

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное
учреждение высшего образования
«СЕВАСТОПОЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И УПРАВЛЕНИЯ
В ТЕХНИЧЕСКИХ СИСТЕМАХ

Кафедра «Информационные технологии и компьютерные системы»

М Е Т О Д Ы ИНТЕЛЛЕКТУАЛЬНОЙ ЗАЩИТЫ ИНФОРМАЦИИ

Методические указания
к выполнению лабораторных работ
по дисциплине «Технологии интеллектуальной защиты информации»
для студентов направления подготовки
09.04.01 «Информатика и вычислительная техника»

Севастополь
СевГУ
2019

Р е ц е н з е н т:

К.В. Кротов – доцент кафедры «Информационные системы»,
канд. техн. наук, доцент

Составители: А.А. Брюховецкий, К.С. Ткаченко

М54 Методы интеллектуальной защиты информации: метод. указания к выполнению лабораторных работ для студентов направления подготовки 09.04.01 «Информатика и вычислительная техника» по дисциплине «Технологии интеллектуальной защиты информации» / Сост. А.А. Брюховецкий, К.С. Ткаченко. – Электрон. дан. – Севастополь: СевГУ, 2019 г. – Режим доступа: свободный после авторизации. – Загл. с экрана. – 37 с.

Настоящие методические указания предназначены для оказания помощи студентам при выполнении лабораторных работ на тему «Методы интеллектуальной защиты информации» по дисциплине «Технологии интеллектуальной защиты информации». Рассматриваются методы интеллектуального анализа данных, используемых в системах защиты информации.

УДК 004.056.5

Методические указания рассмотрены и утверждены на заседании кафедры «Информационные технологии и компьютерные системы», протокол № 3 от 04 декабря 2018 г.

Текстовое (символьное) издание (1,5 Мб).

Системные требования: Intel, 3,4 GHz; 150 Мб; Windows XP/Vista/7; DVD-ROM; 1 Гб свободного места на жестком диске; программа для чтения pdf-файлов: Adobe Acrobat Reader, Foxit Reader.

СОДЕРЖАНИЕ

Введение	4
Краткие теоретические сведения о системах обнаружения вторжений на основе анализа сетевого трафика	7
Лабораторная работа № 1. Основные режимы работы среды WEKA	14
Лабораторная работа № 2. Исследование методов обнаружения вторжений на основе решающих деревьев	24
Лабораторная работа № 3. Исследование методов обнаружения вторжений на основе моделей классификаторов	30
Лабораторная работа № 4. Исследование методов обнаружения вторжений на основе байесовской классификации	33
Литература	37

Введение

Система обнаружения вторжений (COB) – программное или аппаратное средство, предназначенное для выявления фактов неавторизованного доступа в компьютерную систему или сеть либо несанкционированного управления ими в основном через Интернет. Соответствующий английский термин – Intrusion Detection System (IDS). Системы обнаружения вторжений обеспечивают дополнительный уровень защиты компьютерных систем.

Системы обнаружения вторжений используются для обнаружения некоторых типов вредоносной активности, которая может нарушить безопасность компьютерной системы. К такой активности относятся сетевые атаки против уязвимых сервисов, атаки, направленные на повышение привилегий, неавторизованный доступ к важным файлам, а также действия вредоносного программного обеспечения (компьютерных вирусов, троянов и червей).

Обычно архитектура COB включает:

- сенсорную подсистему, предназначенную для сбора событий, связанных с безопасностью защищаемой системы;
- подсистему анализа, предназначенную для выявления атак и подозрительных действий на основе данных сенсоров;
- хранилище, обеспечивающее накопление первичных событий и результатов анализа;
- консоль управления, позволяющая конфигурировать COB, наблюдать за состоянием защищаемой системы и COB, просматривать выявленные подсистемой анализа инциденты.

Существует несколько способов классификации COB в зависимости от типа и расположения сенсоров, а также методов, используемых подсистемой анализа для выявления подозрительной активности. Во многих простых COB все компоненты реализованы в виде одного модуля или устройства.

В сетевой COB, сенсоры расположены на важных для наблюдения точках сети, часто в демилитаризованной зоне, или на границе сети. Сенсор перехватывает весь сетевой трафик и анализирует содержимое каждого пакета на наличие вредоносных компонентов. Протокольные COB используются для отслеживания трафика, нарушающего правила определенных протоколов либо синтаксис языка (например, SQL). В хостовых COB сенсор обычно является программным агентом, который ведет наблюдение за активностью хоста, на который установлен. Также существуют гибридные версии перечисленных видов COB.

Сетевая COB (Network-based IDS, NIDS) отслеживает вторжения, проверяя сетевой трафик и ведет наблюдение за несколькими хостами. Сетевая система обнаружения вторжений получает доступ к сетевому трафику, подключаясь к хабу или свитчу, настроенному на зеркалирование портов, либо сетевое TAP устройство. Примером сетевой COB является Snort.

Основанная на протоколе COB (Protocol-based IDS, PIDS) представляет собой систему (либо агента), которая отслеживает и анализирует коммуникационные протоколы со связанными системами или пользователями. Для веб-сервера подобная COB обычно ведет наблюдение за HTTP и HTTPS протоколами. При использовании HTTPS COB должна располагаться на таком интерфейсе, чтобы просматривать HTTPS пакеты ещё до их шифрования и отправки в сеть.

Основанная на прикладных протоколах COB (Application Protocol-based IDS, APIDS) – это система (или агент), которая ведет наблюдение и анализ данных, передаваемых с использованием специфичных для определенных приложений протоколов. Например, на веб-сервере с SQL базой данных COB будет отслеживать содержимое SQL команд, передаваемых на сервер.

Узловая COB (Host-based IDS, HIDS) – система (или агент), расположенная на хосте, отслеживающая вторжения, используя анализ системных вызовов, логов приложений, модификаций файлов (исполняемых, файлов паролей, системных баз данных), состояния хоста и прочих источников. Примером является система OSSEC.

Гибридная СОВ совмещает два и более подходов к разработке СОВ. Данные от агентов на хостах комбинируются с сетевой информацией для создания наиболее полного представления о безопасности сети. В качестве примера гибридной СОВ можно привести Prelude.

В пассивной СОВ при обнаружении нарушения безопасности информация о нарушении записывается в лог приложения, а также сигналы опасности отправляются на консоль и/или администратору системы по определенному каналу связи. В активной системе, также известной как Система Предотвращения Вторжений (IPS – Intrusion Prevention system (англ.)), СОВ ведет ответные действия на нарушение, сбрасывая соединение или перенастраивая межсетевой экран для блокирования трафика от злоумышленника. Ответные действия могут проводиться автоматически либо по команде оператора.

Хотя и СОВ, и межсетевой экран относятся к средствам обеспечения информационной безопасности, межсетевой экран отличается тем, что ограничивает поступление на хост или подсеть определенных видов трафика для предотвращения вторжений и не отслеживает вторжения, происходящие внутри сети. СОВ, напротив, пропускает трафик, анализируя его и сигнализируя при обнаружении подозрительной активности. Обнаружение нарушения безопасности проводится обычно с использованием эвристических правил и анализа сигнатур известных компьютерных атак.

Обнаружение аномалий – динамический метод работы антивирусов, хостовых и сетевых систем обнаружения вторжений. Программа, использующая этот метод, наблюдает определённые действия (работу программы/процесса, сетевой трафик, работу пользователя), следя за возможными необычными и подозрительными событиями или тенденциями.

Антивирусы, использующие метод обнаружения подозрительного поведения программ, не пытаются идентифицировать известные вирусы. Вместо этого они прослеживают поведение всех программ. Это помогает исключить опасность полиморфизма вирусов. Если программа пытается записать какие-то данные в исполняемый файл (exe-файл), программа-антивирус может пометить этот файл, предупредить пользователя и спросить, что следует сделать.

В отличие от метода соответствия определению вируса в словаре, метод подозрительного поведения даёт защиту от совершенно новых вирусов и сетевых атак, которых ещё нет ни в одной базе вирусов или атак. Однако программы, построенные на этом методе, могут выдавать также большое количество ошибочных предупреждений, что делает пользователя маловосприимчивым к предупреждениям.

Поскольку реальные данные для исследования не всегда доступны, предлагается использовать репозитории, сформированные научным сообществом, в частности UCI Machine Learning Repository, созданный для нужд исследователей в области машинного обучения в Калифорнийском университете Ирвина и содержащий 190 наборов данных по разным областям физики, техники, биологии, медицины, социологии, бизнеса и др.

Предварительные требования к знаниям, умениям и навыкам студентов. Студенты должны владеть основными понятиями из курса дискретной математики: множество, отображение, бинарное отношение, свойства бинарных отношений, частичный порядок, функция, исчисление высказываний и предикатов первого порядка, граф и алгоритм. Знания из курса линейной алгебры включают вычисления с матрицами. Знания из курса теории вероятностей предполагают предварительное знакомство студентов с понятием вероятности, алгеброй событий, независимости событий и теоремой Байеса. Дополнительным требованием является знакомство с понятием информационной энтропии Шеннона.

Основные знания, необходимые для свободного выполнения практикума, получены студентом в рамках лекций соответствующих курсов, тем не менее для каждой лабораторной работы приводится необходимый теоретический минимум. В перечень основных знаний, активно используемых в курсе, входят:

- добыча данных (data mining) и машинное обучение (machine learning) как области современного анализа данных;
- задачи предварительной обработки данных – очистка, шкалирование, дискретизация и другие методы классификации, кластеризации и прогнозирования;

- методы поиска ассоциаций;
- оценка качества результатов анализа данных.

Студенты должны:

- обладать навыками установки и настройки свободного ПО для анализа данных (Weka, и т.д.);
- уметь загружать учебные и исследовательские наборы данных из открытых репозиторий, например, UCI и FIMI и т.п.;
- уметь работать с наборами данных и программным обеспечением. Особое внимание уделяется таким аспектам, как: умение выбрать метод анализа данных в соответствии с поставленной целью, характером задачи и данных;
- понимать математические модели, лежащие в основе методов, описанных в базовых терминах теории множеств, упорядоченных структур, прикладной алгебры и т.п.;
- формулировать и выполнить модельные расчеты, поясняющие суть конкретного метода;
- подготовить учебные (аналитические) отчеты, представляющие собой мини-исследование по применению конкретной модели, метода и данных, с результатами экспериментов, промежуточными отчетами и выводами (фактически протокол выполнения лабораторной работы);
- проявить поисковые умения, направленные на исследование актуальной проблемы или задачи, знания дополнительной научной и учебной литературы, в том числе на английском языке, изучение нового ПО (не описанного в текстах практикума); умение правильно интерпретировать полученные результаты.

Обычно в процессе интеллектуального анализа данных выделяют следующие этапы:

1. Формулировка основных задач анализа в результате исследования предметной области.
2. Процесс сбора данных.
3. Процесс предварительной обработки собранных данных:
 - а) Преобразование данных к форме, которая возможна для анализа.
 - б) Исключение случайных "шумов" и противоречий из полученных данных, т.е. её очистка.
4. Выбор оптимального множества признаков, используемых в методах анализа данных.
5. Настройка и тестирование модели.
6. Анализ данных с помощью алгоритмов интеллектуального анализа.
7. Визуализация и интерпретация полученных результатов.
8. Применение полученных новых знаний.

Рассмотрим реализацию этих этапов средствами программной системы WEKA.

Краткие теоретические сведения о системах обнаружения вторжений на основе анализа сетевого трафика

В этом разделе представлено описание основных характеристик, используемых, в частности, при оценке состояния сетевого трафика. Для учебных целей данные взяты с баз данных KDD Cup 99/DARPA 98/DARPA 99.

Состояния сетевого трафика описывается набором признаков, которое может быть нормальным или аномальным. Аномальное состояние может быть вызвано воздействием компьютерных атак. Атаки делятся на четыре основные категории:

Denial of Service (DoS) - отказ в обслуживании, создание таких условий, при которых легальные пользователи системы не могут получить доступ к предоставляемым системным ресурсам (серверам);

Remote to Local (R2L) - несанкционированный доступ с удаленного компьютера, например, угадан пароль;

User to Root (U2R) - несанкционированный доступ к правам администратора (root), например, различные «переполнения буфера» атак.

Probing (Probe) – наблюдение или слежение, например, сканирование портов.

В отличие от большинства DOS и зондирующих атак, нет последовательных паттернов, которые часто встречаются в записях атак R2L и U2R. Это связано с тем, что атаки DOS и зондирования включают очень много соединений с некоторыми хостами за очень короткий промежуток времени, но атаки R2L и U2R встроены в части данных пакетов и обычно включают только одно соединение.

Некоторые атаки **Probe** проверяют хосты (или порты), используя гораздо больший интервал времени, чем две секунды, например один раз в минуту. Поэтому записи соединений также сортировались по хост-хосту, а функции были созданы с использованием окна из 100 соединений с одним и тем же хостом вместо временного окна. Это дает набор, так называемых функций трафика на основе хоста.

Типы атак:

dos	Back, land, Neptune, pod, smurf, teardrop
u2r	buffer_overflow, loadmodule, perl, rootkit
r2l	ftp_write, guess_passwd, imap, multihop, phf, spy, warezclient, warezmaster
probe	Ipsweep, nmap, portsweep, satan

Основные признаки отдельных TCP-соединений:

Имя параметра	Описание	Пример
duration	Время продолжительности подключения	0
protocol_type	Протокол, используемый при подключении	TCP
service	Сетевая служба, используемая подключением.	ftp_data
src_bytes	Количество отправленных байт за одно соединение	491
dst_bytes	Количество принятых байт за одно соединение	0
flag	Статус соединения – нормальное или с ошибкой	SF
land	Если ip-адреса хоста источника и назначения равны, и аналогичная ситуация с портами, то параметр принимает значение 1, иначе 0.	0
wrong_fragment	Общее число неверных фрагментов за это подключение	0
urgent	Количество urgent-пакетов в этом подключении	0

Признаки контента в соединении:

Имя параметра	Описание	Пример
hot	Количество hot-индикаторов, например таких как: вход в системные директории, создание программ, выполнение программ.	0
num_failed_logins	Количество неудачных попыток входа	0
logged_in	Логин статус. 1 – если успешно вошли в систему, иначе 0	0
num_compromised	Число скомпрометированных состояний	0

root_shell	Число скомпрометированных состояний	0
su_attempted	1, если su root-права получены, иначе 0	0
num_root	Число root-доступов	0
num_file_creations	Число операций по созданию файлов во время соединения	0
num_shells	Число вызовов shell-оболочки	0
num_access_files	Число операций по получению контроля доступа к файлам	0
num_outbound_cmds	Число исходящих команд в FTP-сессии	0
is_hot_login	1, если логин принадлежит hot-листу т.е. если является root или администратором, иначе 0	0
is_guest_login	1, если логин является гостевым, иначе 0	0

Признаки трафика, рассчитанные с использованием двухсекундного окна времени:

feature name	description	type
count	Количество подключений к одному и тому же хосту назначения за последние две секунды	2
error_rate	Процент соединений с хостом из count с SYN-ошибками	0
error_rate	Процент соединений с хостом из count с REJ-ошибками	0
same_srv_rate	Процент соединений с хостом из count использующих одни и те же службы	1
diff_srv_rate	Процент соединений с хостом используя разные службы	0
srv_count	Число соединений с одной и той же службой за последние две секунды.	2
srv_error_rate	Процент соединений с SYN-ошибками при соединении по службе из srv_count	0
srv_error_rate	Процент соединений с REJ-ошибками при соединении по службе из srv_count	0
srv_diff_host_rate	Процент соединений с разными хостами при соединении по службе из srv_count	0

Статистические характеристики работы классификатора.

При исследовании (обработке и анализе) статистических данных система WEKA выдает в качестве результата ряд характеристик, которые представлены в иллюстрации ниже:

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	1,000	0,001	1,000	1,000	1,000	0,999	1,000	1,000	DoS.
	1,000	0,000	0,998	1,000	0,999	0,999	1,000	1,000	normal.
	0,972	0,000	1,000	0,972	0,986	0,986	1,000	0,999	Probe.
	0,400	0,000	1,000	0,400	0,571	0,632	0,898	0,481	R2L.
Weighted Avg.	1,000	0,001	1,000	1,000	0,999	0,999	1,000	1,000	

где

Total_class_samples – число векторов в тестовой выборке i-го класса;

TP (True Positive) – количество правильно классифицированных аномалий;

TN (True Negative) – количество правильно идентифицированных системой образцов нормального трафика;

FN (False Negative) – количество образцов нормального трафика, определенных системой как аномальный;

FP (False Positive) – количество аномальных образцов сетевого трафика, определенных системой как нормальные.

Для расчета характеристик системы обнаружения вторжений (COB) используются следующие соотношения:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$False\ Positive = \frac{FP}{FP + TN}$$

$$True\ Positive = \frac{TP}{Total_class_samples}$$

Значение **Precision** равно проценту верно классифицированных объектов из объектов, отнесённых алгоритмом к рассматриваемому классу (отношение диагонального элемента к сумме элементов столбца).

Значение **F-Measure** вычисляется по формуле

$$F\text{-Measure} = 2 * \text{Precision} * \text{Recall} / (\text{Precision} + \text{Recall})$$

(т.е. это среднее гармоническое Precision и Recall).

Accuracy Metric. Эту метрику можно назвать базовой. Она измеряет количество верно классифицированных объектов относительно общего количества всех объектов. Имейте в виду, что ассигасу имеет некоторые недостатки: она не идеальна для несбалансированных классов, где может быть много экземпляров одного класса и мало другого.

Recall/Sensitivity Metric. Сколько объектов наша модель смогла правильно классифицировать с позитивной меткой из всего множества позитивных.

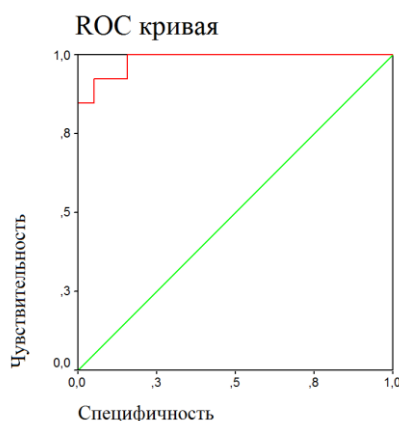
В результате применения метода выводится матрица ошибок размера $l \times l$, где l - число классов, ij -й элемент матрицы равен числу объектов из i -го класса, которые были отнесены к j -му. Число верно классифицированных объектов равно сумме элементов, стоящих на главной диагонали.

=== Confusion Matrix ===

	a	b	c	d	<-- classified as
22725	3	0	0		a = DoS.
1	5776	0	0		b = normal.
4	3	239	0		c = Probe.
0	3	0	2		d = R2L.

В качестве дополнительной характеристики результатов анализа, может быть построена **ROC-кривая** (Receiver operating characteristic, операционная характеристика приёмника) – график, позволяющий оценить качество бинарной классификации, отображает зависимость доли верных положительных классификаций от доли ложных положительных классификаций при варьировании порога решающего правила. Также известна как кривая ошибок. Анализ классификаций с применением ROC-кривых называется ROC-анализом.

Количественную интерпретацию ROC даёт показатель AUC (area under ROC curve) – площадь, ограниченная ROC-кривой и осью доли ложных положительных классификаций. Чем выше показатель AUC, тем качественнее классификатор, при этом значение 0,5 демонстрирует непригодность выбранного метода классификации (соответствует случайному гаданию).



Энтропия системы.

В статистическом смысле мы можем рассматривать эту величину как меру неопределенности системы. Рассмотрим некоторую систему X , которая может принимать конечное множество состояний: $x_1, \dots, x_n$ с вероятностями $p_1, \dots, p_n$, где p_i – вероятность того, что система X примет состояние x_i . Отсюда следует, что X является полной системой событий, следовательно, сумма вероятностей всех состояний равна единице:

$$\sum_{i=1}^n p_i = 1.$$

Энтропией системы X называется величина:

$$H(X) = - \sum_{i=1}^n p_i \log_2 p_i$$

Рассмотрим более частный случай. Предположим, что имеется множество A из n элементов, m из которых обладают некоторым свойством S . Тогда энтропия множества A по отношению к свойству S – это:

$$H(A, S) = -\frac{m}{n} \log_2 \frac{m}{n} - \frac{n-m}{n} \log_2 \frac{n-m}{n}$$

Если же свойство S не бинарное, а может принимать s различных значений, каждое из которых реализуется в m_i случаях, то энтропия обобщается естественным образом:

$$H(A, S) = - \sum_{i=1}^s \frac{m_i}{n} \log \frac{m_i}{n}$$

Количеством информации, содержащемся в отдельном сообщении о том, что система находится в состоянии x_i , будем называть величину:

$$I(x_i) = \log_2 \frac{1}{p_i} = -\log_2 p_i$$

Очевидно, что в результате получения сведений, неопределенность (т.е. энтропия) системы X может быть уменьшена. Чем больше объем получаемых сведений и чем они более содержательны, тем больше будет информация о системе и тем менее неопределенным будет ее состояние. Поэтому логично количество информации измерять уменьшением энтропии той системы, для уточнения состояния которой предназначены сведения.

Далее представлена идея основного принципа выбора такого атрибута разбиения, который дает наибольший прирост информации. Предположим, что множество A элементов, некоторые из которых обладают свойством S , классифицировано посредством атрибута Q , имеющего q возможных значений. Тогда прирост информации определяется как:

$$Gain(A, Q) = H(A, S) - \sum_{i=1}^q \frac{|A_i|}{|A|} H(A_i, S),$$

где H является энтропией, A_i – множество элементов A , на котором атрибут Q имеет значение i . Следовательно, при выборе атрибута разбиения необходимо выбирать тот, у которого значение $Gain$ наибольшее.

Ключом к построению обучающего классификатора является определение наименьшего необходимого набора признаков, требуемого для достижения наилучших показателей – процесс, известный как выделение признаков. Качество признаков является критическим для выполнения обучающего алгоритма. Использование неподходящих или избыточных признаков часто приводит к негативным последствиям для точности большинства обучающих алгоритмов. Также это может сделать более дорогой систему в вычислительном смысле. Следовательно, желательно выбрать подмножество признаков, которое является небольшим, но все же, сохраняет существенную и полезную информацию об интересующих классах.

Величина, на которую энтропия класса уменьшается, отражает дополнительную информацию о классе, обеспеченную атрибутом и называется информационным выигрышем. Каждому атрибут A_i назначается оценка, основанная на информационном выигрыше между самим атрибутом и классом. Наиболее известны следующие алгоритмы:

- **Information Gain Attribute Evaluator**

Оценка признака осуществляется по H , где H – энтропия. Этот метод является одним из самых популярных методов оценки признаков. Главный недостаток этого метода – он не использует информацию о зависимости признаков между собой. Метод выбирает признаки, на основе обработки большого числа значений. Оценка признака осуществляется по следующей формуле:

$$Info\ Gain\ (Class, Attribute) = H(Class) - H(Class|Attribute),$$

- **Gain Ratio Attribute Evaluator**

Значение Gain R должно быть большим, когда данные равномерно распределены и маленьким, когда все данные принадлежат одной ветви. Этот метод также принимает во внимание количество и размер ветвей при выборе атрибута. Gain Ratio Attribute Evaluator является модифицированной версией Information Gain Attribute Evaluator и использует специальные данные – количество информации, необходимое для отнесения признака к определенной ветви дерева. Одной из основных формул при оценке признаков является:

$$Gain\ R(Class, Attribute) = \frac{(H(Class) - H(Class|Attribute))}{H(Attribute)}$$

- **Symmetrical Uncertainty**

Оценка признака осуществляется по формуле:

$$SU = 2 \times \frac{IG}{H(Y) + H(X)},$$

где IG – Info Gain, H энтропия. Величина $SU = 0$ указывает на то, что X и Y являются некоррелируемыми.

Другой класс алгоритмов базируется на оценке корреляции признаков между собой.

- **Метод выбора признаков, основанный на корреляции (A Correlation-based Feature Selector, CFS)**

Этот метод представляет собой простой алгоритм фильтрации, который ранжирует подмножества признаков. Основой алгоритма CFS является эвристика для оценки ценности или качества подмножества признаков. Эта эвристика принимает во внимание полезность отдельных признаков для предсказания метки класса в соответствии с уровнем корреляции внутри них. Предположение, на котором основывается эвристика, звучит следующим образом: хорошие подмножества признаков содержат признаки сильно коррелирующие с классом, однако некоррелирующие друг с другом.

- **Correlation Attribute Evaluation**

В данном методе корреляция определяет зависимость признаков друг от друга. Пусть N является числом кортежей, a_i и b_i являются соответствующими значениями A и B в кортеже i , и являются средними значениями A и B , σ_A и σ_B – соответствующие стандартные отклонения A и B . Значение r_A , r_B лежит в диапазоне от -1 и до 1.

Если A и B положительно коррелируют, то r_A , r_B принимает значение 1, если A и B отрицательно коррелируют, то r_A , r_B принимает значение -1, если A и B полностью независимы, то r_A , r_B равны нулю.

В данном методе определяется зависимость признаков друг от друга

$$r_{A,B} = \frac{\sum_{i=1}^N (a_i - \bar{A})(b_i - \bar{B})}{N\sigma_A\sigma_B} = \frac{\sum_{i=1}^N (a_i b_i) - N\bar{A}\bar{B}}{N\sigma_A\sigma_B},$$

Наконец, одним из самых простых алгоритмов является

• OneR Attribute Evaluator

Данный метод использует **OneR** классификатор. **OneR** представляет из себя простой и очень эффективный алгоритм сжатия данных, который часто используется в приложениях интеллектуального анализа данных. **OneR** – это простой, но точный алгоритм классификации, который генерирует одно правило для каждого признака данных, затем выбирается правило с наименьшей общей ошибкой как свое «одно правило».

Алгоритм 1R.

Для каждого атрибута A:

Для каждого значения V этого атрибута создат правило:

1. Подсчитать, как часто появляется каждый класс.
2. Найти наиболее частый класс, c .
3. Построить правило «если $A = V$, то $C = c$ ».
4. Вычислить частоту ошибок этого правила.
5. Выберите атрибут, правила которого дают наименьшую частоту ошибок.

Пример.

1. Откройте «Weka».
2. Нажмите «Открыть файл ...» и загрузите набор данных (weather_nominal.csv).
3. Перейдите на вкладку «Классификация» и выберите «OneR».
4. Выберите «Play Golf» в качестве цели из списка и нажмите «Пуск».
5. Посмотрите на панель «Выходной сигнал классификатора» для результата.

План проведения исследования (эксперимента)

1. Загрузить заданную в соответствии с вариантом задания базу данных, содержащую вектора, описывающие состояние трафика.
2. Выбрать признаки, которые будут использоваться для классификации, с помощью методов (BestFirst, GreedyStepwise, Ranker).
 - 2.1. GreedyStepwise – Поиск восхождением к вершине – это техника математической оптимизации, принадлежащая семейству алгоритмов локального поиска. Алгоритм реализуется методом итерации, который начинается с произвольного решения, а затем пытается найти лучшее решение путём пошагового изменения одного из элементов. Если результат улучшается, делается приращение для получения нового решения и этот процесс повторяется, пока не достигнем момента, в котором улучшение не возможно.
 - 2.2. BestFirst – лучевой поиск - это эвристический алгоритм поиска, исследует граф, расширяя перспективные узлы в ограниченном их наборе. Лучевой поиск является оптимизацией так называемого «поиска первого наилучшего» (best-first), что существенно снижает его требования к необходимому объёму памяти. Best-first строит множество временных решений (состояний) по заданным эвристикам и пытается предсказать насколько близким является частное решение к целевому состоянию. В лучевом поиске только некоторая часть лучших частных решений хранится в качестве кандидатов.
 - 2.3. Ranker – определяет атрибуты по их индивидуальным оценкам. Используется совместно с методами оценки атрибутов.
3. Выбрать режим обучения классификатора в соответствии с заданием.
4. Выбрать режим тестирования классификатора в соответствии с заданием.
5. Визуализация результатов.
6. Сравнительный анализ исследуемых методов и полученных результатов.

Пример (шаблон) описания результатов эксперимента.

Дерево, построенное со всем 41 параметрам, показало общий результат обнаружения [заполнить], [заполнить] ошибок первого рода и [заполнить] ошибки второго рода. Это значение было принято как исходное. Рассмотрим результаты метода [заполнить]. Было произведено ранжирование параметров. Приведены результаты обнаружения. Изображены графики зависимости полученных результатов, от количества параметров. Наилучший ре-

зультат был показан при [заполнить] параметрах, с общим результатом [заполнить] %, [заполнить] ошибками первого рода и [заполнить] ошибками второго рода.

Наилучший результат показал метод [заполнить] с [заполнить] параметрами, [заполнить] ошибками первого рода, [заполнить] ошибками второго рода, и общим результатом [заполнить] %. Номера параметров следующие: [заполнить]. Чтобы улучшить результат, было уменьшено число наших параметров, но не по ранжиру. Это значит, что в первом эксперименте было создано [заполнить] выборок с полученными [заполнить] параметрами и в каждой забирали по одному разному параметру. С выборкой показавшей наилучший результат был проделаны аналогичные действия, только, естественно, для [заполнить] параметров. Из-за большого числа выборок и громоздкости промежуточных результатов, приведены только наилучшие показатели на каждом этапе.

Лабораторная работа № 1.

Основные режимы работы среды WEKA

Цель работы.

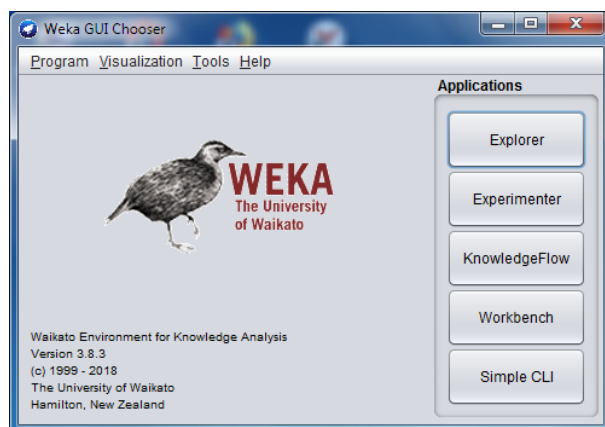
Ознакомиться с графическим интерфейсом пользователя системы WEKA, основными режимами работы, структурой входного файла и последовательностью исследования входных данных.

Краткие теоретические сведения.

Основные сведения о работе в среде WEKA

Программа WEKA написана на языке Java в университете Вайкато (Новая Зеландия), распространяется под лицензией GNU General Public License version 21 (GPLv2) и предоставляет пользователю возможность предобработки данных, решения задач классификации, регрессии, кластеризации и поиска ассоциативных правил, а также визуализации данных и результатов. Откомментированный код программы можно найти в файле weka-src.jar.

Для начала работы достаточно запустить файл RunWeka.bat. Появится окно Weka GUI Chooser, в котором будет предложено выбрать один из четырёх модулей программы.



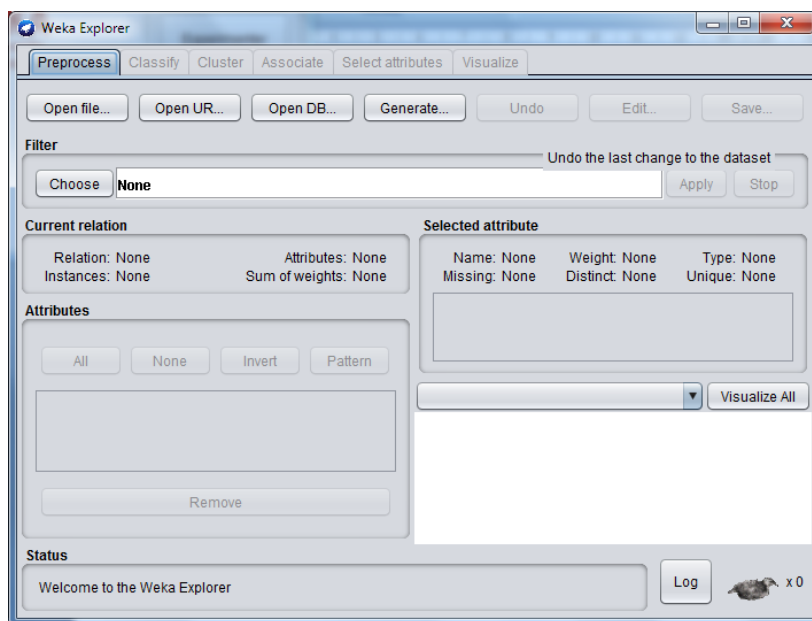
Вкладка Explorer. Это основной модуль программы, который позволяет загрузить и предобработать данные (вкладка Preprocess), решить задачу классификации (Classify), кластеризации (Cluster), поиска ассоциаций (Associate), селекции признаков (Select Attributes) и визуализации (Visualize).

Каждая задача имеет свою вкладку в общем окне. Сначала доступна только вкладка Preprocess, поскольку для выполнения остальных задач нужны данные. Отметим, что последовательность вкладок не всегда соответствует этапам решения задачи. Рассмотрим работу на соответствующих вкладках.

Вкладка Preprocess. Изначально вверху окна активны четыре кнопки, (Open file), (Open URL), (Open DB) и сгенерировать модельные данные (Generate). Чаще всего приходится пользоваться данными из файла. WEKA понимает форматы (arff, CSV, C4.5, бинарные, libsvm). Основной формат - arff. В каталоге data можно посмотреть примеры arff-файлов, которые включены в состав программы.

На верхней панели Filter можно выбрать преобразования данных. Например, нажатием кнопки Choose из иерархического списка выберем опцию Weka/filters/unsupervised/attribute/Normalize/. На панели отобразится название этого фильтра (процедура предобработки данных), щёлкнув по нему правой клавишей мыши можно изменить его параметры. Важно: некоторые фильтры имеют параметры, определяющие, к каким признакам эти фильтры применять (т.е. область применения не всегда определяется пометками на панели Attributes). Нажатие кнопки Apply запускает фильтр. В данном случае происходит нормализация: значения признака переводятся на отрезок [0,1].

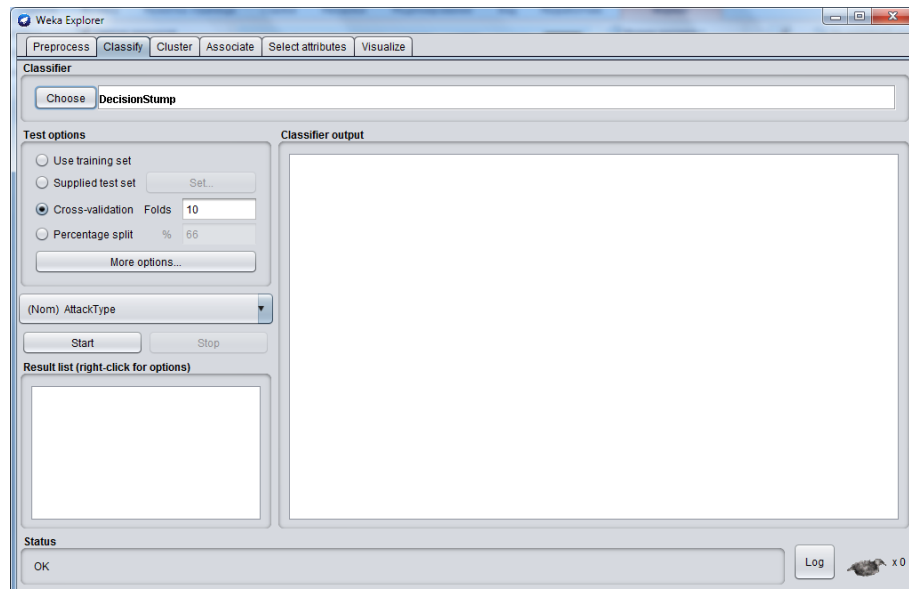
На нижней панели Status отображается статус программы и изображение птички Киви. После значка «х» отображается число запущенных процессов.



Вкладка Classify. На панели Classifier можно выбрать классификатор - кнопка Choose, например, `weka/classifiers/functions/MultilayerPerceptron` (для выбора многослойного перцептрона). Как и для фильтров, нажатие правой кнопки мыши по появившемуся на панели Classifier названию позволяет выбрать параметры классификатора. Если для многослойного перцептрона выбрать значение параметра GUI равным true, то появится возможность редактировать нейронную сеть.

Нажатие на кнопку Start запускает обучение классификатора. На самой большой панели Classifier output отображается отчёт об обучении. Сначала отображается информация о задаче и алгоритме классификации. Затем параметры настроенного классификатора. Затем результаты классификации (если включена опция More options/Output predictions).

На панели «Test options» определяется метод тестирования полученного классификатора: на обучающей выборке (use training set), на тестовой выборке из отдельного файла (supplied test set), по блокам (cross-validation), с помощью разделения исходной выборки на обучение и контроль (percentage split). При выборе некоторых опций придётся указать параметры тестирования. Например, при выборе «cross-validation» надо указать, на сколько блоков (фолдов) разбивать выборку. К примеру, если указать 10 фолдов, то выборка будет разделена на 10 равных частей, затем 9 частей будет использовано для обучения и 1 для тестирования. Процесс будет повторен 10 раз, а результаты усреднены.



Кнопка «More options» позволяет выбрать вид отчёта об обучении классификатора. Рассмотрим параметры, которые можно задать:

1. Output model. Выводится модель классификации на полной обучающей выборке, поэтому результаты могут быть просмотрены и визуализированы.
2. Output per-class stats. Точность/эффективность и истина/ложь статистики для каждого класса.
3. Output entropy evaluation measures. Оценка энтропии на критериях.
4. Output confusion matrix. Матрица ошибок предсказания классификатора.
5. Store predictions for visualization. Предсказания классификатора сгруппированы таким образом, чтобы их можно было визуализировать.
6. Output predictions. Предсказанные значения классов для тестовой выборки.
7. Output additional attributes. В случае если вместе с предсказанными значениями необходимо вывести дополнительные признаки. Например, вывести ID признак для определения неправильно классифицированных объектов
8. Cost-sensitive evaluation. Ошибка классификации определяется с учетом матрицы ценности.
9. Random seed for xval/% Split. Данный параметр используется при рандомизации данных перед их разделением.
10. Preserve order for % Split. Данный параметр отменяет рандомизацию данных перед их разделением на обучающую и тестовую выборку.
11. Output source code. Если классификатор может вывести построенную модель в виде исходного кода на языке Java, можно задать для нее имя класса.

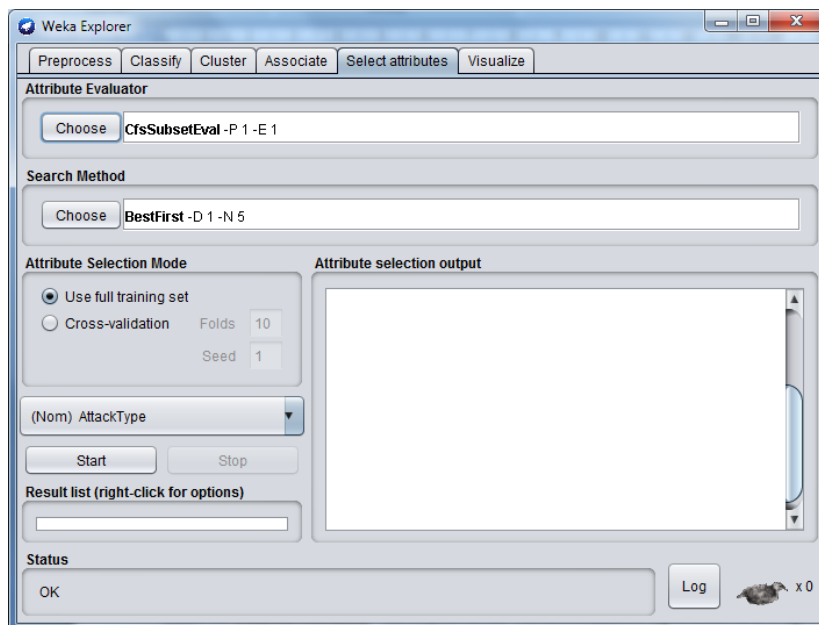
На левой нижней панели Result List выводится перечень всех запусков классификаций данных. Новый элемент в этом списке появляется после завершения настройки и тестирования классификатора (вызванных нажатием кнопки Start). В списке указывается время окончания классификации и название классификатора. Если по элементу списка щёлкнуть правой кнопкой мыши, то появится дополнительное меню, с помощью которого Вы сможете посмотреть отчёт в отдельном окне (View in separate window), загрузить и сохранить модель, т.е. тип и параметры классификатора (Load model, Save model), визуализировать ошибки классификатора (Visualize classifier errors, вкладка **Visualize**), посмотреть различные кривые отступов и ошибок.

Вкладка Select Attributes. Вкладка Select Attributes позволяет выбрать признаки, для последующей классификации (или кластеризации) объектов. Доступны панели:

- **Attribute Evaluator** – выбирается функция оценки качества признаков.

• **Search Method** – метод поиска оптимального признакового пространства (например, генетический алгоритм, случайный поиск или полный перебор). Схема работы с этими панелями аналогична схеме выбора фильтра или классификатора.

• **Attribute Selection Mode** - выбирается режим селекции: по всей выборке (use full training set) или с помощью контроля по блокам (cross-validation). Есть компонент для выбора целевого признака (класса) и кнопка Start для запуска процедуры отбора признаков, результаты которой отобразятся потом на самой большой панели **Attribute selection output**.



Вкладки Cluster и Associate. Работа на вкладках кластеризации и поиска ассоциаций аналогична работе при классификации, а сами вкладки имеют аналогичную структуру (даже более простую).

Пример ввода и подготовки данных для анализа в WEKA

На вкладке **Preprocess** выбираем файл для загрузки. Основной формат – arff. ARFF-файл является ASCII текстовым файлом, который описывает список объектов с общими атрибутами. Структурно такой файл разделяется на две части: заголовок и данные.

В заголовке описывается имя данных и их метаданные (имена атрибутов и их типы). Например,

```
% комментарий
@RELATION myproblem
@ATTRIBUTE firstfeature REAL
@ATTRIBUTE secondfeature REAL
@ATTRIBUTE noise REAL
@ATTRIBUTE class {A,B}
```

Во второй части представлены сами данные. Например,

```
@ DATA
0,0,1.1,A
0,0,3.2,A
0,0,4.7,A
0,0,3.1,A
0,1,2.2,B
0,1,4.3,B
1,0,1.1,B
1,0,2.5,B
1,1,3.7,A
1,1,4.2,A
```

1,1,3.9,A

1,1,5.1,A

Заголовок содержит информацию об имени файла и метаданные о представленных в нем данных. Имя описывается в следующем формате

@relation< имя>

Именем может быть любая последовательность символов. Если содержит пробелы, то оно должно быть взято в кавычки. Например,

@relation weather

@relation "weather nominal"

Метаданные описывают атрибуты представленных в файле данных. Информация о каждом атрибуте записывается в отдельной строке и включает имя атрибута и его тип. Очевидно, что все имена должны быть уникальными. Порядок их описания должен совпадать с порядком колонок в описании самих данных. Общий формат описания атрибута следующий:

@attribute< имя атрибута> <тип атрибута>

Например,

@attribute outlook {sunny, overcast, rainy}

@attribute temperature real

Имя атрибута должно начинаться с символа @. В случае если в имени содержатся пробелы, оно должно быть взято в кавычки.

Поле< тип> может иметь одно из следующих значений:

- real
- integer
- <категория>
- string
- date [<формат даты>]

Категориальные типы описываются перечнем категорий (возможных значений).

Например:

@attribute outlook {sunny, overcast, rainy}

При описании даты можно указать формат, в котором она записывается (Например, "yyyy-MM-dd").

Данные представляются в ARFF формате в виде списка значений атрибутов объектов после тега @data. Каждая строка списка отвечает одному объекту. Каждая колонка отвечает атрибуту, описанному в заголовке. Причем порядок колонок должен совпадать с порядком описания атрибутов. Например:

@data

overcast,75,55,false,will_play

sunny,85,85,false,will_play

sunny,80,90,true,may_play

Часто в терминологии data mining такие строки называют векторами.

Данные могут содержать пропущенные (неизвестные) значения. В ARFF они представляются символом «?», Например:

@data

4.4,?,1.5,?,Iris-setosa

Строковые данные, в случае, если они содержат разделяющие символы, должны браться в кавычки. Например,

@relation LCCvsLCSH

@attribute LCC string

@attribute LCSH string

@data

AG5, 'Encyclopedias and dictionaries.;Twentieth century.'

AS262, 'Science -- Soviet Union -- History.'

AE5, 'Encyclopedias and dictionaries.'

AS281, 'Astronomy, Assyro-Babylonian.;Moon -- Phases.'

AS281, 'Astronomy, Assyro-Babylonian.;Moon -- Tables.'

Даты также должны браться в кавычки.

@relation Timestamps

@attribute timestamp DATE "yyyy-MM-dd HH:mm:ss"

@data

"2001-04-03 12:12:12"

"2001-05-03 12:59:55"

Как только данные были загружены, на панели предварительной обработки информации появляется информация о данных.

Нажатие на кнопку «Edit» позволяет редактировать исходные данные, появляется окно «Viewer».

Область «**Current relation**» содержит:

- **Relation.** Название отношения, которое было указано в файле. Применение фильтров может менять название.

- **Instances.** Количество экземпляров (записей/объектов) в данных

- **Attributes.** Количество признаков (атрибутов) в данных. Содержит три кнопки и список признаков, который содержит три колонки:

- **No.** Номер, идентифицирующий признак в том порядке, в котором они определены в файле данных.

- **Selection tick boxes.** Позволяет выбрать признак для дальнейших операций.

- **Name.** Имя признака, обозначенное в файле данных.

При выборе одного из признаков (подсвечивается в списке синим цветом), изменяется содержимое правой панели «Selected attribute». Содержимое этой панели характеризует выбранный признак с именем Name:

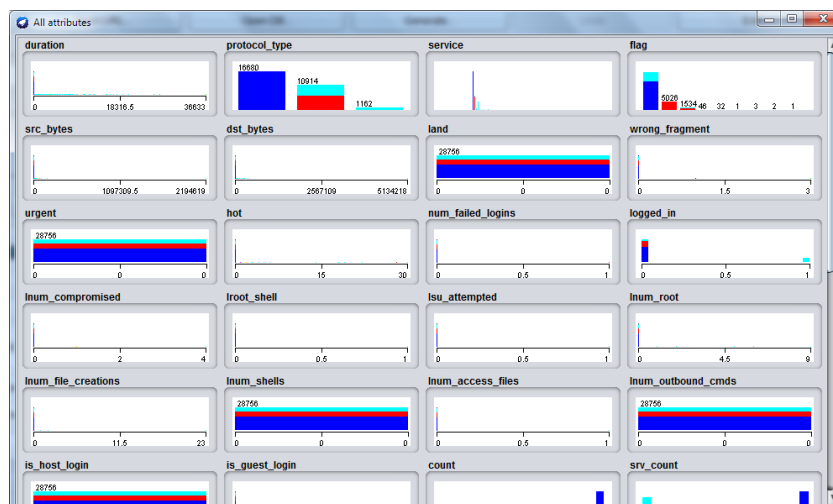
Type. Тип признака Nominal либо Numeric. Если признак имеет тип nominal, список содержит все возможные значения признака с количеством их появления. Если признак относится к numeric, то список представляет следующую статистическую информацию: минимум, максимум, мат ожидание и стандартное отклонение.

- **Missing.** Количество и процент экземпляров, для которых значение этого признака потеряно (не определено).

- **Distinct.** Количество различных значений признака.

- **Unique.** Количество и процент экземпляров, для которых значение признака имеет уникальное значение (такого значения нет ни у одного другого экземпляра)

Ниже представлена цветная гистограмма значений признака сетевого трафика, которая в зависимости от признака, выбранного в качестве определяющего класс, показывает распределение значений признаков. Можно посмотреть гистограммы всех признаков нажатием кнопки «Visualize all».



Выше списка всех признаков находятся кнопки, позволяющие выбирать признаки (при помощи чекбоксов). Выбранные таким образом признаки можно удалить кнопкой «Remove».

Замечание. Некоторые фильтры работают по-разному в ситуациях, когда признак класса был задан и когда класс не был задан (при помощи выпадающего списка над гистограммой). Так, например, "supervised filters" требуют, чтоб класс был задан, в то же время, когда "unsupervised attribute filters" не принимают данный параметр во внимание.

Пример выполнения работы.

Приведем описание последовательности обработки данных заданным методом, среди которых могут быть методы: наивный байесовский, J4.8 (модификация C4.5), ID3, 1R, SVM (в среде Weka он называется SMO). В случае невозможности применить метод к данным необходимо воспользоваться фильтрами.

Пусть в соответствии с заданием выбран пример `airline.arff`.

Шаг 1. Загрузить файл.

Нажав на кнопку «Open file» вкладки Preprocess в каталоге data можно посмотреть примеры arff-файлов, которые включены в состав программы.

Шаг 2. Просмотр признаков, подлежащих обработке.

В этом режиме доступны следующие элементы интерфейса:

- **Attributes** – выбор признаков,
- **Selected attribute** – отображается гистограмма значений признака. В верхней части панели Selected attribute отображается простая статистика: максимальное, минимальное и среднее значения признака, выборочная дисперсия, число пропусков, тип признака, число уникальных значений.
- **Visualize all** – просмотр гистограммы всех признаков. нажатием кнопки Слева от этой кнопки находится компонент выбора целевого признака: какой признак считать классом при визуализации (выбор
- **NoClass** – соответствует тому, что все объекты приписываются одному классу и для каждого признака показывается гистограмма распределения всех его значений).

Шаг 3. Редактирование признаков.

Редактирование может осуществляться как вручную, так и наложением на данные фильтра для их очищения и/или трансформации. Фильтры необходимы для модификации данных, таким образом, чтобы стало возможно применение различных методов. Так, например, некоторые методы могут работать только с перечислимым типом данных. Для этого при помощи фильтра RemoveType можно убрать из набора данные неподходящих типов.

Слева в области «Filter» находится кнопка выбора фильтра «Choose». С ее помощью можно выбрать один из предварительно определенных фильтров. После выбора название фильтра из иерархического списка появится справа от кнопки «Choose». Щелкнув по названию фильтра, вызывается диалоговое окно настройки параметров фильтра (The GenericObjectEditor Dialog Box). Все поля данного окна имеют всплывающие подсказки о назначении фильтра и его параметров настройки.

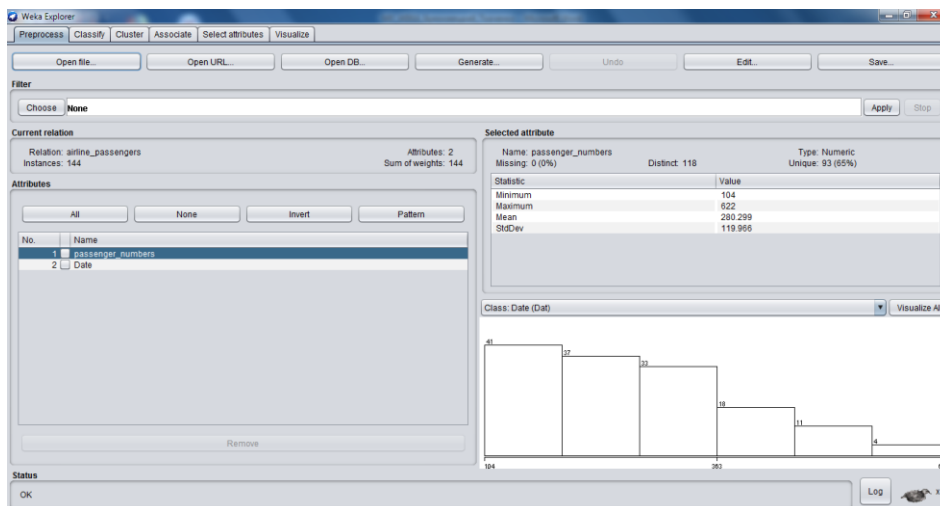
Такое же диалоговое окно используется при настройке других объектов программы WEKA, например, параметры классификаторов и кластеров.

Нажав на кнопку «Apply» в правой части панели «Filter», происходит фильтрация (предобработка) исходных данных в соответствии с выбранным алгоритмом.

Имеется возможность выполнить следующие операции над признаками:

- **Remove** приводит к удалению признаков, помеченных галочками на левой нижней панели Attributes.
- **Edit** позволяет редактировать значения признаков: появляется Viewer.
- **Save** сохраняет данные.

Ниже на рисунках приводятся результаты загрузки и визуализации данных для `airline.arff`:

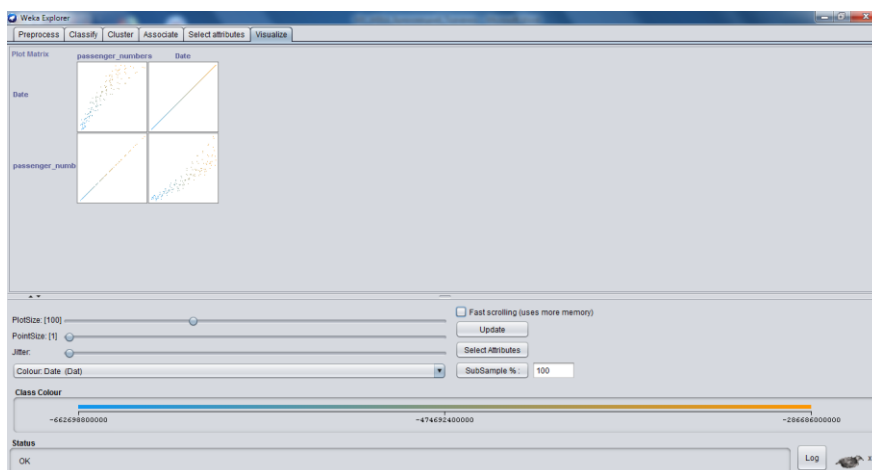


Результат загрузки файла с данными в систему

Шаг 4. Визуализация результатов обработки.

Вкладка **Visualize**. Эта вкладка позволяет визуализировать выборки (значения признаков).

- **Select Attributes** – позволяет выбрать признаки для визуализации: будут построены картинки-проекции на всевозможные пары этих признаков.
- **PlotSize** – выбор размера картинок,
- **PointSize** – размер точек, изображающих объекты,
- **Jitter** – уровень шумов, которые специально добавляются к признакам.



Пример визуализации данных

Шаг 5. Классификация в соответствии с заданием.

Во вкладке **Classify** при помощи кнопки **Choose** выбирается метод классификации. После выбора метода классификации (классификатора, classifier) необходимо выбрать метод проверки. Основным методом является кросс-проверка (cross-validation). Суть ее в том, что исходный набор данных в какой-либо пропорции разбивается на обучающее и проверочное множества. Таким образом, вычисляется ошибка. Можно также проводить проверку результатов анализа:

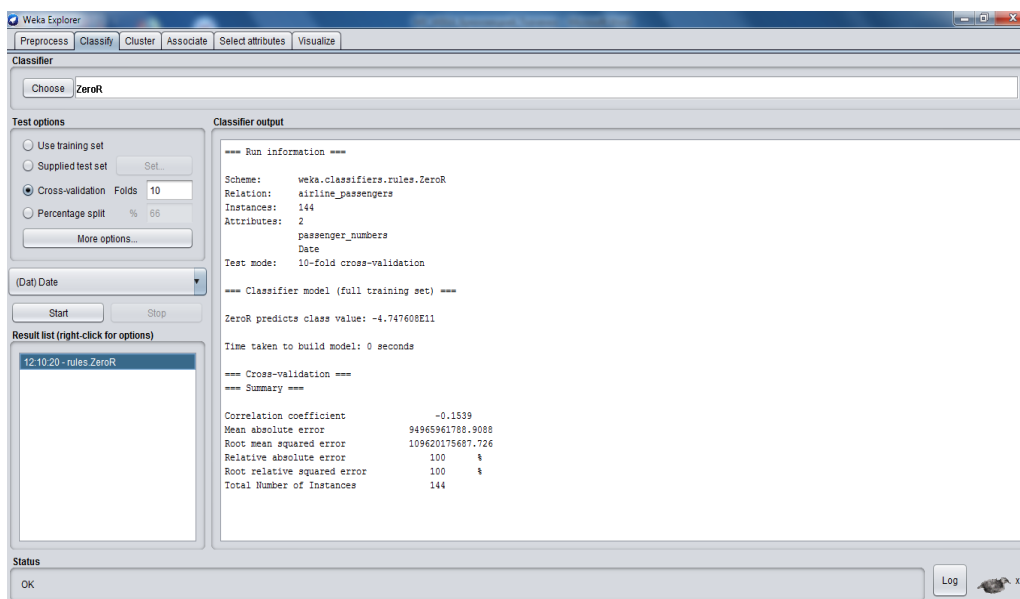
- на обучающем множестве (training set),
- на специальном тестовом множестве (supplied test set),
- на тестовой части обучающего множества (Percentage Split).

На панели **Classifier** доступны следующие режимы.

- **Choose** – выбор метода и параметров классификатора.
- **Test options** – выбор режима и параметров тестирования и классификатора.
- **More options** – позволяет выбрать вид отчёта об обучении классификатора.

Ниже панели Test options находится компонент для выбора целевого признака (что будет считаться классом в задаче).

- **Start** - запускает обучение классификатора.
- **Classifier output** – отображается отчёт об обучении.



Пример классификации

Шаг 6. Операции над результатами классификации.

• **Result List** – выводится перечень всех запусков классификаций данных. Новый элемент в этом списке появляется после завершения настройки и тестирования классификатора (вызванных нажатием кнопки Start). В списке указывается время окончания классификации и название классификатора. Если по элементу списка щёлкнуть правой кнопкой мыши, то появится дополнительное меню, с помощью которого Вы сможете посмотреть отчёт в отдельном окне (**View in separate window**), визуализировать ошибки классификатора (**Visualize classifier errors**), посмотреть различные кривые точности классификации.

Задание на лабораторную работу.

Выбрать из таблицы файлы примеров в формате .arff либо .csv (например, cpu.arff, airline.arff, labor.arff) и метод классификации:

N – порядковый номер студента в списке группы (для ЗФО – последние две цифры номера зачетной книжки).

$$N1 = N(\text{MOD } 18) + 1,$$

$$N2 = (N1 + 6)(\text{MOD } 18) + 1,$$

$$N3 = (N2 + 6)(\text{MOD } 18) + 1.$$

Таблица выбора вариантов задания

№	Файл примера	Метод
1.	cpu.arff	BayesNet
2.	cpu.arff	NaiveBayes
3.	cpu.arff	NaiveBayesMultinomialText
4.	cpu.arff	DecisionStump
5.	cpu.arff	HoeffdingTree
6.	cpu.arff	RandomTree
7.	airline.arff	BayesNet
8.	airline.arff	NaiveBayes
9.	airline.arff	NaiveBayesMultinomialText
10.	airline.arff	DecisionStump
11.	airline.arff	HoeffdingTree
12.	airline.arff	RandomTree

13.	labor.arff	BayesNet
14.	labor.arff	NaiveBayes
15.	labor.arff	NaiveBayesMultinomialText
16.	labor.arff	DecisionStump
17.	labor.arff	HoeffdingTree
18.	labor.arff	RandomTree

2. Выполнить обработку каждого из трех заданных примеров по шагам:

Шаг 1. Загрузить файл.

Шаг 2. Просмотр признаков, подлежащих обработке.

Шаг 3. Редактирование признаков.

Шаг 4. Визуализация результатов обработки.

Шаг 5. Классификация в соответствии с заданием.

Шаг 6. Операции над результатами классификации.

3. Представить краткое описание последовательности обработки.

Содержание отчета

Отчет о проделанной работе должен содержать:

Титульный лист.

1.Цель работы.

2.Постановка задачи.

3.Вариант индивидуального задания: выбранные файлы данных.

4.Описание экспериментов.

5.Результаты исследования со снимками экранов WEKA.

Выводы.

Контрольные вопросы

1. Что такое система WEKA?

2. Какие режимы работы системы WEKA Вы знаете?

3. В каких форматах можно загрузить данные в систему WEKA?

4. Как проводить классификацию данных в системе WEKA?

5. Перечислите шаги по обработке данных в системе WEKA.

Лабораторная работа № 2.

Исследование методов обнаружения вторжений на основе решающих деревьев

Цель работы.

Выполнить исследование в WEKA процесса обнаружения вторжений методами решающих деревьев на примерах, описывающих состояние трафика и взятых с баз данных KDD Cup 99/DARPA 98/DARPA 99.

Краткие теоретические сведения.

Деревья решений – это способ представления правил в иерархической, последовательной структуре, где каждому объекту соответствует единственный узел, дающий решение. Под правилом понимается логическая конструкция, представленная в виде "если ... то ...".

Пусть через $\{C_1, C_2, \dots, C_k\}$ обозначены классы (значения меток классов), тогда существуют 3 ситуации:

1. Множество T содержит один или более примеров, относящихся к одному классу C_k . Тогда дерево решений для T – это лист, определяющий класс C_k ;

2. Множество T не содержит ни одного примера, т.е. пустое множество. Тогда это снова лист, и класс, ассоциированный с листом, выбирается из другого множества отличного от T , скажем, из множества, ассоциированного с родителем;

3. Множество T содержит примеры, относящиеся к разным классам. В этом случае следует разбить множество T на некоторые подмножества. Для этого выбирается один из признаков, имеющий два и более отличных друг от друга значений $O_1, O_2, \dots, O_n$.

T разбивается на подмножества $T_1, T_2, \dots, T_n$, где каждое подмножество T_i содержит все примеры, имеющие значение O_i для выбранного признака. Это процедура будет рекурсивно продолжаться до тех пор, пока конечное множество не будет состоять из примеров, относящихся к одному и тому же классу.

Фрагмент бинарного дерева представлен на рисунке, где изображено дерево принятия решений по выдаче кредита:



В данной работе исследуются следующие методы:

Алгоритм ID3 – выбор атрибута происходит на основании прироста информации (англ. Gain). Алгоритм является одним из самых известных алгоритмов для построения деревьев принятия решений. Одной из главных причин популярности ID3 стала простота в его понимании и реализации. Общая идея алгоритма состоит в следующем:

1. Взять все неиспользованные признаки и посчитать их энтропию относительно тестовых образцов.

2. Выбрать признак, для которого энтропия минимальна (а информационная выгода соответственно максимальна).

3. Создать узел дерева, содержащий этот признак.

Сам же алгоритм работает следующим образом:

ID3 (Таблица примеров-Examples(vi), Целевой признак, Признаки)

1. Если все примеры положительны, то вернуть узел с меткой «+».

2. Если все примеры отрицательны, то вернуть узел с меткой «-».

3. Если множество признаков пустое, то вернуть узел с меткой, которая чаще других встречается в значениях целевого признака в примерах.

4. Иначе:

4.1. Пусть A – признак, который лучше всего классифицирует примеры (с максимальной информационной выгодой).

4.2. Создать корень дерева решения; признаком в корне будет являться A.

4.3. Для каждого возможного значения A выполнить:

4.3.1. Добавить новую ветвь дерева ниже корня с узлом со значением $A = v_i$

4.3.2. Выделить подмножество Examples(vi) примеров, у которых $A = v_i$.

4.3.3. Если подмножество примеров пусто, то ниже этой новой ветви добавить узел с меткой, которая больше других встречается в значениях целевого признака в примерах.

4.4. Иначе, ниже этой новой ветви добавить поддереву, вызывая рекурсивно ID3(Examples(vi), Целевой признак, Признаки)

5. Возврат корня.

C4.5 – алгоритм для построения деревьев решений. C4.5 является усовершенствованной версией алгоритма ID3. В частности, в новую версию были добавлены отсечение ветвей (англ. pruning), возможность работы с числовыми атрибутами, а также возможность построения дерева из неполной обучающей выборки, в которой отсутствуют значения некоторых атрибутов. Алгоритм C4.5 выбирает атрибут на основании нормализованного прироста информации (англ. Gain Ratio).

Для того чтобы с помощью C4.5 построить решающее дерево и применять его, данные должны удовлетворять нескольким условиям:

1. Описание атрибутов. Каждый атрибут должен иметь дискретное или числовое значение и количество атрибутов должно быть фиксированным для всех примеров.

2. Определенные классы. Каждый пример должен быть ассоциирован с конкретным классом, т.е. один из атрибутов должен быть выбран в качестве метки класса.

3. Дискретные классы. Классы должны быть дискретными, т.е. иметь конечное число значений. Каждый пример должен однозначно относиться к конкретному классу. Количество классов должно быть значительно меньше количества примеров. Для сокращения деревьев используется отсечение ветвей.

Реализации:

• **J48** – аналог алгоритма C4.5, реализованного на языке Java

• **C5.0** (для Linux) / **See5** (для Windows) – реализация на языке C.

HoeffdingTree – инкрементный алгоритм индукции дерева решений, который способен учиться на потоках данных, предполагая, что примеры генерации распределения не меняются со временем. Деревья Hoeffding используют тот факт, что небольшой выборки часто бывает достаточно, чтобы выбрать оптимальный атрибут расщепления. Эта идея поддерживается математически оценкой Хёффдинга, которая количественно оценивает число наблюдений (в нашем случае, примеры), необходимых для оценки некоторых статистических данных заданной точности (<http://weka.sourceforge.net/doc.dev/weka/classifiers/trees/HoeffdingTree.html>).

LMT – Классификатор для построения «деревьев логистической модели», которые представляют собой деревья классификации с функциями логистической регрессии у листьев. Алгоритм может обрабатывать двоичные и многоклассовые целевые переменные, число-

вые и номинальные атрибуты и отсутствующие значения (<http://weka.sourceforge.net/doc.dev/weka/classifiers/trees/LMT.html>).

RandomForest – алгоритм машинного обучения, заключающийся в использовании комитета (ансамбля) решающих деревьев. Алгоритм сочетает в себе две основные идеи: метод бэггинга Бреймана, и метод случайных подпространств. Алгоритм применяется для задач классификации, регрессии и кластеризации. Основная идея заключается в использовании большого ансамбля решающих деревьев, каждое из которых само по себе даёт очень невысокое качество классификации, но за счёт их большого количества результат получается хорошим (<http://weka.sourceforge.net/doc.dev/weka/classifiers/trees/RandomForest.html>).

RandomTree – Класс для построения дерева, который рассматривает K случайно выбранных атрибутов на каждом узле. Также есть возможность разрешить оценку вероятностей класса на основе установленного набора (backfitting – метод настройки с возвращениями совмещает многомерную линейную регрессию и одномерное сглаживание. Таким образом, нелинейная задача сводится к решению последовательности линейных задач) (<http://weka.sourceforge.net/doc.dev/weka/classifiers/trees/RandomTree.html>).

DecisionStump – модель машинного обучения, состоящая из одноуровневого дерева решений. Это деревья решений с одним внутренним узлом (корнем), который немедленно подключается к терминальным узлам (его листьям). Решение делает прогноз, основанный на значении только одной функции ввода. Иногда их также называют 1-правилами (<http://weka.sourceforge.net/doc.dev/weka/classifiers/trees/DecisionStump.html>).

Деревья решений, используемые в Data Mining, бывают двух основных типов:

- Дерево для классификации, когда предсказываемый результат является классом, к которому принадлежат данные;
- Дерево для регрессии, когда предсказываемый результат можно рассматривать как вещественное число (например, цена на дом, или продолжительность пребывания пациента в больнице).

Перечисленные методы имеют некоторые сходства (рекурсивный алгоритмы построения), а также некоторые различия, такие, как критерии выбора разбиения в каждом узле.

Некоторые методы позволяют построить более одного дерева решений (ансамбли деревьев решений –

https://www.researchgate.net/publication/278019662_Istoria_razvitia_ansamblevyh_metodov_klassifikacii_v_masinnom_obucenii):

1. Бэггинг над деревьями решений, наиболее ранний подход. Подобно тому, как усреднение нескольких наблюдений снижает оценку дисперсии данных, так и разумным способом снижения дисперсии прогноза является извлечение большого количества порций данных из генеральной совокупности, построение предсказательной модели по каждой обучающей выборке и усреднение полученных предсказаний. Если вместо отдельных обучающих выборок (которых, как правило, всегда не хватает) выполнить бутстреп и на основе сгенерированных псевдо-выборок построить B деревьев регрессии, то средний коллективный прогноз

$$f_{\text{bag}} = (f_1(x) + f_2(x) + \dots + f_B(x)) / B$$

будет обладать более низкой дисперсией. Эта процедура и называется бэггингом (сокр. от **bootstrap aggregating**).

2. Классификатор «Случайный лес» основан на бэггинге, однако в дополнение к нему случайным образом выбирает подмножество признаков в каждом узле, с целью сделать деревья более независимыми;

3. Бустинг. Идея метода заключается в итеративном процессе последовательного построения частных моделей. Каждая новая модель обучается с использованием информации об ошибках, сделанных на предыдущем этапе, а результирующая функция представляет собой линейную комбинацию всего ансамбля моделей с учетом минимизации любой штрафной функции.

4. «Вращение леса» – деревья, в которых каждое дерево решений анализируется применением метода главных компонент (PCA) на случайные подмножества входных функций.

Цель исследования заключается в поиске оптимального набора параметров, необходимых для эффективного нахождения сетевых атак. Для обучения использовался набор KDD-Train+, для тестирования – KDDTest+. Методами предварительного выбора признаков были взяты: Correlation Attribute Evaluation, Information Gain Attribute Evaluator, Gain Ratio Attribute Evaluator, OneR Attribute Evaluator, Symmetrical Uncertainty. Критерием выбора эффективного набора параметров было взято одновременное наименьшее значение ошибок первого и второго рода. При неоднозначности выбора преимущество давалось тому набору, который содержал наименьшее число ошибок второго рода, т.к. пропуск атаки важнее ложного срабатывания. Также важно снижение числа рассматриваемых параметров.

Пример выполнения работы

Пусть заданы: Dataset_Anomaly_II.csv и метод RandomTree.

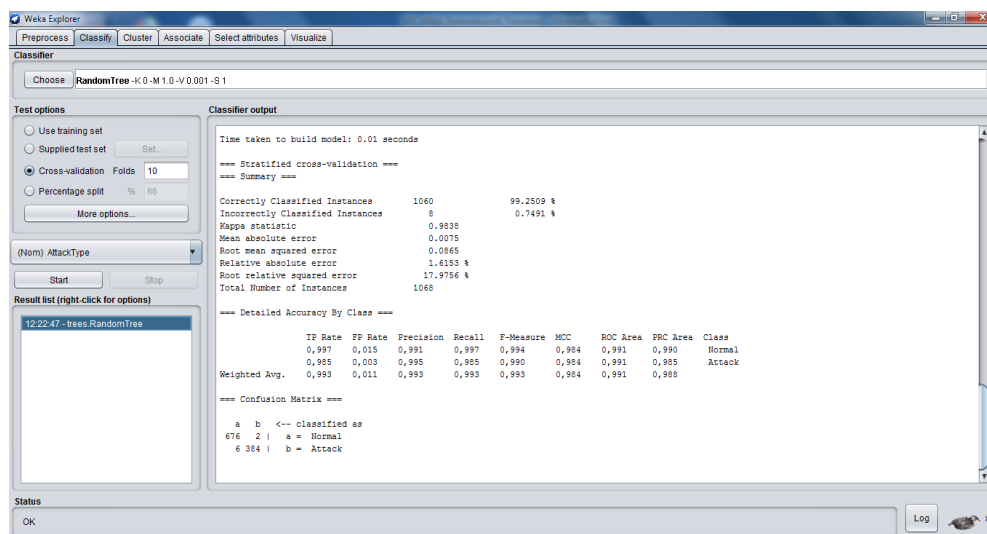
Набор данных, который используется в работе содержат следующие данные: каждое TCP / IP соединение состоит из 41 количественного и качественного признака. Последний 42-ой признак характеризует состояние трафика, которое может быть нормальным или аномальным.

Пример вектора признаков сетевого трафика из файла базы образцов:

91, udp, domain u, SF, 87, 45, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0.00, 0.00, 0.00, 0.00, 1.00, 0.00, 0.00, 0.00, 147, 140, 0.95, 0.01, 0.18, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, normal.

Выделено пять основных классов – один нормальный (normal) и четыре аномальных класса вторжения: DoS, R2L, U2R, Probe, которые сведены в один класс – аномальный.

Ниже на рисунках приводятся результат обнаружения вторжения с помощью метода RandomTree:



Значения статистических характеристик для метода RandomTree:

```
=== Run information ===
```

```
Scheme:      weka.classifiers.trees.RandomTree -K 0 -M 1.0 -V 0.001 -S 1
Relation:    Dataset_Anomaly_II
Instances:   1068
Attributes:  42
```

Test mode: 10-fold cross-validation

```
=== Classifier model (full training set) ===
```

RandomTree
=====

```
dst bytes < 0
```

```

|   error_rate < 0
|   |   service < 0.04
|   |   |   dst_host_same_srv_rate < 0 : Attack (2/0)
|   |   |   |   dst_host_same_srv_rate >= 0
|   |   |   |   |   flag < 0.03
|   |   |   |   |   |   src_bytes < 0
|   |   |   |   |   |   |   dst_host_error_rate < 0 : Normal (3/0)
|   |   |   |   |   |   |   |   dst_host_error_rate >= 0 : Attack (2/0)
|   |   |   |   |   |   |   |   |   src_bytes >= 0 : Normal (14/0)
|   |   |   |   |   |   |   |   |   |   flag >= 0.03 : Attack (1/0)
|   |   |   |   |   |   |   |   |   |   |   service >= 0.04 : Attack (120/0)
|   |   |   |   |   |   |   |   |   |   |   |   error_rate >= 0 : Attack (200/0)
|   |   |   |   |   |   |   |   |   |   |   |   |   dst_bytes >= 0
|   |   |   |   |   |   |   |   |   |   |   |   |   |   num_compromised < 0.05
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   dst_host_srv_count < 0.13
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   hot < 0.05
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   service < 0.04 : Normal (87/0)
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   service >= 0.04 : Attack (3/0)
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   hot >= 0.05 : Attack (5/0)
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   dst_host_srv_count >= 0.13 : Normal (574/0)
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   num_compromised >= 0.05 : Attack (57/0)

```

Size of the tree : 23

Time taken to build model: 0.01 seconds

=== Stratified cross-validation ===
 === Summary ===

Correctly Classified Instances	1060	99.2509 %
Incorrectly Classified Instances	8	0.7491 %
Kappa statistic	0.9838	
Mean absolute error	0.0075	
Root mean squared error	0.0865	
Relative absolute error	1.6153 %	
Root relative squared error	17.9756 %	
Total Number of Instances	1068	

=== Detailed Accuracy By Class ===

Class	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	
mal	0,997	0,015	0,991	0,997	0,994	0,984	0,991	0,990	Normal
attack	0,985	0,003	0,995	0,985	0,990	0,984	0,991	0,985	Attack
Weighted Avg.	0,993	0,011	0,993	0,993	0,993	0,993	0,984	0,991	0,988

=== Confusion Matrix ===

```

a   b   <-- classified as
676   2   |   a = Normal
6 384   |   b = Attack

```

Задание на лабораторную работу.

В качестве исходных данных предлагаются файлы, описывающие нормальное и аномальное состояние сетевого трафика и содержащие значения признаков трафика, которые характерны для компьютерных вторжений и признака принадлежности классу. Признаки описаны в лабораторной работе № 2.

Выбор варианта. N – порядковый номер студента в списке группы (или последние две цифры номера зачетной книжки):

$$N1 = N(\text{MOD } 20) + 1,$$

$$N2 = (N1 + 5)(\text{MOD } 20) + 1,$$

$$N3 = (N2 + 5)(\text{MOD } 20) + 1.$$

Таблица выбора вариантов задания.

№	Файл с данными	Метод 1	Метод 2	Метод 3
1.	Dataset_Anomaly_I.csv	HoeffdingTree	LMT	RandomForest
2.	Dataset_Anomaly_I.csv	J48	DecisionStump	RandomTree
3.	Dataset_Anomaly_I.csv	LMT	J48	HoeffdingTree
4.	Dataset_Anomaly_I.csv	RandomForest	HoeffdingTree	J48
5.	Dataset_Anomaly_I.csv	RandomTree	RandomForest	LMT

6.	Dataset_Anomaly_II.csv	HoeffdingTree	LMT	RandomForest
7.	Dataset_Anomaly_II.csv	J48	DecisionStump	RandomTree
8.	Dataset_Anomaly_II.csv	LMT	J48	HoeffdingTree
9.	Dataset_Anomaly_II.csv	RandomForest	HoeffdingTree	J48
10.	Dataset_Anomaly_II.csv	RandomTree	RandomForest	LMT
11.	Dataset_Anomaly_III.csv	HoeffdingTree	LMT	RandomForest
12.	Dataset_Anomaly_III.csv	J48	DecisionStump	RandomTree
13.	Dataset_Anomaly_III.csv	LMT	J48	HoeffdingTree
14.	Dataset_Anomaly_III.csv	RandomForest	HoeffdingTree	J48
15.	Dataset_Anomaly_III.csv	RandomTree	RandomForest	LMT
16.	Dataset_Anomaly_IV.csv	HoeffdingTree	LMT	RandomForest
17.	Dataset_Anomaly_IV.csv	J48	DecisionStump	RandomTree
18.	Dataset_Anomaly_IV.csv	LMT	J48	HoeffdingTree
19.	Dataset_Anomaly_IV.csv	RandomForest	HoeffdingTree	J48
20.	Dataset_Anomaly_IV.csv	RandomTree	RandomForest	LMT

Содержание отчета

Отчет о проделанной работе должен содержать:

Титульный лист.

Цель работы.

1. Вариант задания.
 2. Формальное описание заданных методов интеллектуальной обработки.
 3. Описание последовательности выполненных режимов в среде Weka.
 4. Скриншоты состояний экранов WEKA.
 5. Протоколы работы классификаторов (Classifier output).
 6. Сравнительный анализ полученных результатов при обработке состояния трафика различными методами.
- Вывод.

Контрольные вопросы

1. Что такое дерево решений?
2. Какие методы классификации на основе деревьев решений Вы знаете?
3. Как работает алгоритм ID3?
4. Нарисуйте идеально сбалансированное двоичное дерево.
5. Перечислите требования к исходным данным.

Лабораторная работа № 3. Исследование методов обнаружения вторжений на основе моделей классификаторов

Цель работы

Выполнить исследование в среде WEKA методов обнаружения вторжений на основе моделей классификаторов.

Краткие теоретические сведения

В данной работе исследуются следующие методы:

DecisionTable – Информация, выраженная в таблицах решений, также может быть представлена в виде деревьев решений или на языке программирования как серия операторов if-then-else и switch-case. Таблицы принятия решений оказались более легкими для понимания и анализа, чем для кода, и были широко использованы и успешно используются для разработки спецификаций для сложных систем (https://en.wikipedia.org/wiki/Decision_table).

JRip – Классы рассматриваются по возрастающему размеру и создается начальный набор правил для класса. Алгоритм является оптимизированной версии IREP (Incremental Reduced Error Pruning <http://weka.sourceforge.net/doc.dev/weka/classifiers/rules/JRip.html>).

OneR – представляет из себя простой и очень эффективный алгоритм сжатия данных, который часто используется в приложениях интеллектуального анализа данных. OneR – это подход к нахождению правила классификации, путем дискретизации числовых атрибутов. В результате генерируется одноуровневое дерево решений, другими словами, использует атрибут минимальной ошибки для прогнозирования (<http://www.soc.napier.ac.uk/~peter/vldb/dm/node8.html>).

OneR – генерирует одно правило для каждого признака данных, затем выбирается правило с наименьшей общей ошибкой как свое «одно правило».

Алгоритм 1R.

Для каждого атрибута A:

Для каждого значения V этого атрибута создать правило:

1. Подсчитать, как часто появляется каждый класс.
2. Найти наиболее частый класс, c.
3. Построить правило «если $A = V$, то $C = c$ ».

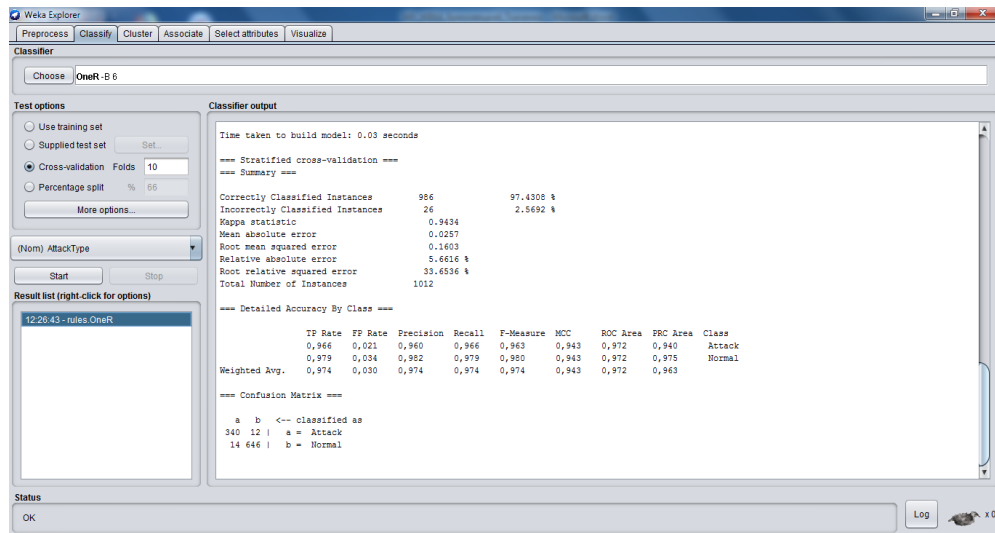
Вычислить частоту ошибок этого правила.

Выберите атрибут, правила которого дают наименьшую частоту ошибок.

PART – Класс для составления списка решений для PART. Использует алгоритм «разделяй-и-властвуй». Создает частичное дерево решений C4.5 на каждой итерации и превращает «лучший» лист в правило ([https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D0%B7%D0%B4%D0%B5%D0%BB%D1%8F%D0%B9_%D0%B8_%D0%B2%D0%BB%D0%B0%D1%81%D1%82%D0%B2%D1%83%D0%B9_\(%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0\)](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D0%B7%D0%B4%D0%B5%D0%BB%D1%8F%D0%B9_%D0%B8_%D0%B2%D0%BB%D0%B0%D1%81%D1%82%D0%B2%D1%83%D0%B9_(%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0))).

Пример выполнения работы

Ниже на рисунках приводятся результат обнаружения вторжения с помощью метода OneR:



Значения статистических характеристик для метода OneR:

=== Run information ===

Scheme: weka.classifiers.rules.OneR -B 6
 Relation: Dataset_Anomaly_III
 Instances: 1012
 Attributes: 42

Test mode: 10-fold cross-validation

=== Classifier model (full training set) ===

```
src_bytes:
  < 6.5E-5      -> Attack
  < 0.00115     -> Normal
  < 0.00133     -> Attack
  < 0.010055    -> Normal
  < 0.010335    -> Attack
  < 0.014530000000000001 -> Normal
  >= 0.014530000000000001 -> Attack
(995/1012 instances correct)
```

Time taken to build model: 0.03 seconds

=== Stratified cross-validation ===
 === Summary ===

Correctly Classified Instances	986	97.4308 %
Incorrectly Classified Instances	26	2.5692 %
Kappa statistic	0.9434	
Mean absolute error	0.0257	
Root mean squared error	0.1603	
Relative absolute error	5.6616 %	
Root relative squared error	33.6536 %	
Total Number of Instances	1012	

=== Detailed Accuracy By Class ===

Class	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area
Attack	0,966	0,021	0,960	0,966	0,963	0,943	0,972	0,940
Normal	0,979	0,034	0,982	0,979	0,980	0,943	0,972	0,975
Weighted Avg.	0,974	0,030	0,974	0,974	0,974	0,943	0,972	0,963

=== Confusion Matrix ===

```
a  b  <-- classified as
340 12 | a = Attack
14 646 | b = Normal
```

Задание на лабораторную работу.

В качестве исходных данных предлагаются файлы, описывающие нормальное и аномальное состояние сетевого трафика и содержащие значения признаков трафика, которые характерны для компьютерных вторжений. Признаки описаны в лабораторной работе № 2.

Выбор варианта. N – порядковый номер студента в списке группы (для ЗФО – последние две цифры номера зачетной книжки):

$$N1=N(\text{MOD } 16)+1,$$

$$N2=(N1+4)(\text{MOD } 16)+1,$$

$$N3=(N2+4)(\text{MOD } 16)+1.$$

Таблица выбора вариантов задания

№	Файл с данными	Метод 1	Метод 2	Метод 3
1.	Dataset_Anomaly_I.csv	DecisionTable	JRip	OneR
2.	Dataset_Anomaly_I.csv	JRip	PART	DecisionTable
3.	Dataset_Anomaly_I.csv	OneR	DecisionTable	PART
4.	Dataset_Anomaly_I.csv	PART	OneR	JRip
5.	Dataset_Anomaly_II.csv	DecisionTable	JRip	OneR
6.	Dataset_Anomaly_II.csv	JRip	PART	DecisionTable
7.	Dataset_Anomaly_II.csv	OneR	DecisionTable	PART
8.	Dataset_Anomaly_II.csv	PART	OneR	JRip
9.	Dataset_Anomaly_III.csv	DecisionTable	JRip	OneR
10.	Dataset_Anomaly_III.csv	JRip	PART	DecisionTable
11.	Dataset_Anomaly_III.csv	OneR	DecisionTable	PART
12.	Dataset_Anomaly_III.csv	PART	OneR	JRip
13.	Dataset_Anomaly_IV.csv	DecisionTable	JRip	OneR
14.	Dataset_Anomaly_IV.csv	JRip	PART	DecisionTable
15.	Dataset_Anomaly_IV.csv	OneR	DecisionTable	PART
16.	Dataset_Anomaly_IV.csv	PART	OneR	JRip

Содержание отчета.

Отчет о проделанной работе должен содержать:

Титульный лист.

Цель работы.

1. Вариант задания.
2. Формальное описание заданных методов интеллектуальной обработки.
3. Описание последовательности выполненных режимов в среде Weka.
4. Скриншоты состояний экранов WEKA.
5. Протоколы работы классификаторов (Classifier output).
6. Сравнительный анализ полученных результатов при обработке состояния трафика.

Вывод.

Контрольные вопросы:

1. Сформулируйте алгоритм работы OneR.
2. Какие частотные методы классификации Вы знаете?

Лабораторная работа №4. Исследование методов обнаружения вторжений на основе байесовской классификации

Цель работы.

Выполнить исследование в WEKA методов обнаружения вторжений на основе моделей байесовской классификации.

Краткие теоретические сведения.

Описание адаптивного алгоритма. Априорная вероятность $P(C_j)$ для каждого класса оценивается путем подсчета, как часто встречается каждый класс в наборе данных T . Аналогично, считается условная вероятность $P(A_{ij} | C_j)$ – для каждого значения атрибута A_{ij} принадлежащего классу C_j в наборе данных T .

Предположим, что пример e_i имеет независимые значения атрибута $\{A_{i1}, A_{i2}, \dots, A_{ip}\}$, зная $P(A_{ij} | C_j)$ для каждого класса C_j , находим апостериорную вероятность как произведение $P(C_j) * P(A_{ij} | C_j)$.

Таким образом, каждый атрибут имеет вычисленные вероятности к каждому из l -классов и соответственно каждый пример e_i будет содержать вычисленные вероятности к каждому из l -классов. После чего обновляем метки класса и производим перерасчет вероятности до тех пор, пока не будут происходить ошибки в классификации.

Алгоритм классифицирует каждый пример по значению максимальной апостериорной вероятности, которая определяет метку класса C_l .

Адаптивный алгоритм на основе байесовского классификатора:

1. На вход поступает обучающая выборка T .
2. Выбрать лучший атрибут по $\text{gain}=\max$. Разделить на подмножества D . Для каждого подмножества выполнить:
3. Рассчитаем априорную вероятность $P(C_j)$ для каждого класса C_j в обучающем множестве T по формуле:

$$P(C_j) = \frac{\sum_{i=1}^n t_{i \rightarrow C_j}}{\sum_{i=1}^n t_i},$$

4. Рассчитаем условную вероятность для значения каждого атрибута принадлежащего каждому классу C_j по формуле

$$P(A_{ij} | C_j) = \frac{\sum_{i=1}^n A_{i \rightarrow C_j}}{\sum_{i=1}^n t_{i \rightarrow C_j}},$$

5. Рассчитаем апостериорную вероятность по формуле для каждого примера (8) и выбираем те примеры, для которых вероятность принимает максимальное значение:

$$P(e_i | C_j) = P(C_j) \prod P(A_{ij} | C_j),$$

6. Обновляем метки принадлежности примеров к классам по максимальным значениям апостериорной вероятности

$$P(e_i | C_j); C_j = C_i \rightarrow P_{\max}(e_i | C_j).$$

7. Повторяем пп. 3-6 для новых меток классов.
8. Если встречаются ошибки классификации, то переход на п.2. Производим разбиение множества. Для полученных подмножеств повторяем пункты.
9. Повторяем пока все примеры неклассифицируются правильно в выборке T .
10. Конец. (Формируем систему правил классификации)

В данной работе исследуются следующие методы:

BayesNet – метод для классификатора Bayes Network. Предоставляет структуры данных (структура сети, условные распределения вероятности и т. Д.) и средства, общие для алгоритмов обучения в сети Bayes, таких как K2 и B. Работает с номинальными переменными и без пропущенных значений

(<http://weka.sourceforge.net/doc.dev/weka/classifiers/bayes/BayesNet.html>).

NaiveBayes – метод для классификатора Naive Bayes с использованием классов оценщиков. Числовые значения точности оценки выбираются на основе анализа данных обучения. По этой причине классификатор не является UpdateableClassifier (который при типичном использовании инициализируется нулевыми экземплярами обучения) - если вам нужна функция UpdateableClassifier, используйте классификатор NaiveBayesUpdateable (<http://weka.sourceforge.net/doc.dev/weka/classifiers/bayes/NaiveBayes.html>).

NaiveBayesMultinomial – метод для построения и использования многокомпонентного классификатора Наивного Байеса

(<http://weka.sourceforge.net/doc.dev/weka/classifiers/bayes/NaiveBayesMultinomial.html>).

NaiveBayesMultinomialText – многокомпонентный классификатор для текстовых данных. Управляет напрямую (и только) атрибутами String. Другие типы входных атрибутов принимаются, но игнорируются во время обучения и классификации (<http://weka.sourceforge.net/doc.dev/weka/classifiers/bayes/NaiveBayesMultinomialText.html>).

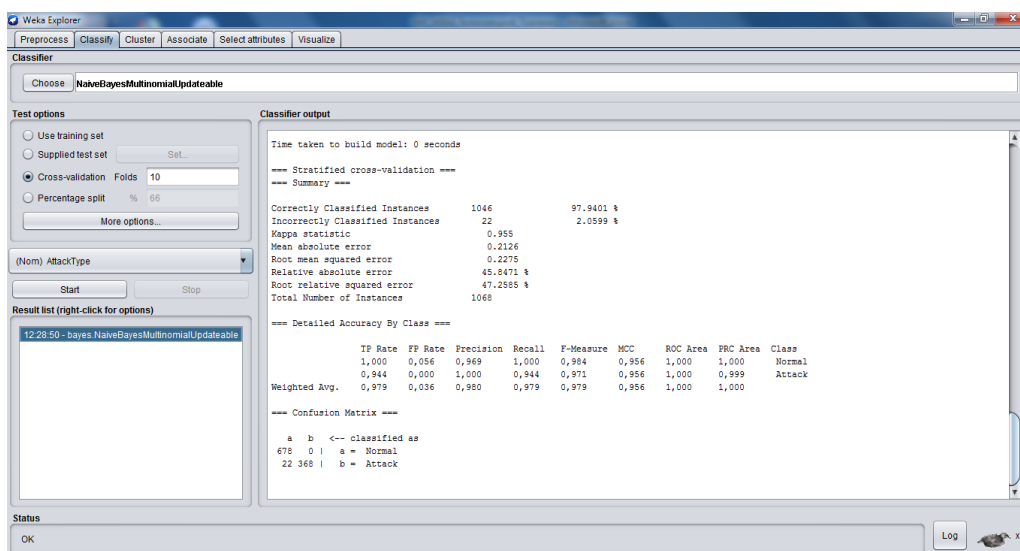
NaiveBayesMultinomialUpdateable – метод для построения и использования обновляемого многокомпонентного классификатора Наивного Байеса (<http://weka.sourceforge.net/doc.dev/weka/classifiers/bayes/NaiveBayesMultinomialUpdateable.html>).

NaiveBayesUpdateable – метод для классификатора Naive Bayes с использованием методов оценщиков. Это обновляемая версия NaiveBayes. Этот классификатор будет использовать стандартную точность 0,1 для числовых атрибутов при вызове buildClassifier с нулевыми экземплярами обучения

(<http://weka.sourceforge.net/doc.dev/weka/classifiers/bayes/NaiveBayesUpdateable.html>).

Пример выполнения работы

Ниже на рисунках приводятся результат обнаружения вторжения с помощью метода NaiveBayesMultinomialUpdateable:



Значения статистических характеристик для метода NaiveBayesMultinomialUpdateable:

=== Run information ===

```

Scheme:      weka.classifiers.bayes.NaiveBayesMultinomialUpdateable
Relation:    Dataset_Anomaly_II
Instances:   1068
Attributes:  42

```

```

Test mode:   10-fold cross-validation

```

=== Classifier model (full training set) ===

The class counts (including Laplace correction)

 Normal 679
 Attack 391

The probability of a word given the class

 Normal Attack

duration	0	0
protocol_type	0	0
service	0	0.08
flag	0	0.04
src_bytes	0.01	0.06
dst_bytes	0.06	0.01
land	0	0
wrong_fragment	0	0
urgent	0	0
hot	0	0.03
num_failed_logins	0	0
logged_in	0.13	0.02
num_compromised	0	0.02
root_shell	0	0
su_attempted	0	0
num_root	0	0
num_file_creations	0	0
num_shells	0	0
num_access_files	0	0
num_outbound_cmds	0	0
is_host_login	0	0
is_guest_login	0	0
count	0.01	0.1
srv_count	0.01	0.07
serror_rate	0	0.03
srv_error_rate	0	0.04
rerror_rate	0	0.02
srv_rerror_rate	0	0.02
same_srv_rate	0.13	0.05
diff_srv_rate	0	0.02
srv_diff_host_rate	0.01	0
dst_host_count	0.14	0.13
dst_host_srv_count	0.29	0.03
dst_host_same_srv_rate	0.13	0.03
dst_host_diff_srv_rate	0	0.02
dst_host_same_src_port_rate	0.01	0.03
dst_host_srv_diff_host_rate	0	0
dst_host_serror_rate	0	0.03
dst_host_srv_serror_rate	0	0.04
dst_host_rerror_rate	0	0.01
dst_host_srv_rerror_rate	0	0.02

Time taken to build model: 0 seconds

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	1046	97.9401 %
Incorrectly Classified Instances	22	2.0599 %
Kappa statistic	0.955	
Mean absolute error	0.2126	
Root mean squared error	0.2275	
Relative absolute error	45.8471 %	
Root relative squared error	47.2585 %	
Total Number of Instances	1068	

=== Detailed Accuracy By Class ===

Class	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area
Normal	1,000	0,056	0,969	1,000	0,984	0,956	1,000	1,000
Attack	0,944	0,000	1,000	0,944	0,971	0,956	1,000	0,999
Weighted Avg.	0,979	0,036	0,980	0,979	0,979	0,956	1,000	1,000

=== Confusion Matrix ===

a	b	<-- classified as
678	0	a = Normal
22	368	b = Attack

Задание на лабораторную работу.

В качестве исходных данных предлагаются файлы, описывающие состояние сетевого трафика двух типов: нормальное и аномальное.

Выбор варианта. N – порядковый номер студента в списке группы (или последние две цифры номера зачетной книжки):

$$N1=N(\text{MOD } 20)+1,$$

$$N2=(N1+5)(\text{MOD } 20)+1,$$

$$N3=(N2+5)(\text{MOD } 20)+1.$$

Таблица выбора вариантов задания

№	Файл с данными	Метод 1	Метод 2	Метод 3
1.	Dataset_Anomaly_I.csv	BayesNet	NaiveBayesMultinomialText	NaiveBayes
2.	Dataset_Anomaly_I.csv	NaiveBayes	BayesNet	NaiveBayesMultinomial
3.	Dataset_Anomaly_I.csv	NaiveBayesMultinomial	NaiveBayes	NaiveBayesMultinomialText
4.	Dataset_Anomaly_I.csv	NaiveBayesMultinomialText	NaiveBayesMultinomial	NaiveBayesMultinomialUpdateable
5.	Dataset_Anomaly_I.csv	NaiveBayesMultinomialUpdateable	NaiveBayesUpdateable	BayesNet
6.	Dataset_Anomaly_II.csv	BayesNet	NaiveBayesMultinomialText	NaiveBayes
7.	Dataset_Anomaly_II.csv	NaiveBayes	BayesNet	NaiveBayesMultinomial
8.	Dataset_Anomaly_II.csv	NaiveBayesMultinomial	NaiveBayes	NaiveBayesMultinomialText
9.	Dataset_Anomaly_II.csv	NaiveBayesMultinomialText	NaiveBayesMultinomial	NaiveBayesMultinomialUpdateable
10.	Dataset_Anomaly_II.csv	NaiveBayesMultinomialUpdateable	NaiveBayesUpdateable	BayesNet
11.	Dataset_Anomaly_III.csv	BayesNet	NaiveBayesMultinomialText	NaiveBayes
12.	Dataset_Anomaly_III.csv	NaiveBayes	BayesNet	NaiveBayesMultinomial
13.	Dataset_Anomaly_III.csv	NaiveBayesMultinomial	NaiveBayes	NaiveBayesMultinomialText
14.	Dataset_Anomaly_III.csv	NaiveBayesMultinomialText	NaiveBayesMultinomial	NaiveBayesMultinomialUpdateable
15.	Dataset_Anomaly_III.csv	NaiveBayesMultinomialUpdateable	NaiveBayesUpdateable	BayesNet
16.	Dataset_Anomaly_IV.csv	BayesNet	NaiveBayesMultinomialText	NaiveBayes
17.	Dataset_Anomaly_IV.csv	NaiveBayes	BayesNet	NaiveBayesMultinomial
18.	Dataset_Anomaly_IV.csv	NaiveBayesMultinomial	NaiveBayes	NaiveBayesMultinomialText
19.	Dataset_Anomaly_IV.csv	NaiveBayesMultinomialText	NaiveBayesMultinomial	NaiveBayesMultinomialUpdateable
20.	Dataset_Anomaly_IV.csv	NaiveBayesMultinomialUpdateable	NaiveBayesUpdateable	BayesNet

Содержание отчета.

Отчет о проделанной работе должен содержать:

Титульный лист.

Цель работы.

1. Вариант задания.
 2. Формальное описание заданных методов интеллектуальной обработки.
 3. Описание последовательности выполненных режимов в среде Weka.
 4. Скриншоты состояний экранов WEKA.
 5. Протоколы работы классификаторов (Classifier output).
 6. Сравнительный анализ полученных результатов при обработке состояния трафика.
- Вывод.

Контрольные вопросы:

1. Какие методы байесовской классификации Вы знаете?
2. Как определяется вероятность встречи класса в наборе данных?
3. Как определяется условная вероятность для значения каждого атрибута?
4. Как определяется апостериорная вероятность?

Литература

1. Барсегян А.А. Анализ данных и процессов: учеб. пособие / А.А. Барсегян, М.С. Куприянов, И.И. Холод, М.Д. Тесс, С.И. Елизаров. – 3-е изд., перераб. и доп. – СПб.: БХВ-Петербург, 2009. – 512 с.: ил. + CD-ROM – (Учебная литература для вузов)
2. Вуколов Э.А. Основы статистического анализа. Практикум по статистическим методам и исследованию операций с использованием пакетов STATISTICA и EXCEL: Учебное пособие. - М.: ФОРУМ: ИНФРА-М, 2004. - 464 с. - ISBN 5-8199-0141 -X, 5-16-002003-9. (см. стр. 8-18)
3. Рутковская Д. Нейронные сети, генетические алгоритмы и нечеткие системы / Д. Рутковская, М Пилиньский, Л. Рутковский - пер. с польск. И. Д. Рудинского. – М.: Горячая линия – Телеком, 2008. – 452 с., ил.
4. Прикладная статистика. Основы эконометрики: учебник для вузов: В 2 т. 2-е изд., испр. - Т. 2: Айвазян С.А. Основы эконометрики. - М.: ЮНИТИ-ДАНА, 2001. – 432 с. (см. стр. 49-53).
5. <https://www.cs.waikato.ac.nz/~ml/weka/>
6. <https://ranalytics.github.io/data-mining/index.html>

**М Е Т О Д Ы
ИНТЕЛЛЕКТУАЛЬНОЙ
ЗАЩИТЫ ИНФОРМАЦИИ**

*Методические указания
к выполнению лабораторных работ*

Составители: Брюховецкий А.А., Ткаченко К.С.

В авторской редакции

Изд. № 23/19. Объем 2,45 уч.-изд. л.
РИИЦМ ФГАОУ ВО «Севастопольский государственный университет»