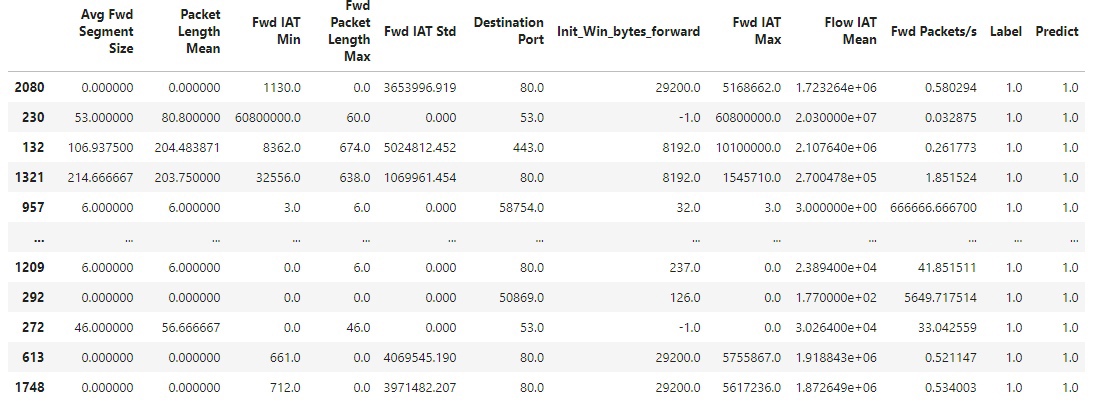


Рисунок 13 – тестовый анализ нашего метода

# 3.4. Вывод по третьей главе

В данной главе был разобран метод машинного обучения, а именно случайный лес. С его помощью удалось классифицировать большой объём данных о сетевых атаках. По значимости признаков сети можно определить является ли взаимодействие с сетью атакой или нет.

# ЗАКЛЮЧЕНИЕ

В ходе выполнения проекта были рассмотрены и проанализированы основные методы искусственного интеллекта, которые могут быть использованы для реализации защиты от сетевых атак. Были изучены теоретические основы и принципы работы методов искусственного интеллекта, а также их применение в области информационной безопасности.

Проведённый анализ показал, что использование методов искусственного интеллекта может значительно повысить эффективность систем защиты от сетевых атак, обеспечивая более точное и быстрое обнаружение аномалий в сетевом трафике. Это позволяет оперативно реагировать на угрозы и предотвращать возможные негативные последствия.

Однако стоит отметить, что внедрение методов искусственного интеллекта в системы защиты требует тщательного подхода к выбору алгоритмов и моделей, а также к настройке параметров их работы. Необходимо учитывать специфику защищаемой сети, её топологию, типы используемых сервисов и приложений, а также характеристики потенциальных угроз.

Кроме того, необходимо обеспечить безопасность самих систем искусственного интеллекта, чтобы предотвратить возможность их использования злоумышленниками для проведения атак или обхода защитных механизмов. Для этого следует применять меры по защите данных, используемых для обучения и настройки моделей, а также обеспечивать контроль доступа к системам искусственного интеллекта.

Таким образом, можно сделать вывод о том, что применение методов искусственного интеллекта является перспективным направлением развития систем защиты от сетевых атак. Однако для достижения максимальной эффективности и надёжности необходимо проводить дальнейшие исследования и разработки в этой области, а также внедрять полученные результаты в практику. Цели и задачи проекта были достигнуты.

# Список использованных источников

1. Конституция РФ
2. Собрание законодательства Российской Федерации
3. Российская Федерация. Законы. Федеральный закон от  27.07.2006 № 149-ФЗ «Об  информации, информационных технологиях и  о защите информации». — Доступ из справ.-правовой системы «КонсультантПлюс».
4. Федеральный закон от 24.04.2020 г. № 123-ФЗ «О проведении эксперимента по установлению специального регулирования в целях создания необходимых условий для разработки и внедрения технологий искусственного интеллекта в субъекте Российской Федерации - городе федерального значения Москве и внесении изменений в статьи 6 и 10 Федерального закона «О персональных данных» [Электронный ресурс]. URL: http://publication.pravo.gov.ru/Document/View/0001202004240030 (дата обращения: 25.11.2023).
5. . Федеральный закон от 27.07.2006 № 152-ФЗ «О персональных данных» [Электронный ресурс]. URL: http://www.consultant.ru/document/  
   cons\_doc\_LAW\_61801/ (дата обращения: 25.11.2023).
6. Федеральный закон "О безопасности критической информационной инфраструктуры Российской Федерации" от 26.07.2017 N 187-ФЗ URL: <https://www.consultant.ru/document/cons_doc_LAW_220885>
7. Указ Президента РФ от 5 декабря 2016 г. N 646 "Об утверждении Доктрины информационной безопасности Российской Федерации" URL: https://base.garant.ru/71556224/
8. Дoмбрoвскaя, Л.A. Современные подходы к защите информации, методы,
9. средства и инструменты защиты / Л.А. Домбровская, Н.А. Яковлева, Р.Е. Стахно. — Электрон. текстовые дан. — Режим доступа: https://www.3minut.ru (дата обращения: 18.06.2023)
10. Кияев, В.И. Безопасность информационных систем / В.И. Кияев, О.Н. Граничин. — М.: Открытый Университет «ИНТУИТ», 2016. — 192 с.
11. Современные системы защиты информации от НСД // Компания «Инфозащита». — Электрон. текстовые дан. — Режим доступа: http://itprotect.ru (дата обращения: 10.06.2023).
12. . Бринк Х., Ричардс Д., Феверолф М. Машинное обучение. СПб.: «Питер», 2017. 336 с
13. Василишин Н.С., Ушаков И.А, Котенко И.В. Исследование алгоритмов анализа сетевого трафика с использованием технологий больших данных для обнаружения компьютерных атак // Аллея науки. 2018. № 6(22). С. 1012–1021.
14. . Браницкий А.А., Котенко И.В. Анализ и классификация методов обнаружения сетевых атак // Труды СПИИРАН. 2016. № 2 (45). С. 207–244.
15. Безопасность информационного пространства : сб. тр. XIX Всерос. науч.-практ. конф. студентов, аспирантов и молодых ученых (Екатеринбург, 8−11 декабря 2020 г.) / [отв. за вып.: А. Ю. Коковихин, Д. М. Назаров, ред. кол.: А. Е. Плахин, Е. Н. Стариков] ; М-во науки и высш. образования Рос. Федерации, Урал. гос. экон. ун-т. — Екатеринбург : Изд-во Урал. гос. экон. ун-та, 2021. — 265 с.
16. Бенгфорт Б., Билбро Р., Океда Т. Прикладной анализ текстовых данных на Python. Машинное обучение и создание приложений обработки естественного языка / Пер. с англ. – СПб.: Питер, 2019. – 368 с.
17. . Intrusion Detection Evaluation Dataset (CICIDS2017). Available at: https://www.unb.ca/cic/datasets/ids2017.html, accessed 16.08.2020.

# Приложение

Приложение А

[141]:

**import pandas as pd import numpy as np**

**from sklearn.ensemble import** RandomForestClassifier **from sklearn.model\_selection import** train\_test\_split **from sklearn.metrics import** r2\_score

**import matplotlib.pyplot as plt**

**import seaborn as sns**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [142]: | df=pd.read\_csv('web\_attacks\_balanced.csv') df.sample(5) *#* | | | | | *,* | *.* | |
| [142]: |  | Flow ID | Source IP | Source Port | Destination IP | | Destination Port | \ |
|  | 4266 | 5085 | 845 | 57386.0 | 1601 | | 80.0 |  |
|  | 899 | 184 | 540 | 443.0 | 1592 | | 59462.0 |  |
|  | 845 | 22681 | 1258 | 50959.0 | 2601 | | 443.0 |  |
|  | 846 | 54093 | 1260 | 23971.0 | 1599 | | 53.0 |  |
|  | 6840 | 76258 | 1268 | 51052.0 | 1599 | | 53.0 |  |

Protocol Timestamp Flow Duration Total Fwd Packets \

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 4266 | 6.0 | | 24 | | 5274392.0 | | | | 3.0 |
| 899 | 6.0 | | 206 | | 50.0 | | | | 3.0 |
| 845 | 6.0 | | 206 | | 60325382.0 | | | | 13.0 |
| 846 | 17.0 | | 206 | | 1174.0 | | | | 2.0 |
| 6840 | 17.0 | | 114 | | 664.0 | | | | 2.0 |
|  | Total Backward | | Packets | | … | min\_seg\_size\_forward | | | Active Mean \ |
| 4266 |  | | 1.0 | | … | 32.0 | | | 0.00000 |
| 899 |  | | 0.0 | | … | 20.0 | | | 0.00000 |
| 845 |  | | 12.0 | | … | 20.0 | | | 69471.16667 |
| 846 |  | | 2.0 | | … | 32.0 | | | 0.00000 |
| 6840 |  | | 2.0 | | … | 20.0 | | | 0.00000 |
|  | Active Std | Active | | Max | Active | | Min | Idle Mean | Idle Std \ |
| 4266 | 0.00000 |  | | 0.0 |  | | 0.0 | 0.000 | 0.00000 |
| 899 | 0.00000 |  | | 0.0 |  | | 0.0 | 0.000 | 0.00000 |
| 845 | 45499.66466 | 162324.0 | | | 50229.0 | | | 9984732.333 | 55483.36011 |
| 846 | 0.00000 | 0.0 | | | 0.0 | | | 0.000 | 0.00000 |
| 6840 | 0.00000 | 0.0 | | | 0.0 | | | 0.000 | 0.00000 |

Idle Max Idle Min Label 4266 0.0 0.0 Web Attack – XSS

899 0.0 0.0 BENIGN

845 10000000.0 9872117.0 BENIGN

|  |  |  |  |
| --- | --- | --- | --- |
| 846 | 0.0 | 0.0 | BENIGN |
| 6840 | 0.0 | 0.0 | BENIGN |

[143]:

[5 rows x 84 columns]

df.info() *# ,*

<class 'pandas.core.frame.DataFrame'> RangeIndex: 7267 entries, 0 to 7266

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Data # | columns (total 84 columns): Column | Non-Null Count | | Dtype | |
| 0 Flow ID | | 7267 | non-null |  | int64 |
| 1 Source IP | | 7267 | non-null |  | int64 |
| 2 Source Port | | 7267 | non-null |  | float64 |
| 3 Destination IP | | 7267 | non-null |  | int64 |
| 4 Destination Port | | 7267 | non-null |  | float64 |
| 5 Protocol | | 7267 | non-null |  | float64 |
| 6 Timestamp | | 7267 | non-null |  | int64 |
| 7 Flow Duration | | 7267 | non-null |  | float64 |
| 8 Total Fwd Packets | | 7267 | non-null |  | float64 |
| 9 Total Backward Packets | | 7267 | non-null |  | float64 |
| 10 Total Length of Fwd Packets | | 7267 | non-null |  | float64 |
| 11 Total Length of Bwd Packets | | 7267 | non-null |  | float64 |
| 12 Fwd Packet Length Max | | 7267 | non-null |  | float64 |
| 13 Fwd Packet Length Min | | 7267 | non-null |  | float64 |
| 14 Fwd Packet Length Mean | | 7267 | non-null |  | float64 |
| 15 Fwd Packet Length Std | | 7267 | non-null |  | float64 |
| 16 Bwd Packet Length Max | | 7267 | non-null |  | float64 |
| 17 Bwd Packet Length Min | | 7267 | non-null |  | float64 |
| 18 Bwd Packet Length Mean | | 7267 | non-null |  | float64 |
| 19 Bwd Packet Length Std | | 7267 | non-null |  | float64 |
| 20 Flow Bytes/s | | 7267 | non-null |  | float64 |
| 21 Flow Packets/s | | 7267 | non-null |  | float64 |
| 22 Flow IAT Mean | | 7267 | non-null |  | float64 |
| 23 Flow IAT Std | | 7267 | non-null |  | float64 |
| 24 Flow IAT Max | | 7267 | non-null |  | float64 |
| 25 Flow IAT Min | | 7267 | non-null |  | float64 |
| 26 Fwd IAT Total | | 7267 | non-null |  | float64 |
| 27 Fwd IAT Mean | | 7267 | non-null |  | float64 |
| 28 Fwd IAT Std | | 7267 | non-null |  | float64 |
| 29 Fwd IAT Max | | 7267 | non-null |  | float64 |
| 30 Fwd IAT Min | | 7267 | non-null |  | float64 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 31 Bwd | IAT Total | 7267 | non-null | float64 |
| 32 Bwd | IAT Mean | 7267 | non-null | float64 |
| 33 Bwd | IAT Std | 7267 | non-null | float64 |
| 34 Bwd | IAT Max | 7267 | non-null | float64 |
| 35 Bwd | IAT Min | 7267 | non-null | float64 |
| 36 Fwd | PSH Flags | 7267 | non-null | float64 |
| 37 Bwd | PSH Flags | 7267 | non-null | float64 |
| 38 Fwd | URG Flags | 7267 | non-null | float64 |
| 39 Bwd | URG Flags | 7267 | non-null | float64 |
| 40 Fwd | Header Length | 7267 | non-null | float64 |
| 41 Bwd | Header Length | 7267 | non-null | float64 |
| 42 Fwd | Packets/s | 7267 | non-null | float64 |
| 43 Bwd | Packets/s | 7267 | non-null | float64 |
| 44 Min | Packet Length | 7267 | non-null | float64 |
| 45 Max | Packet Length | 7267 | non-null | float64 |
| 46 Packet Length Mean | | 7267 | non-null | float64 |
| 47 Packet Length Std | | 7267 | non-null | float64 |
| 48 Packet Length Variance | | 7267 | non-null | float64 |
| 49 FIN Flag Count | | 7267 | non-null | float64 |
| 50 SYN Flag Count | | 7267 | non-null | float64 |
| 51 RST Flag Count | | 7267 | non-null | float64 |
| 52 PSH Flag Count | | 7267 | non-null | float64 |
| 53 ACK Flag Count | | 7267 | non-null | float64 |
| 54 URG Flag Count | | 7267 | non-null | float64 |
| 55 CWE Flag Count | | 7267 | non-null | float64 |
| 56 ECE Flag Count | | 7267 | non-null | float64 |
| 57 Down/Up Ratio | | 7267 | non-null | float64 |
| 58 Average Packet Size | | 7267 | non-null | float64 |
| 59 Avg Fwd Segment Size | | 7267 | non-null | float64 |
| 60 Avg Bwd Segment Size | | 7267 | non-null | float64 |
| 61 Fwd Avg Bytes/Bulk | | 7267 | non-null | float64 |
| 62 Fwd Avg Packets/Bulk | | 7267 | non-null | float64 |
| 63 Fwd Avg Bulk Rate | | 7267 | non-null | float64 |
| 64 Bwd Avg Bytes/Bulk | | 7267 | non-null | float64 |
| 65 Bwd Avg Packets/Bulk | | 7267 | non-null | float64 |
| 66 Bwd Avg Bulk Rate | | 7267 | non-null | float64 |
| 67 Subflow Fwd Packets | | 7267 | non-null | float64 |
| 68 Subflow Fwd Bytes | | 7267 | non-null | float64 |
| 69 Subflow Bwd Packets | | 7267 | non-null | float64 |
| 70 Subflow Bwd Bytes | | 7267 | non-null | float64 |
| 71 Init\_Win\_bytes\_forward | | 7267 | non-null | float64 |
| 72 Init\_Win\_bytes\_backward | | 7267 | non-null | float64 |
| 73 act\_data\_pkt\_fwd | | 7267 | non-null | float64 |
| 74 min\_seg\_size\_forward | | 7267 | non-null | float64 |
| 75 Active Mean | | 7267 | non-null | float64 |
| 76 Active Std | | 7267 | non-null | float64 |
| 77 Active Max | | 7267 | non-null | float64 |
| 78 Active Min | | 7267 | non-null | float64 |

|  |  |  |
| --- | --- | --- |
| 79 Idle Mean | 7267 non-null | float64 |
| 80 Idle Std | 7267 non-null | float64 |
| 81 Idle Max | 7267 non-null | float64 |
| 82 Idle Min | 7267 non-null | float64 |
| 83 Label | 7267 non-null | object |

[144]:

df.describe() *#*

dtypes: float64(79), int64(4), object(1) memory usage: 4.7+ MB

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| [144]: |  | Flow ID | Source IP | Source Port | Destination IP \ |
|  | count | 7267.000000 | 7267.000000 | 7267.000000 | 7267.000000 |
|  | mean | 27653.189212 | 1204.670290 | 42830.419155 | 1831.772121 |
|  | std | 27279.926011 | 499.069154 | 20680.019914 | 874.466190 |
|  | min | 0.000000 | 3.000000 | 0.000000 | 0.000000 |
|  | 25% | 4783.500000 | 845.000000 | 36203.000000 | 1594.000000 |
|  | 50% | 15191.000000 | 1258.000000 | 50576.000000 | 1599.000000 |
|  | 75% | 44920.500000 | 1263.000000 | 57959.500000 | 1601.000000 |
|  | max | 89865.000000 | 4200.000000 | 65533.000000 | 5213.000000 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Destination Port | | Protocol | | Timestamp | Flow Duration | | \ | |
| count | 7267.000000 | | 7267.000000 | | 7267.000000 | 7.267000e+03 | |  | |
| mean | 5779.689693 | | 9.733315 | | 134.704280 | 1.022024e+07 | |  | |
| std | 16011.940093 | | 5.212933 | | 83.988227 | 2.638671e+07 | |  | |
| min | 0.000000 | | 0.000000 | | 0.000000 | 0.000000e+00 | |  | |
| 25% | 53.000000 | | 6.000000 | | 48.000000 | 2.585000e+02 | |  | |
| 50% | 80.000000 | | 6.000000 | | 121.000000 | 1.068230e+05 | |  | |
| 75% | 443.000000 | | 17.000000 | | 217.000000 | 5.562295e+06 | |  | |
| max | 64295.000000 | | 17.000000 | | 240.000000 | 1.199942e+08 | |  | |
|  | Total | Fwd Packets | Total | Backward Packets | | … | act\_data\_pkt\_fwd | | \ |
| count |  | 7267.000000 |  | 7267.000000 | | … | 7267.000000 | |  |
| mean |  | 9.182882 |  | 8.108298 | | … | 5.045824 | |  |
| std |  | 79.655021 |  | 118.647957 | | … | 74.275710 | |  |
| min |  | 1.000000 |  | 0.000000 | | … | 0.000000 | |  |
| 25% |  | 2.000000 |  | 1.000000 | | … | 0.000000 | |  |
| 50% |  | 3.000000 |  | 1.000000 | | … | 1.000000 | |  |
| 75% |  | 3.000000 |  | 2.000000 | | … | 1.000000 | |  |
| max |  | 4411.000000 |  | 7127.000000 | | … | 4358.000000 | |  |

min\_seg\_size\_forward Active Mean Active Std Active Max \

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| count | 7267.000000 | 7.267000e+03 | 7.267000e+03 | 7.267000e+03 |
| mean | 27.422595 | 5.498477e+04 | 3.038243e+04 | 1.119693e+05 |
| std | 6.111466 | 5.666915e+05 | 2.681633e+05 | 8.214085e+05 |
| min | 0.000000 | 0.000000e+00 | 0.000000e+00 | 0.000000e+00 |
| 25% | 20.000000 | 0.000000e+00 | 0.000000e+00 | 0.000000e+00 |
| 50% | 32.000000 | 0.000000e+00 | 0.000000e+00 | 0.000000e+00 |

75% 32.000000 0.000000e+00 0.000000e+00 0.000000e+00

max 44.000000 3.550000e+07 9.340273e+06 3.550000e+07

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Active Min | Idle Mean | Idle Std | Idle Max | Idle Min |
| count | 7.267000e+03 | 7.267000e+03 | 7.267000e+03 | 7.267000e+03 | 7.267000e+03 |
| mean | 3.925752e+04 | 2.475254e+06 | 8.708279e+04 | 2.535257e+06 | 2.380353e+06 |
| std | 5.289145e+05 | 1.071754e+07 | 1.386914e+06 | 1.090592e+07 | 1.060371e+07 |
| min | 0.000000e+00 | 0.000000e+00 | 0.000000e+00 | 0.000000e+00 | 0.000000e+00 |
| 25% | 0.000000e+00 | 0.000000e+00 | 0.000000e+00 | 0.000000e+00 | 0.000000e+00 |
| 50% | 0.000000e+00 | 0.000000e+00 | 0.000000e+00 | 0.000000e+00 | 0.000000e+00 |
| 75% | 0.000000e+00 | 0.000000e+00 | 0.000000e+00 | 0.000000e+00 | 0.000000e+00 |
| max | 3.550000e+07 | 1.190000e+08 | 3.770000e+07 | 1.190000e+08 | 1.190000e+08 |

[8 rows x 83 columns]

[145]:

df.columns

1. : Index(['Flow ID', 'Source IP', 'Source Port', 'Destination IP', 'Destination Port', 'Protocol', 'Timestamp', 'Flow Duration', 'Total Fwd Packets', 'Total Backward Packets',

'Total Length of Fwd Packets', 'Total Length of Bwd Packets', 'Fwd Packet Length Max', 'Fwd Packet Length Min',

'Fwd Packet Length Mean', 'Fwd Packet Length Std', 'Bwd Packet Length Max', 'Bwd Packet Length Min',

'Bwd Packet Length Mean', 'Bwd Packet Length Std', 'Flow Bytes/s', 'Flow Packets/s', 'Flow IAT Mean', 'Flow IAT Std', 'Flow IAT Max', 'Flow IAT Min', 'Fwd IAT Total', 'Fwd IAT Mean', 'Fwd IAT Std', 'Fwd IAT Max', 'Fwd IAT Min', 'Bwd IAT Total', 'Bwd IAT Mean', 'Bwd IAT Std', 'Bwd IAT Max', 'Bwd IAT Min', 'Fwd PSH Flags',

'Bwd PSH Flags', 'Fwd URG Flags', 'Bwd URG Flags', 'Fwd Header Length', 'Bwd Header Length', 'Fwd Packets/s', 'Bwd Packets/s',

'Min Packet Length', 'Max Packet Length', 'Packet Length Mean', 'Packet Length Std', 'Packet Length Variance', 'FIN Flag Count',

'SYN Flag Count', 'RST Flag Count', 'PSH Flag Count', 'ACK Flag Count', 'URG Flag Count', 'CWE Flag Count', 'ECE Flag Count', 'Down/Up Ratio', 'Average Packet Size', 'Avg Fwd Segment Size', 'Avg Bwd Segment Size', 'Fwd Avg Bytes/Bulk', 'Fwd Avg Packets/Bulk', 'Fwd Avg Bulk Rate',

'Bwd Avg Bytes/Bulk', 'Bwd Avg Packets/Bulk', 'Bwd Avg Bulk Rate', 'Subflow Fwd Packets', 'Subflow Fwd Bytes', 'Subflow Bwd Packets', 'Subflow Bwd Bytes', 'Init\_Win\_bytes\_forward', 'Init\_Win\_bytes\_backward', 'act\_data\_pkt\_fwd', 'min\_seg\_size\_forward', 'Active Mean', 'Active Std', 'Active Max', 'Active Min', 'Idle Mean', 'Idle Std', 'Idle Max', 'Idle Min', 'Label'],

dtype='object')

1. :

col=['Flow ID', 'Source IP', 'Source Port', 'Destination IP', 'Destination Port', 'Protocol', 'Timestamp', 'Flow Duration',

1. :

X=pd.DataFrame()

**for** i **in** col:

**if** df[i].dtype.name != 'object': X[i]=df[i].copy() X.loc[X[i].isna(), i]=X[i].median()

**else**:

X[i]=pd.factorize(df[i])[0]

'Total Fwd Packets', 'Total Backward Packets',

'Total Length of Fwd Packets', 'Total Length of Bwd Packets', 'Fwd Packet Length Max', 'Fwd Packet Length Min',

'Fwd Packet Length Mean', 'Fwd Packet Length Std', 'Bwd Packet Length Max', 'Bwd Packet Length Min',

'Bwd Packet Length Mean', 'Bwd Packet Length Std', 'Flow Bytes/s', 'Flow Packets/s', 'Flow IAT Mean', 'Flow IAT Std', 'Flow IAT Max', 'Flow IAT Min', 'Fwd IAT Total', 'Fwd IAT Mean', 'Fwd IAT Std', 'Fwd IAT Max', 'Fwd IAT Min', 'Bwd IAT Total', 'Bwd IAT Mean', 'Bwd IAT Std', 'Bwd IAT Max', 'Bwd IAT Min', 'Fwd PSH Flags',

'Bwd PSH Flags', 'Fwd URG Flags', 'Bwd URG Flags', 'Fwd Header Length', 'Bwd Header Length', 'Fwd Packets/s', 'Bwd Packets/s',

'Min Packet Length', 'Max Packet Length', 'Packet Length Mean', 'Packet Length Std', 'Packet Length Variance', 'FIN Flag Count',

'SYN Flag Count', 'RST Flag Count', 'PSH Flag Count', 'ACK Flag Count', 'URG Flag Count', 'CWE Flag Count', 'ECE Flag Count', 'Down/Up Ratio', 'Average Packet Size', 'Avg Fwd Segment Size', 'Avg Bwd Segment Size', 'Fwd Avg Bytes/Bulk', 'Fwd Avg Packets/Bulk', 'Fwd Avg Bulk Rate',

'Bwd Avg Bytes/Bulk', 'Bwd Avg Packets/Bulk', 'Bwd Avg Bulk Rate', 'Subflow Fwd Packets', 'Subflow Fwd Bytes', 'Subflow Bwd Packets', 'Subflow Bwd Bytes', 'Init\_Win\_bytes\_forward', 'Init\_Win\_bytes\_backward', 'act\_data\_pkt\_fwd', 'min\_seg\_size\_forward', 'Active Mean', 'Active Std', 'Active Max', 'Active Min', 'Idle Mean', 'Idle Std', 'Idle Max', 'Idle Min', 'Label']

1. :

X.sample(100)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| [148]: | Flow ID | Source IP | Source Port | Destination IP | Destination Port | \ |
| 6851 | 6495 | 1264 | 52636.0 | 1123 | 80.0 |  |
| 5818 | 7771 | 920 | 80.0 | 1596 | 35945.0 |  |
| 4217 | 85070 | 1266 | 34168.0 | 5164 | 443.0 |  |
| 1690 | 5223 | 845 | 58930.0 | 1601 | 80.0 |  |
| 2996 | 4127 | 845 | 45756.0 | 1601 | 80.0 |  |
| … | … | … | … | … | … |  |
| 2471 | 946 | 1258 | 52719.0 | 841 | 443.0 |  |
| 4574 | 5364 | 845 | 60572.0 | 1601 | 80.0 |  |
| 3937 | 14219 | 1263 | 61111.0 | 1590 | 53.0 |  |
| 1940 | 2355 | 1258 | 52017.0 | 975 | 443.0 |  |
| 2529 | 3102 | 1264 | 50168.0 | 1006 | 443.0 |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Protocol | Timestamp | Flow Duration | Total | Fwd | Packets | \ |
| 6851 | 6.0 | 114 | 5332734.0 |  |  | 3.0 |  |
| 5818 | 6.0 | 67 | 36.0 |  |  | 1.0 |  |
| 4217 | 6.0 | 23 | 4186562.0 |  |  | 38.0 |  |
| 1690 | 6.0 | 218 | 5756138.0 |  |  | 3.0 |  |
| 2996 | 6.0 | 234 | 5441768.0 |  |  | 3.0 |  |
| … | … | … | … |  | … |  |  |
| 2471 | 6.0 | 227 | 115361310.0 |  |  | 49.0 |  |
| 4574 | 6.0 | 29 | 5177649.0 |  |  | 3.0 |  |
| 3937 | 17.0 | 18 | 1349736.0 |  |  | 1.0 |  |
| 1940 | 6.0 | 220 | 1025.0 |  |  | 2.0 |  |
| 2529 | 6.0 | 227 | 87879.0 |  |  | 3.0 |  |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Total | Backward | | Packets | | … | min\_seg\_size\_forward | | | Active Mean | \ |
| 6851 |  |  | | 1.0 | | … | 20.0 | | | 0.00 |  |
| 5818 |  |  | | 1.0 | | … | 32.0 | | | 0.00 |  |
| 4217 |  |  | | 44.0 | | … | 20.0 | | | 0.00 |  |
| 1690 |  |  | | 1.0 | | … | 32.0 | | | 0.00 |  |
| 2996  …  2471 |  |  | | 1.0  … …  71.0 | | …  … | 32.0  …  20.0 | | | 0.00  …  27274.25 |  |
| 4574 |  |  | | 1.0 | | … | 32.0 | | | 0.00 |  |
| 3937 |  |  | | 1.0 | | … | 20.0 | | | 0.00 |  |
| 1940 |  |  | | 0.0 | | … | 20.0 | | | 0.00 |  |
| 2529 |  |  | | 0.0 | | … | 20.0 | | | 0.00 |  |
|  | Active Std | | Active Max | | | Active Min | |  | Idle Mean | Idle Std | \ |
| 6851 | 0.00000 | | 0.0 | | | 0.0 | |  | 0.000 | 0.000 |  |
| 5818 | 0.00000 | | 0.0 | | | 0.0 | |  | 0.000 | 0.000 |  |
| 4217 | 0.00000 | | 0.0 | | | 0.0 | |  | 0.000 | 0.000 |  |
| 1690 | 0.00000 | | 0.0 | | | 0.0 | |  | 0.000 | 0.000 |  |
| 2996  … | 0.00000  … | | 0.0  … | | | 0.0  … | | … | 0.000  … | 0.000 |  |
| 2471 | 41919.07667 | | 160385.0 | | | 14992.0 | | 9586163.667 | | 1438342.022 | |
| 4574 | 0.00000 | | 0.0 | | | 0.0 | | 0.000 | | 0.000 | |
| 3937 | 0.00000 | | 0.0 | | | 0.0 | | 0.000 | | 0.000 | |
| 1940 | 0.00000 | | 0.0 | | | 0.0 | | 0.000 | | 0.000 | |
| 2529 | 0.00000 | | 0.0 | | | 0.0 | | 0.000 | | 0.000 | |
|  | Idle | Max | Idle | | Min | Label | | | | | |
| 6851 |  | 0.0 |  | | 0.0 | 0 | | | | | |
| 5818 |  | 0.0 |  | | 0.0 | 0 | | | | | |
| 4217 |  | 0.0 |  | | 0.0 | 0 | | | | | |
| 1690 |  | 0.0 |  | | 0.0 | 1 | | | | | |
| 2996  … | … | 0.0 | … | | 0.0  … | 1 | | | | | |

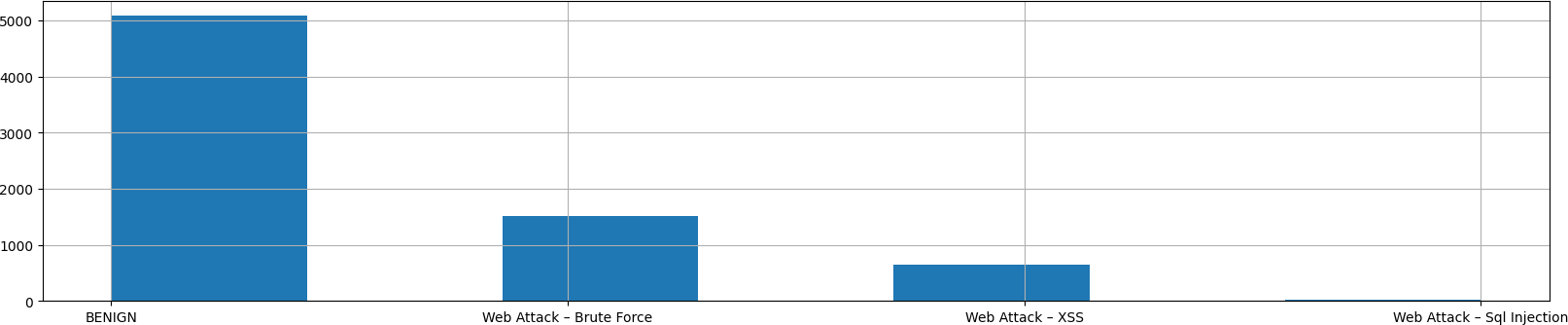
2471 10000000.0 5018832.0 0

|  |  |  |  |
| --- | --- | --- | --- |
| 4574 | 0.0 | 0.0 | 2 |
| 3937 | 0.0 | 0.0 | 0 |
| 1940 | 0.0 | 0.0 | 0 |
| 2529 | 0.0 | 0.0 | 0 |

[100 rows x 84 columns]

1. :

df['Label'].hist(bins=7, figsize=(20,4)); *#*



1. :

*# Y*

*1*

Y=df['Label'].apply(**lambda** x: 0 **if** x=='BENING' **else** 1).values

1. :

y = df['Label'].values

X = df.drop(columns=['Label']) print(X.shape, y.shape)

(7267, 83) (7267,)

1. :

*#*

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3,␣

↪random\_state=42)

print(X\_train.shape, y\_train.shape) print(X\_test.shape, y\_test.shape)

1. :

(5086, 83) (5086,)

(2181, 83) (2181,)

**from sklearn.ensemble import** RandomForestClassifier

rf = RandomForestClassifier(n\_estimators=50, random\_state=42, oob\_score=**True**) rf.fit(X\_train, y\_train)

*# Score = mean accuracy on the given test data and labels*

print('R^2 Training Score: **{:.2f} \n**R^2 Validation Score: **{:.2f} \n**Out-of-bag␣

↪Score: **{:.2f}**'

.format(rf.score(X\_train, y\_train), rf.score(X\_test, y\_test), rf.

↪oob\_score\_))

1. :

features = X.columns

importances = rf.feature\_importances\_ indices = np.argsort(importances)[::-1] webattack\_features = []

**for** index, i **in** enumerate(indices[:20]): webattack\_features.append(features[i]) print('**{}**.**\t**#**{}\t{:.3f}\t{}**'.format(index + 1, i, importances[i],␣

↪features[i]))

R^2 Training Score: 1.00 R^2 Validation Score: 1.00 Out-of-bag Score: 1.00

indices = np.argsort(importances)[-20:] plt.rcParams['figure.figsize'] = (10, 6) plt.title('Feature Importances')

plt.barh(range(len(indices)), importances[indices], color='#cccccc',␣

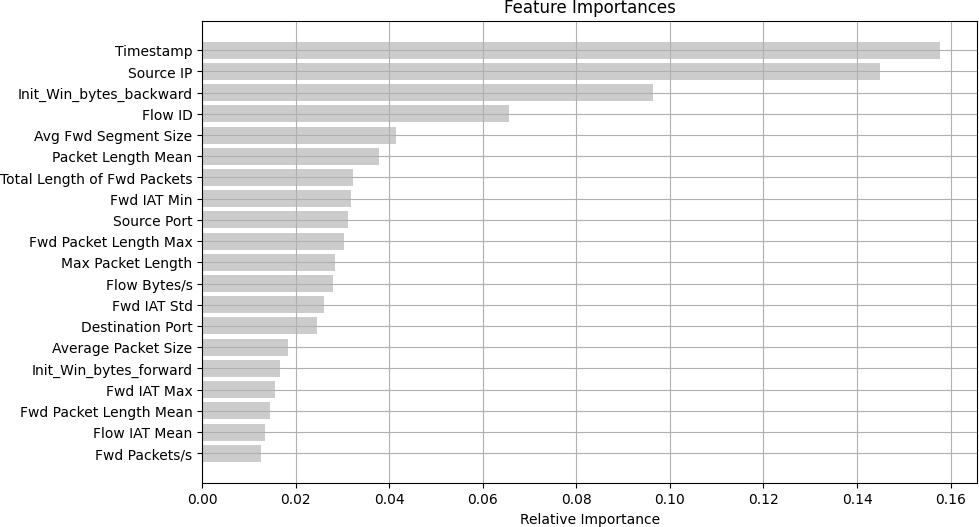
↪align='center')

plt.yticks(range(len(indices)), [features[i] **for** i **in** indices]) plt.xlabel('Relative Importance')

plt.grid()

plt.savefig('feature\_importances.png', dpi=300, bbox\_inches='tight') plt.show()

|  |  |  |  |
| --- | --- | --- | --- |
|  | 1. #6 | 0.158 | Timestamp |
| 2. #1 | 0.145 | Source IP |
| 3. #72 | 0.096 | Init\_Win\_bytes\_backward |
| 4. #0 | 0.066 | Flow ID |
| 5. #59 | 0.041 | Avg Fwd Segment Size |
| 6. #46 | 0.038 | Packet Length Mean |
| 7. #10 | 0.032 | Total Length of Fwd Packets |
| 8. #30 | 0.032 | Fwd IAT Min |
| 9. #2 | 0.031 | Source Port |
| 10. #12 | 0.030 | Fwd Packet Length Max |
| 11. #45 | 0.028 | Max Packet Length |
| 12. #20 | 0.028 | Flow Bytes/s |
| 13. #28 | 0.026 | Fwd IAT Std |
| 14. #4 | 0.025 | Destination Port |
| 15. #58 | 0.018 | Average Packet Size |
| 16. #71 | 0.017 | Init\_Win\_bytes\_forward |
| 17. #29 | 0.015 | Fwd IAT Max |
| 18. #14 | 0.015 | Fwd Packet Length Mean |
| 19. #22 | 0.014 | Flow IAT Mean |
| [155]: | 20. #42 | 0.013 | Fwd Packets/s |



[156]:

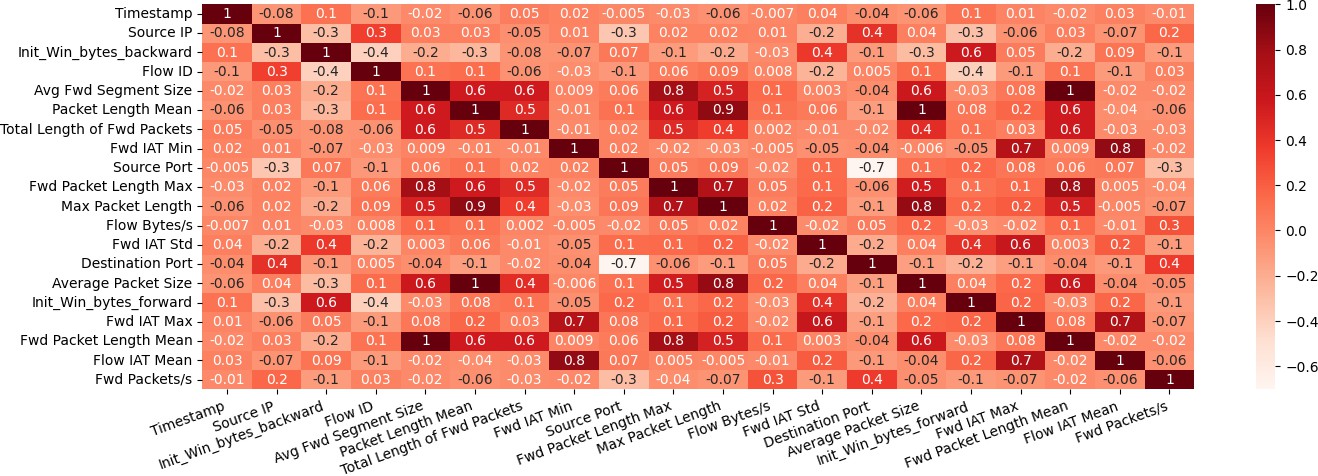
**import seaborn as sns**

corr\_matrix = df[webattack\_features].corr() plt.rcParams['figure.figsize'] = (16, 5)

g = sns.heatmap(corr\_matrix, annot=**True**, fmt='.1g', cmap='Reds') g.set\_xticklabels(g.get\_xticklabels(), verticalalignment='top',␣

↪horizontalalignment='right', rotation=20);

plt.savefig('corr\_heatmap.png', dpi=300, bbox\_inches='tight')



[157]:

to\_be\_removed = {'Timestamp', 'Source IP',

'Flow ID', 'Init\_Win\_bytes\_backward',

'Source Port', 'Average Packet Size', 'Flow Bytes/s',

'Max Packet Length',

'Fwd Packet Length Mean', 'Total Length of Fwd Packets'}

webattack\_features = [item **for** item **in** webattack\_features **if** item **not in**␣

↪to\_be\_removed]

webattack\_features = webattack\_features[:10] webattack\_features

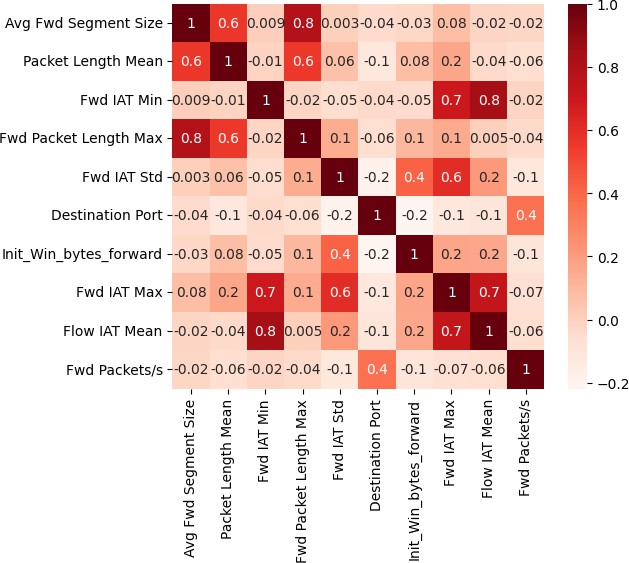
1. : ['Avg Fwd Segment Size', 'Packet Length Mean', 'Fwd IAT Min',

'Fwd Packet Length Max', 'Fwd IAT Std', 'Destination Port', 'Init\_Win\_bytes\_forward', 'Fwd IAT Max',

'Flow IAT Mean', 'Fwd Packets/s']

1. :

corr\_matrix = df[webattack\_features].corr() plt.rcParams['figure.figsize'] = (6, 5) sns.heatmap(corr\_matrix, annot=**True**, fmt='.1g', cmap='Reds');



1. :

**from sklearn.model\_selection import** GridSearchCV

df = pd.read\_csv('web\_attacks\_balanced.csv')

df['Label'] = df['Label'].apply(**lambda** x: 0 **if** x == 'BENIGN' **else** 1) y = df['Label'].values

X = df[webattack\_features] print(X.shape, y.shape)

(7267, 10) (7267,)

1. :

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3,␣

↪random\_state=42)

print(X\_train.shape, y\_train.shape) print(X\_test.shape, y\_test.shape)

(5086, 10) (5086,)

(2181, 10) (2181,)

1. :

rfc = RandomForestClassifier(max\_depth=17, max\_features=10, min\_samples\_leaf=3,␣

↪n\_estimators=50, random\_state=42, oob\_score=**True**)

*# rfc = RandomForestClassifier(n\_estimators=250, random\_state=1)*

rfc.fit(X\_train, y\_train)

1. : RandomForestClassifier(max\_depth=17, max\_features=10, min\_samples\_leaf=3,

n\_estimators=50, oob\_score=True, random\_state=42)

1. :

**import time**

seconds = time.time()

y\_pred = rfc.predict(X\_test)

print("Total operation time:", time.time() - seconds, "seconds")

print("Benign records detected (0), attacks detected (1):") unique, counts = np.unique(y\_pred, return\_counts=**True**) dict(zip(unique, counts))

Total operation time: 0.011967897415161133 seconds Benign records detected (0), attacks detected (1):

[162]: {0: 1525, 1: 656}

1. :

features = X.columns

importances = rfc.feature\_importances\_ indices = np.argsort(importances)[::-1]

**for** index, i **in** enumerate(indices[:10]): print('**{}**.**\t**#**{}\t{:.3f}\t{}**'.format(index + 1, i, importances[i],␣

↪features[i]))

|  |  |  |
| --- | --- | --- |
| 1. #1 | 0.609 | Packet Length Mean |
| 2. #5 | 0.251 | Destination Port |
| 3. #6 | 0.040 | Init\_Win\_bytes\_forward |
| 4. #0 | 0.037 | Avg Fwd Segment Size |
| 5. #2 | 0.023 | Fwd IAT Min |
| 6. #8 | 0.023 | Flow IAT Mean |
| 7. #3 | 0.006 | Fwd Packet Length Max |
| 8. #9 | 0.005 | Fwd Packets/s |
| 9. #7 | 0.003 | Fwd IAT Max |
| 10. #4 | 0.002 | Fwd IAT Std |

1. :

**from sklearn.metrics import** confusion\_matrix y\_pred = rfc.predict(X\_test) confusion\_matrix(y\_test, y\_pred)

1. : array([[1525, 6],

[ 0, 650]], dtype=int64)

1. :

**import sklearn.metrics as metrics**

accuracy = metrics.accuracy\_score(y\_test, y\_pred) precision = metrics.precision\_score(y\_test, y\_pred) recall = metrics.recall\_score(y\_test, y\_pred)

f1 = metrics.f1\_score(y\_test, y\_pred) print('Accuracy =', accuracy) print('Precision =', precision) print('Recall =', recall)

print('F1 =', f1)

1. :

Accuracy = 0.9972489683631361

Precision = 0.9908536585365854

Recall = 1.0

F1 = 0.9954058192955589

y\_pred = rfc.predict(X\_test) confusion\_matrix(y\_test, y\_pred)

1. : array([[1525, 6],

[ 0, 650]], dtype=int64)

1. :

predict = pd.DataFrame({'Predict': rfc.predict(X\_test)}) label = pd.DataFrame({'Label': y\_test})

result = X\_test.join(label).join(predict) result[result['Predict'] == 1]

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [167]: | Avg | Fwd | Segment Size | | Packet | Length Mean | | Fwd IAT Min | \ |
| 2080 |  |  | 0.000000 | |  | 0.000000 | | 1130.0 |  |
| 230 |  |  | 53.000000 | |  | 80.800000 | | 60800000.0 |  |
| 132 |  |  | 106.937500 | |  | 204.483871 | | 8362.0 |  |
| 1321 |  |  | 214.666667 | |  | 203.750000 | | 32556.0 |  |
| 957  …  1209 |  |  | 6.000000  …  6.000000 | |  | 6.000000  …  6.000000 | | 3.0  …  0.0 |  |
| 292 |  |  | 0.000000 | |  | 0.000000 | | 0.0 |  |
| 272 |  |  | 46.000000 | |  | 56.666667 | | 0.0 |  |
| 613 |  |  | 0.000000 | |  | 0.000000 | | 661.0 |  |
| 1748 |  |  | 0.000000 | |  | 0.000000 | | 712.0 |  |
|  | Fwd | Packet | | Length Max | Fwd IAT Std | | Destination Port | | \ |
| 2080 |  |  | | 0.0 | 3653996.919 | | 80.0 | |  |
| 230 |  |  | | 60.0 | 0.000 | | 53.0 | |  |
| 132 |  |  | | 674.0 | 5024812.452 | | 443.0 | |  |
| 1321 |  |  | | 638.0 | 1069961.454 | | 80.0 | |  |
| 957  …  1209 |  |  | | 6.0  …  6.0 | 0.000  …  0.000 | | 58754.0  …  80.0 | |  |
| 292 |  |  | | 0.0 | 0.000 | | 50869.0 | |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 272 | 46.0 | 0.000 | 53.0 | | |
| 613 | 0.0 | 4069545.190 | 80.0 | | |
| 1748 | 0.0 | 3971482.207 | 80.0 | | |
|  | Init\_Win\_bytes\_forward | Fwd IAT Max | Flow IAT Mean | Fwd Packets/s | \ |
| 2080 | 29200.0 | 5168662.0 | 1.723264e+06 | 0.580294 |  |
| 230 | -1.0 | 60800000.0 | 2.030000e+07 | 0.032875 |  |
| 132 | 8192.0 | 10100000.0 | 2.107640e+06 | 0.261773 |  |
| 1321 | 8192.0 | 1545710.0 | 2.700478e+05 | 1.851524 |  |
| 957  …  1209 | 32.0  …  237.0 | 3.0  …  0.0 | 3.000000e+00  …  2.389400e+04 | 666666.666700  …  41.851511 |  |
| 292 | 126.0 | 0.0 | 1.770000e+02 | 5649.717514 |  |
| 272 | -1.0 | 0.0 | 3.026400e+04 | 33.042559 |  |
| 613 | 29200.0 | 5755867.0 | 1.918843e+06 | 0.521147 |  |
| 1748 | 29200.0 | 5617236.0 | 1.872649e+06 | 0.534003 |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Label | Predict |
| 2080 | 1.0 | 1.0 |
| 230 | 1.0 | 1.0 |
| 132 | 1.0 | 1.0 |
| 1321 | 1.0 | 1.0 |
| 957 | 1.0 | 1.0 |
| … | … | … |
| 1209 | 1.0 | 1.0 |
| 292 | 1.0 | 1.0 |
| 272 | 1.0 | 1.0 |
| 613 | 1.0 | 1.0 |
| 1748 | 1.0 | 1.0 |
| [204 | rows x | 12 columns] |