МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

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

ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Кафедра «Информационные технологии и компьютерные системы»

Р А З Р А Б О Т К А УЧЕБНОЙ ЭКСПЕРТНОЙ СИСТЕМЫ ОБНАРУЖЕНИЯ КОМПЬЮТЕРНЫХ АТАК

Методические указания
к выполнению курсовой работы
по дисциплине «Функциональное и логическое программирование»
для студентов направления подготовки
09.03.01 «Информатика и вычислительная техника»,
по дисциплине «Языки обработки знаний»
для студентов направления подготовки
09.03.04 «Программная инженерия»
на тему: «Разработка учебной экспертной системы
обнаружения компьютерных атак»

Севастополь
СевГУ
2023

УДК 004.056:001.891-057.87(075.8)
ББК 32.973-018.2в6я73
Р177

Р е ц е н з е н т :
С.Г. Лелеков – доцент кафедры «Информационные технологии и компьютерные системы», канд. физ.-мат. наук, доцент

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

Р177 **Разработка учебной экспертной системы обнаружения компьютерных атак** : методические указания к выполнению курсовой работы по дисциплине «Функциональное и логическое программирование» для студентов направления подготовки 09.03.01 «Информатика и вычислительная техника», по дисциплине «Языки обработки знаний» для студентов направления подготовки 09.03.04 «Программная инженерия» / Севастопольский государственный университет, Институт информационных технологий, кафедра «Информационные технологии и компьютерные системы» ; сост.: А. А. Брюховецкий, К. С. Ткаченко. – Севастополь : СевГУ, 2023. – 26 с. – Текст : электронный.

Целью методических указаний является оказание методической помощи бакалаврам при выполнении КР на тему «Разработка учебной экспертной системы обнаружения компьютерных атак» по дисциплинам «Функциональное и логическое программирование» и «Языки обработки знаний» при разработке программ на LISP.

Методические указания содержат:
- краткие теоретические сведения;
- постановку задачи для выполнения работы;
- пример выполнения курсовой работы;
- содержание отчета;
- список литературы;
- индивидуальные задания для выполнения курсовой работы.

УДК 004.056:001.891-057.87(075.8)
ББК 32.973-018.2в6я73

Методические указания рассмотрены и утверждены на заседании кафедры «Информационные технологии и компьютерные системы», протокол № 04 от 28 января 2023 г.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В современном мире рост производительности программиста практически достигается только в тех случаях, когда часть интеллектуальной нагрузки берут на себя компьютеры. Одним из способов достигнуть максимального прогресса в этой области является искусственный интеллект, когда компьютер не только берет на себя однотипные, многократно повторяющиеся операции, но и сам может обучаться. Кроме того, создание полноценного искусственного интеллекта открывает перед человечеством новые горизонты развития.

Курсовая работа рассматривает базовые понятия экспертных систем, методику построения ЭС, а также все этапы построения ЭС: идентификацию, концептуализацию, формализацию, выполнение задачи, тестирование. На практике построение ЭС ведется с использованием средств языка обработки знаний Lisp. Дается разъяснение конкретных путей, которые существуют в настоящее время для повышения интеллектуальности систем проектирования в любой из отраслей производства промышленной продукции и пояснении роли ЭС в современном проектировании.

Имеется, по крайней мере, две точки зрения на то, что следовало бы считать искусственным интеллектом. Первую можно назвать нейробионической. Ее сторонники ставят перед собой цель воспроизвести искусственным образом те процессы, которые протекают в мозгу человека, — это путь изучения естественного мозга, выявление способов его работы, создания технических средств для повторения биологических структур и протекающих в них процессов.

Вторая точка зрения, доминирующая в проблеме искусственного интеллекта, может быть названа информационной. Сторонники информационного исхода считают, что основной целью работ в искусственном интеллекте является не построение технического аналога биологической системы, а создание средств для решения задач, традиционно считающихся интеллектуальными.

Информационная точка зрения в свою очередь неоднородна. В ней можно выделить три направления.

Часть специалистов считает, что можно найти свой способ ее решения на ЭВМ, который даст либо результат, подобный человеческому, либо даже лучший. Специалисты этого направления неоднократно демонстрировали свое искусство по созданию программ такого рода. Достаточно назвать, например, программы для игры в шахматы, которые играют в эту игру лучше подавляющего большинства людей, проводящих время за шахматной доской. Но делают это программы совсем не так, как люди.

Другая часть специалистов считает, что искусственный интеллект должен имитировать не решение отдельных (пусть и весьма творческих) задач. Ибо естественный интеллект человека – это его способность при необходимости обучаться тому или иному виду творческой деятельности, значит, и программы, создаваемые в искусственном интеллекте, должны быть ориентированы не на решение конкретных задач, а на создание для автоматического построения необходимых программ решения конкретных задач, когда в этом возникает необходимость. Именно эта группа исследователей сейчас определяет лицо искус-

ственного интеллекта, составляя основную массу специалистов этого профиля.

Третья часть специалистов – это программисты, чьими руками делают программы для решения задач искусственного интеллекта. Они склонны рассматривать область своей деятельности как новый виток развития программирования. Они считают, что средства, разрабатываемые для написания программ решения интеллектуальных задач, в конце концов, есть средства, позволяющие по описанию задачи на профессиональном естественном языке построить нужную программу на основании тех стандартных программных модулей, которые хранятся в памяти машины. Все метасредства, которые предлагают те, кто рассматривает искусственный интеллект как способ разобраться на информационном уровне, какие функции реализует естественный интеллект, когда он решает задачу, программисты видят сквозь призму своей цели — создания интеллектуального программного обеспечения (по существу, комплекса средств, автоматизирующих деятельность самого программиста).

Система называется интеллектуальной, если в ней реализованы следующие основные функции:

- накапливать знания об окружающем систему мире, классифицировать и оценивать их с точки зрения прагматической полезности и непротиворечивости, инициировать процессы получения новых знаний, осуществлять соотнесение новых знаний с ранее хранимыми;
- пополнять поступившие знания с помощью логического вывода, отражающего закономерности в окружающем систему мире в накопленных ею ранее знаниях, получать обобщенные знания на основе более частных знаний и логически планировать свою деятельность;
- общаться с человеком на языке, максимально приближенном к естественному человеческому языку;
- получать информацию от каналов, аналогичных тем, которые использует человек при восприятии окружающего мира;
- уметь формировать для себя или по просьбе человека (пользователя) объяснение собственной деятельности;
- оказывать пользователю помощь за счет тех знаний, которые хранятся в памяти, и тех логических средств рассуждений, которые присущи системе.

Наиболее значительные успехи в настоящее время достигнуты в таком классе интеллектуальных систем, как экспертные системы (ЭС).

1. Разработка учебной экспертной системы обнаружения компьютерных атак средствами Лисп. Теоретические сведения

Принципы проектирования экспертных систем. Первые ЭС были статического типа. Типичная статическая ЭС должна включать следующие компоненты: базу знаний (БЗ); базу данных (БД, рабочую память); решатель (интерпретатор); систему объяснений; компоненты приобретения знаний; интерфейс с пользователем. БЗ ЭС предназначена для хранения долгосрочных данных, описывающих рассматриваемую область, и правил, описывающих целесообразные преобразования данных этой области. БД ЭС служит для хранения текущих данных решаемой задачи. Решатель формирует последовательность применения правил и осуществляет их обработку, используя данные из рабочей памяти и знания из БЗ. Система объяснений показывает, каким образом система получила решение задачи, и какие знания при этом использовались. Это облегчает тестирование системы и повышает доверие пользователя к полученному результату. Компоненты приобретения знаний необходимы для заполнения ЭС знаниями в диалоге с пользователем-экспертом, а также для добавления и модификации заложенных в систему знаний.

К разработке ЭС привлекаются специалисты из разных предметных областей, а именно: эксперты той проблемной области, к которой относятся задачи, решаемые ЭС; инженеры по знаниям, являющиеся специалистами по разработке интеллектуальных информационных систем (ИИС); программисты, осуществляющие реализацию ЭС. Любая ЭС должна иметь, по крайней мере, два режима работы: режим приобретения знаний; режим консультаций. В режиме приобретения знаний эксперт наполняет ЭС знаниями, которые позволят ЭС в дальнейшем решать конкретные задачи из описанной проблемной области. Эксперт описывает проблемную область в виде совокупности данных об объектах и правил, определяющих взаимные связи между данными, и способы манипулирования данными. В режиме консультаций пользователь ЭС сообщает системе конкретные данные о решаемой задаче и стремится получить с её помощью результат. При этом входные данные о задаче поступают в рабочую память. Решатель на основе данных из БД и правил из БЗ формирует решение.

Динамические ЭС, наряду с компонентами статических ЭС, должны содержать: подсистему моделирования внешнего мира; подсистему связи с внешним окружением. Подсистема моделирования необходима для прогнозирования, анализа и адекватной оценки состояния внешней среды. Изменения окружения решаемой задачи требуют изменения хранимых в ЭС знаний, для того чтобы отразить временную логику происходящих в реальном мире событий.

1.1. Общее представление о деревьях решений.Применение аппарата решающих деревьев к решению задач классификации

В настоящее время деревья решений стали одним из наиболее популярных методов классификации в интеллектуальном анализе данных и бизнес-аналитике. Поэтому они входят в состав практически любого аналитического ПО.

Разработано большое количество различных алгоритмов построения деревьев решений. Наиболее известным является семейство алгоритмов, основанное на критерии прироста информации (information gain) - ID3, C4.5, C5.0, - предложенное Россом Куинленом в начале 1980-х.

Широкая популярность деревьев решений обусловлена следующими их преимуществами:

— правила в них формируются практически на естественном языке, что делает объясняющую способность деревьев решений очень высокой;

— могут работать как с числовыми, так и с категориальными данными;

— требуют относительно небольшой предобработки данных, в частности, не требуют нормализации, создания фиктивных переменных, могут работать с пропусками;

— могут работать с большими объемами данных.

Вместе с тем, деревьям решений присущ ряд ограничений:

— неустойчивость - даже небольшие изменения в данных могут привести к значительным изменениям результатов классификации;

— они не гарантируют построения оптимального дерева;

— склонность к переобучению.

Деревья решений становятся важным инструментом управления бизнес-процессами и поддержки принятия решений.

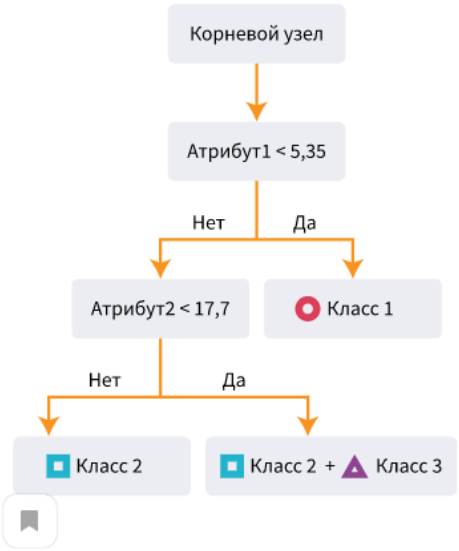
Дерево решений – классификатор, построенный на основе решающих правил вида "если - то", упорядоченных в древовидную иерархическую структуру. В основе работы дерева решений лежит процесс рекурсивного разбиения исходного множества объектов на подмножества, ассоциированные с предварительно заданными классами. Разбиение производится с помощью решающих правил, в которых осуществляется проверка значений атрибутов по заданному условию.

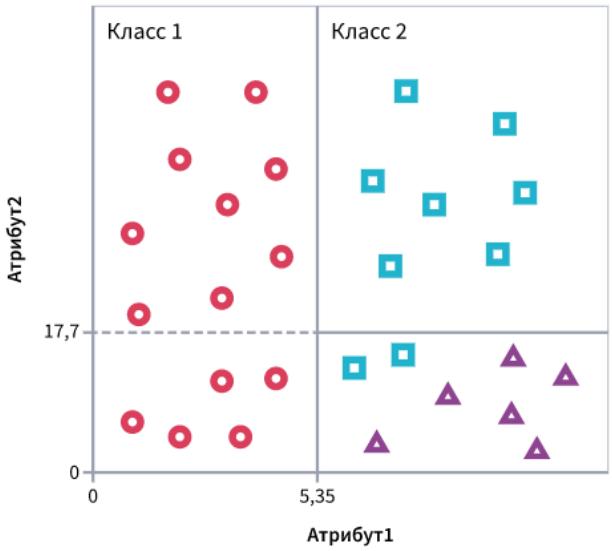
Строятся деревья на основе обучения с учителем. В качестве обучающего набора данных используется множество наблюдений, для которых предварительно задана метка класса.

Структурно дерево решений состоит из объектов двух типов - узлов (node) и листьев (leaf). В узлах расположены решающие правила и подмножества наблюдений, которые им удовлетворяют. В листьях содержатся классифицированные деревом наблюдения: каждый лист ассоциируется с одним из классов, и объекту, который распределяется в лист, присваивается соответствующая метка класса. Ветвление в листьях не производится, и они заканчивают собой ветвь дерева (поэтому их иногда называют терминальными узлами).

Если класс, присвоенный деревом, совпадает с целевым классом, то объект является распознанным, в противном случае - нераспознанным. Самый верхний узел дерева называется корневым (root node). В нем содержится весь обучающий или рабочий набор данных.

Дерево решений является линейным классификатором, т.е. производит разбиение объектов в многомерном пространстве плоскостями (в двумерном случае - линиями).





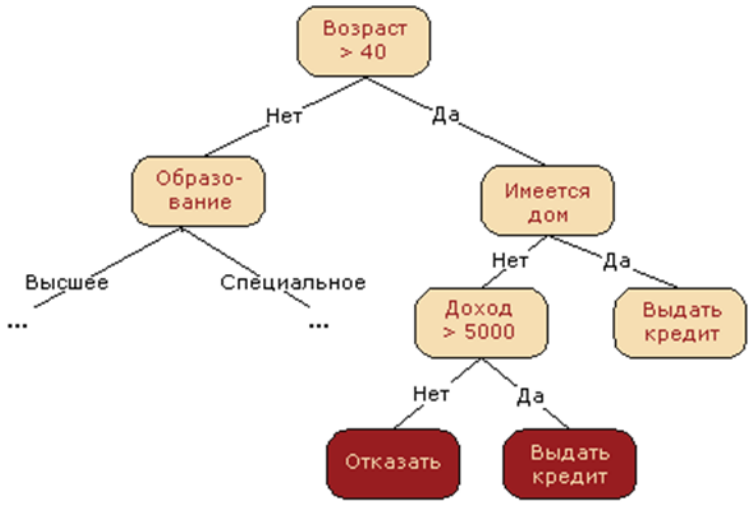
Дерево, представленное на рисунке, решает задачу классификации объектов по двум атрибутам на три класса.

На рисунке кружки представляют объекты класса 1, квадраты — класса 2, а треугольники — класса 3. Пространство признаков разделено линиями на три подмножества, ассоциированных с классами. Эти же подмножества будут соответствовать и трем возможным исходам классификации. В классе «треугольников» имеются нераспознанные примеры («квадраты»), т.е. примеры, попавшие в подмножества, ассоциированные с другим классом.

Теоретически, алгоритм может генерировать новые разбиения до тех пор, пока все примеры не будут распознаны правильно, т.е. пока подмножества, ассоциированные с листьями, не станут однородными по классовому составу. Однако это приводит к усложнению дерева: большое число ветвлений, узлов и листьев усложняет его структуру и ухудшает его интерпретируемость. Поэтому на практике размер дерева ограничивают даже за счет некоторой потери точности. Данный процесс называется упрощением деревьев решений и может быть реализован с помощью методов ранней остановки и отсечения ветвей.

Деревья решений могут быть дихотомичными (бинарными), имеющими только два потомка в узле, и полихотомичными — имеющими более 2-х потомков в узле. Дихотомичные деревья являются более простыми в построении и интерпретации.

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



1.2. 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:

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

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

3. Если множество признаков пустое, то возвратить 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 = vi

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

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.

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

5. Возврат корня.

1.3. Алгоритм C4.5. Базовый алгоритм C4.5 построения деревьев решений, являющийся улучшенной модификацией алгоритма ID3 использует критерий информационной энтропии, или прироста информации (information gain). Обучающий набор данных для алгоритма C4.5 представляет собой множество примеров T=T1,T2,...,Tn, для которых предварительно задана метка класса. Каждый пример представляет собой m-мерный вектор значений атрибутов X={x1,x2,...,xm}.

В каждом узле дерева решений производится выбор атрибута, который позволяет разбить множество попавших в него примеров на подмножества, максимально «чистые» по классовому составу. Чем однороднее полученные подмножества, тем больше их информация и меньше энтропия.

В процессе обучения деревьев решений производится рекурсивное разбиение узлов на узлы-потомки, которые должны быть более однородными по классовому составу попавших в них примеров, чем родительский узел. Следовательно, энтропия дочерних узлов должна быть меньше, чем родительских, а внутренняя информация — больше.

Пусть {C1, C2, ... Ck} – классы (значения меток классов), тогда существуют 3 ситуации:

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

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 x1, x2, ... xn.

T разбивается на подмножества T1, T2, ... Tn, где каждое подмножество Ti содержит все примеры, имеющие значение xi 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.

Разбиение в каждом узле дерева производится по определенному атрибуту из обучающего множества. Поскольку атрибутов несколько, при каждом разбиении приходится решать задачу выбора наилучшего атрибута. Наилучшим атрибутом будет считаться тот, который обеспечит максимально возможное увеличение однородности классов в дочернем узле относительно родительского или, что одно и то же, максимальное снижение энтропии (прирост информации). Следовательно, атрибут, выбираемый для разбиения в узле, должен обеспечивать максимальный прирост информации, или уменьшение энтропии, в результирующих подмножествах. Алгоритм рекурсивно продолжает процедуру разбиения до тех пор, пока в листьях не останутся примеры одного класса. Затем производится упрощение дерева решений путем отсечения ветвей.

Таким образом, критерий прироста информации реализует следующую последовательность действий:

1. Строятся разбиения по всем доступным атрибутам.

2. Выбор атрибута на каждом разбиении производится с помощью критерия прироста информации:

$Gain(X)=Info(T)-InfoX(T)$,

где Info(T) — информация множества до разбиения, InfoX(T) — информация после раз-биения по атрибуту X.

Вычисляется прирост информации (уменьшение энтропии) узла T в результате разбиения.

3. Выбирается атрибут, разбиение по которому обеспечит наибольший прирост информации: $Gain(X) = max$.

В большинстве случаев применение критерия прироста информации для определения значимости атрибутов показывает хорошие результаты. Проблемы возникают, когда атрибут имеет большое разнообразие уникальных значений. В этом случае дерево решений оказывается склонным к переобучению.

Для решения данной проблемы в алгоритмах C4.5 и C5.0 вместо критерия прироста информации используется отношение прироста информации (gain-ratio).

2. Задание на курсовую работу и методика выполнения работы

Курсовая работа выполняется на примере данных, описывающих качественное состояние реального сетевого трафика компьютерной сети. Состояния сетевого трафика описывается набором признаков. Аномальное состояние может быть вызвано внешним воздействием, к которому относятся следующие основные типы атак:

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 курсовой работе рассматриваются только два класса состояний сетевого трафика: нормальное и аномальное.

Цель исследования заключается в поиске оптимального набора параметров, необходимых для эффективного нахождения сетевых атак. Для обучения использовался набор KDDTrain+, для тестирования – KDDTest+. В качестве методов предварительного выбора признаков используются:

- Correlation Attribute Evaluation,
- Information Gain Attribute Evaluator,
- Gain Ratio Attribute Evaluator,
- OneR Attribute Evaluator,
- Symmetrical Uncertainty.

Цель КР – построить решающее дерево классификации векторов, описывающих состояние сетевого трафика компьютерной сети и сравнить полученные результаты с результатами моделирования в среде WEKA.

Постановка задачи. Введем следующие обозначения: Задано состояние сетевого трафика в виде множества векторов Xi={x1, x2, xj, …, xm, Ck} , i = 1, n – количество векторов. Вектор состоит из признаков x_j , j=1,m – количество признаков.

PN_j – нормированное значение признака x_j, 0 <= PN_j <= 1:

$$PN_j = (xj - x_{min})/(x_{max} - x_{min})$$

Gj – граница классификации j-го PN_j признака, 0 <= G_j <= 1;

Ck – признак класса: C_k = 0 – Normal (нормальный), Ck = 1 – Attack (аномальный).

Fj,k – частота появления признака PN_j , соответствующая заданному классу C_k.

Формула для нормализации значений признаков:

2.1. Выбор варианта индивидуального задания

1) **Выбрать файл с множеством векторов и два метода выбора признаков для моделирования в среде WEKA** : N= (I mod 20)+ 1, I – порядковый номер студента в списке группы.

№ N	Файл с данными	Метод 1 выбора признака	Метод 2 выбора признака
1.	Dataset_Anomaly_I.csv	Information Gain Attribute Evaluator	Correlation Attribute Evaluation
2.	Dataset_Anomaly_I.csv	Gain Ratio Attribute Evaluator	Symmetrical Uncertainty Attribute Evaluator
3.	Dataset_Anomaly_I.csv	OneR Attribute Evaluator	Information Gain Attribute Evaluator
4.	Dataset_Anomaly_I.csv	Correlation Attribute Evaluation	Gain Ratio Attribute Evaluator
5.	Dataset_Anomaly_I.csv	Symmetrical Uncertainty Attribute Evaluator	OneR Attribute Evaluator
6.	Dataset_Anomaly_II.csv	Gain Ratio Attribute Evaluator	OneR Attribute Evaluator
7.	Dataset_Anomaly_II.csv	OneR Attribute Evaluator	Correlation Attribute Evaluation
8.	Dataset_Anomaly_II.csv	Correlation Attribute Evaluation	Symmetrical Uncertainty Attribute Evaluator
9.	Dataset_Anomaly_II.csv	Symmetrical Uncertainty Attribute Evaluator	Information Gain Attribute Evaluator
10.	Dataset_Anomaly_II.csv	Information Gain Attribute Evaluator	Gain Ratio Attribute Evaluator
11.	Dataset_Anomaly_III.csv	OneR Attribute Evaluator	Symmetrical Uncertainty Attribute Evaluator
12.	Dataset_Anomaly_III.csv	Correlation Attribute Evaluation	Information Gain Attribute Evaluator
13.	Dataset_Anomaly_III.csv	Symmetrical Uncertainty Attribute Evaluator	Gain Ratio Attribute Evaluator
14.	Dataset_Anomaly_III.csv	Information Gain Attribute Evaluator	OneR Attribute Evaluator
15.	Dataset_Anomaly_III.csv	Gain Ratio Attribute Evaluator	Correlation Attribute Evaluation
16.	Dataset_Anomaly_IV.csv	Correlation Attribute Evaluation	Gain Ratio Attribute Evaluator
17.	Dataset_Anomaly_IV.csv	Symmetrical Uncertainty Attribute Evaluator	OneR Attribute Evaluator
18.	Dataset_Anomaly_IV.csv	Information Gain Attribute Evaluator	Correlation Attribute Evaluation
19.	Dataset_Anomaly_IV.csv	Gain Ratio Attribute Evaluator	Symmetrical Uncertainty Attribute Evaluator
20.	Dataset_Anomaly_IV.csv	OneR Attribute Evaluator	Information Gain Attribute Evaluator

Исследовать в среде WEKA качество классификации векторов сетевого трафика с использованием алгоритма C4.8 с помощью двух методов выбора признаков, заданных в соответствии с вариантом.

2) **В режиме тестирования используется ограниченное число векторов и признаков файла «ТестХ» . Для этого:**

– выбрать входной файл «ТестX», X = (I mod 4)+1;

– из файла «ТестX» выбрать двадцать векторов (половина из которых принадлежит классу атак Ck=Attack, а вторая – нормальных Ck=Normal);

– выбранные вектора с признаком класса сетевого трафика должны содержать следующие признаки Pi (i=1,5):

Первый признак выбирается по правилу P1=(I mod 17)+1,

Второй – P2=((P1+4) mod 17)+1,

Третий – P3=((P2+4) mod 17)+1,

Четвертый – P4=((P3+4) mod 17)+1,

Пятый – P5=((P4+4) mod 17)+1.

Структура исходного файла с признаками представлена ниже. Файл содержит 17 признаков и класс состояния трафика (нормальный - Normal, аномальный - Attack).

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	service	src_bytes	dst_bytes	count	srv_count	serror_rate	srv_error_rate	rerror_rate	same_srv_rate	diff_srv_rate	dst_host_count	dst_host_srv_count	dst_host_same_	dst_host_diff_srv	dst_host_same_sr	dst_host_srv_dif	dst_host_serro	AttackType
2	0,12	0	0	0,002	0,001	0,05	0,1	0	0,05	0,1	0,218	0,001	0	0,095	0,096	0	0,096	Attack
3	0,12	0	0	0,007	0,001	0,071	0,1	0,029	0,014	0,057	0,255	0,001	0	0,002	0,003	0	0,002	Attack
4	0	0,00209	0,15642	0,003	0,003	0	0	0	0,1	0	0,031	0,255	0,1	0	0,003	0,004	0	Normal
5	0	0,00245	0,01651	0,013	0,013	0	0	0	0,1	0	0,235	0,255	0,1	0	0	0,001	0	Normal
6	0	0,00253	0,01906	0,008	0,008	0	0	0	0,1	0	0,021	0,156	0,1	0	0,005	0,004	0	Normal
7	0	0,00242	0,02164	0,022	0,023	0	0	0	0,1	0	0,166	0,255	0,1	0	0,001	0,002	0	Normal
8	0	0,00234	0,13187	0,003	0,003	0	0	0	0,1	0	0,072	0,072	0,1	0	0,001	0	0	Normal
9	0,03	9,00E-05	0,00034	0,001	0,001	0	0	0	0,1	0	0,098	0,018	0,01	0,005	0,001	0,011	0	Normal
10	0	0,00196	0,06107	0,001	0,001	0	0	0	0,1	0	0,039	0,255	0,1	0	0,003	0,004	0	Normal
11	0	0,00322	0,01713	0,013	0,013	0	0	0	0,1	0	0,025	0,255	0,1	0	0,004	0,004	0	Normal
12	0	0,00208	0,01415	0,011	0,011	0	0	0	0,1	0	0,255	0,255	0,1	0	0	0	0	Normal
13	0	0,00307	0,0528	0,008	0,008	0	0	0	0,1	0	0,049	0,255	0,1	0	0,004	0,002	0	Normal
14	0,12	0	0	0,001	0,001	0,1	0,1	0	0,1	0	0,214	0,001	0	0,095	0,096	0	0,096	Attack
15	0,04	8,00E-05	0,00136	0,001	0,001	0	0	0	0,1	0	0,033	0,017	0,009	0,012	0,003	0,012	0	Normal
16	0	0,00197	0,01368	0,005	0,005	0	0	0	0,1	0	0,043	0,255	0,1	0	0,002	0,004	0	Normal
17	0,09	0,01032	0	0,315	0,315	0	0	0	0,1	0	0,147	0,002	0,001	0,002	0,001	0	0	Attack
18	0,12	0	0	0,169	0,001	0,004	0	0,089	0,001	0,099	0,255	0,001	0	0,067	0	0	0,002	Attack
19	0,03	9,00E-05	0,00035	0,003	0,001	0	0	0	0,033	0,067	0,035	0,012	0,011	0,011	0,003	0,017	0	Normal
20	0,01	0,01073	0,00333	0,001	0,001	0	0	0	0,1	0	0,073	0,045	0,062	0,007	0,001	0	0	Normal
21	0	0,00239	0,02195	0,014	0,014	0	0	0	0,1	0	0,049	0,255	0,1	0	0,002	0,004	0	Normal
22	0	0,00228	0,08595	0,014	0,014	0	0	0	0,1	0	0,041	0,132	0,1	0	0,002	0,004	0	Normal
23	0	0,00213	0,30725	0,005	0,005	0	0	0	0,1	0	0,044	0,255	0,1	0	0,002	0,005	0	Normal
24	0	0,003	0,00385	0,017	0,017	0	0	0	0,1	0	0,255	0,255	0,1	0	0	0	0	Normal
25	0,01	0,00869	0,00335	0,001	0,001	0	0	0	0,1	0	0,058	0,11	0,055	0,009	0,002	0,002	0	Normal
26	0	0,00327	0,00729	0,002	0,002	0	0	0	0,1	0	0,016	0,255	0,1	0	0,006	0,001	0	Normal
27	0	0,00255	0,08224	0,001	0,001	0	0	0	0,1	0	0,222	0,255	0,1	0	0	0,001	0	Normal
28	0	0,00141	0,04027	0,001	0,001	0	0	0	0,1	0	0,093	0,255	0,1	0	0,001	0,004	0	Normal
29	0	0,00294	0,016	0,008	0,008	0	0	0	0,1	0	0,109	0,255	0,1	0	0,001	0,004	0	Normal
30	0,12	0	0	0,001	0,001	0,1	0,1	0	0,1	0	0,249	0,001	0	0,096	0,097	0	0,097	Attack
31	0,12	0	0	0,002	0,001	0,1	0,1	0	0,05	0,1	0,027	0,001	0,004	0,07	0,07	0	0,07	Attack
32	0	0,00307	0,00468	0,009	0,014	0	0	0	0,1	0	0,255	0,255	0,1	0	0	0	0	Normal
33	0	0,5454	0,08314	0,006	0,006	0	0	0,017	0,1	0	0,215	0,215	0,1	0	0	0	0	Attack
34	0	0,5454	0,08314	0,011	0,011	0	0	0	0,1	0	0,092	0,092	0,1	0	0,001	0	0	Attack

3) Провести исследование методов построения решающих деревьев и качества классификации векторов сетевого трафика.

3. Разработка учебной экспертной системы обнаружения компьютерных атак. Построение решающего дерева

Режим обучения. Определить (по заданному варианту) последовательность признаков, которые будут использоваться для классификации состояний на каждом уровне. Для каждого признака подсчитывается частота Fj его появления – число ситуаций Pj > Gj, где Pj є Ck, Ck=attack. Таким образом, для каждой вершины будет выбран признак, имеющий Fj = max.

Алгоритм покрытия. Данный метод представляет из себя простой и очень эффективный алгоритм сжатия данных, который часто используется в приложениях интеллектуального анализа данных. Это простой, но точный алгоритм классификации, который генерирует одно правило для каждого признака данных, затем выбирается правило с наибольшой частотой появления признака для заданного множества примеров.

Алгоритм покрытия.

Для каждого значения Pj этого атрибута создать правило Pj >= Gj:

1. Подсчитать частоту Fj признака появления каждого класса Ck, удовлетворяющего условию Pj >= Gj.

2. Найти наиболее частый класс Ck.

3. Построить правило «если Pj >= Gj, то C = Ck».

4. Определить частоту появления этого класса для заданного Pj.

5. Выбрать атрибут, правила которого дают максимальную частоту появления признака для выбранного класса.

4. Режим тестирования и оценка результатов тестирования

Предварительная обработка данных:

Сформировать файл для построения решающего дерева в соответствии с заданием. Структура файла. Для заданного варианта текущее состояние трафика представляется шестимерным вектором. Структура вектора с признаками и соответствующими границами имеет вид:

P1	P2	P3	P4	P5	Ck
G1	G2	G3	G5	G5	Attack/Normal

С целью упрощения алгоритмов обработки данных возможно произвести транспонирование структуры входного файла. Структура полученного файла:

	1	2	3	4	5	6	7	8	9	10	11	12	13
1	count	0,002	0,007	0,003	0,013	…	0,001	0,001	0,005	0,315	0,169	0,003	0,001
2	srv_count	0,001	0,001	0,003	0,013	…	0,001	0,001	0,005	0,315	0,001	0,001	0,001
3	serror_rat	0,05	0,071	0	0	…	0,1	0	0	0	0,004	0	0
4	same_srv_	0,05	0,014	0,1	0,1	…	0,1	0,1	0,1	0,1	0,001	0,033	0,1
5	dst_host_	0,218	0,255	0,031	0,235	…	0,214	0,033	0,043	0,147	0,255	0,035	0,073
6	AttackTyp	Attack	Attack	Normal	Normal	…	Attack	Normal	Normal	Attack	Attack	Normal	Normal

Преобразовать файл.xlsx в формат Common Lisp .LSP. Структура файла .LSP:

```
((count 0.002 0.007 0.003 0.013 … 0.001 0.001 0.005)
(srv_count 0.001 0.001 0.003 0.013 … 0.001 0.001 0.005)
(serror_rate 0.05 0.071 0 0 … 0.1 0 0)
(same_srv_rate 0.05 0.014 0.1 0.1 … 0.1 0.1 0.1)
(dst_host_count 0.21 0.255 0.03 0.235 … 0.214 0.033 0.043)
(AttackType Attack Attack Normal Normal … Attack Normal Normal))
```

Разработать функции на Лиспе для вычисления (или использовать встроенные средства Лиспа):

минимум,

максимум,

нормирования для каждого признака,

среднего значения (граница) для каждого признака,

частоту появления признака для заданного класса,

подсчет числа векторов заданного класса во множестве данных,

выбора вершины с наибольшей частотой появления признака для выбранного класса,

исключения выбранного признака с наибольшей частотой из множества для следующей итерации,

формирование списков типов атак, полученных в результате классификации,

сравнение списков типов атак с эталонным (входным) списком,

формирования матрицы ошибок,

вычисления значений метрик по матрице ошибок.

Пример выполнения функций программы с заданными списками (dst_host_count и same_srv_rate):

```
(setf amin (min 0.218 0.255 0.031 0.235 0.214 0.033)) ; минимальное значение dst_host_count
(setf amax (max 0.218 0.255 0.031 0.235 0.214 0.033)) ; максимальное значение dst_host_count
(setf bmin (min 0.05 0.014 0.1 0.1 0.1 0.1)) ; минимальное значение same_srv_rate
(setf bmax (max 0.05 0.014 0.1 0.1 0.1 0.1)) ; максимальное значение same_srv_rate
(print amin)
(print amax)
(print bmin)
(print bmax)
(setf amaxmin (- amax amin)) ; разброс для dst_host_count
(setf bmaxmin (- bmax bmin)) ; разброс для same_srv_rate
(print amaxmin)
(print bmaxmin)
```

Выполнение программы:

0.031
0.255
0.014
0.1
0.224
0.086

Иллюстрация разработанного студентом алгоритма построения решающего дерева

1. Входное множество нормальных и аномальных векторов:

P1=dst_host_count	P2=same_srv_rate	Класс
0,218	0,05	Attack
0,255	0,014	Attack
0,031	0,1	Normal
0,235	0,1	Normal
0,214	0,1	Attack
0,033	0,1	Normal

2. Входное множество после операции нормирования:

P1=dst_host_count	P2=same_srv_rate	Класс
0,835	0,419	Attack
1,000	0,000	Attack
0,000	1,000	Normal
0,911	1,000	Normal
0,817	1,000	Attack
0,009	1,000	Normal

3. Разбиение множества на два подмножества и выбор признака с наибольшей частотой появления для каждого подмножества. Для признака P1 имеют место условия:

dst_host_count <= 0,5
dst_host_count > 0,5

Правило 1:

Исследуется структура правила на первом (корневом) уровне дерева вида:

Если (?) то Класс = Attack

Вместо вопросительного знака должно быть записано условие одно из ниже проверяемых условий.

Применяются условия ко множеству векторов с целью выбора наиболее часто встречающегося признака:
P1=dst_host_count <= 0,5 – 0/6
P1=dst_host_count > 0,5 – 3/6
P2=same_srv_rate <= 0,5 – 2/6
P2=same_srv_rate > 0,5 – 1/6
В результате формируется множество векторов. Красным цветом помечены выбранные признаки векторов и их классы, принадлежащие множеству, для которого выполняется условие dst_host_count > 0,5 и Ck=Attack (3/6).

P1=dst_host_count	P2=same_srv_rate	Класс
0,835	0,419	Attack
1,000	0,000	Attack
0,000	1,000	Normal
0,911	1,000	Normal
0,817	1,000	Attack
0,009	1,000	Normal

Таким образом, в левое подмножество на втором уровне (см. рисунок ниже) попадут четыре вектора и будет сформировано первое правило:

Если (P1=dst_host_count > 0,5) то Класс = Attack

Правило 2:
Исследуется структура правила на втором уровне дерева вида:

Если (P1=dst_host_count > 0,5 и ?) то Класс = Attack

Применяются условия ко множеству векторов с целью выбора наиболее часто встречающегося признака:
P2=same_srv_rate <= 0,5 – 2/4
P2=same_srv_rate > 0,5 – 1/4

P1=dst_host_count	P2=same_srv_rate	Класс
0,835	0,419	Attack
1,000	0,000	Attack
0,911	1,000	Normal
0,817	1,000	Attack

В результате будет сформировано второе правило:

Если (P1=dst_host_count > 0,5 и P2=same_srv_rate <= 0,5) то Класс = Attack

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

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.

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

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

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

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

$$True\ Positive = \frac{TP}{Total\_class\_samples}$$

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

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

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

Значение **F-Measure** вычисляется по формуле

F-Measure = 2*Precision*Recall/(Precision+Recall)

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

В результате применения метода выводится матрица ошибок размера l x 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.
```

Важно знать метрики, используемые для оценки качества деревьев решений. Основными используемыми метриками являются точность, чувствительность, специфичность, точность, частота пропусков, частота ложных обнаружений и частота ложных пропусков. Все эти метрики основаны на количестве истинных положительных, ложных положительных, истинных отрицатель-

ных и ложных отрицательных результатов, полученных при выполнении обработки с помощью модели классификации дерева решений. Для отображения этих результатов формируется матрица ошибок. Все эти основные показатели говорят о сильных и слабых сторонах модели классификации, построенной на основе вашего дерева решений.

Для примера рассмотрим приведенную ниже матрицу ошибок. Матрица ошибок показывает нам, что построенный классификатор модели дерева решений дал 11 истинных положительных результатов, 1 ложноположительный результат, 45 ложных отрицательных результатов и 105 истинных отрицательных результатов.

	Прогноз: Attack	Прогноз: Normal
Факт: Attack	TP =11	FN =45
Факт: Normal	FP =1	TN =105

Теперь мы рассчитаем оценки значений: точность, чувствительность, специфичность, точность, частоту пропусков, частоту ложных обнаружений и частоту ложных пропусков (accuracy, sensitivity, specificity, precision, miss rate, false discovery rate, and false omission rate.)

Точность:

Accuracy=(TP+TN)/(TP+TN+FP+FN)

(11+105) /162=71,60%

Чувствительность (TPR - Истинно положительная частота)

TPR=TP/(TP+FN)

11 / (11+45)=19.64%

Специфичность (TNR - Истинно Отрицательный показатель):

TNR=TN/(TN+FP)

105 / (105+1)=99.06%

Частота ложных обнаружений(FDR):

FDR=FP/(FP+TP)

1/ (1+11)=8.30%

Частота ложных пропусков (FOR):

FOR=FN/(FN+TN)

45 / (45+105)=30.00%

5. Содержание отчета о выполнении курсовой работы

Оформить отчет, содержащий:
Титульный лист.
Цель работы.
1. Постановка задачи (по варианту).
2. Описание структур данных.
3. Результаты построения решающего дерева в среде WEKA:
- иллюстрация режима выбора признаков (описание режима и параметров выбора, сканы экрана),
- оценки информативности выбранных признаков (скан экрана),
- иллюстрация режима работы классификатора (описание и сканы экрана),
- структура матрицы ошибок,
- описание метрик классификации, основные соотношения, полученные значения,
- сформированное решающее дерево (структурная схема и система правил).

выполнения программ в виде сканов экрана.
4. Разработка программы построения решающего дерева:
- схема алгоритма построения решающего дерева,
- описание функций алгоритма (входные/выходные данные,
- структура программной системы.

5. Тестирование разработанной программы.

Предоставить результаты проверки построенной системы правил классификации:
- таблица нормированных значений,
- подсчетом числа векторов заданного класса, соответствующих указанному условию, для заданной вершины,
- частоту появления признака для заданной вершины дерева,
- формирование матрицы ошибок,
- расчет метрик классификации для листовых вершин,

6. Сравнительный анализ результатов качества классификации векторов.
Выводы.

Контрольные вопросы

1. Что понимают под экспертной системой?
2. Назовите особенности экспертных систем.
3. Приведите структуру экспертной системы.
4. Какими качествами должна обладать ЭС?
5. Назовите трудности, возникающие при разработке ЭС.
6. Что называют метазнаниями?
7. Что дают пользователю экспертные системы?
8. Назовите типы задач, решаемых ЭС.
9. Какие действия выполняет диалоговый компонент ЭС?
10. Какие функции выполняет решатель?
11. Какие действия выполняет объяснительный компонент?
12. Опишите режимы работы ЭС.
13. Что означает термин «знание» в искусственном интеллекте?
14. Покажите различие между алгоритмическим и эвристическим методами.
15. Опишите режим консультации ЭС.
16. Что входит в компонент приобретения знаний?
17. Что такое система, основанная на знаниях?

Библиографический список

1. Джарратано, Джозеф, Райли, Гари. Экспертные системы: принципы разработки и программирования, 4-е издание. – М.: Вильямс, 2007. – 1152 с.

2. Хендерсон П. Функциональное программирование. – М.: Мир,1983. – 347 с.

3. Хювенен Э., Сеппянен Й. Мир Лиспа : пер. с фин. – В 2-х т. : Т.1, 2. – М.: Мир, 1990.

4. Городняя Л .В. Основы функционального программирования. – Новосибирск, 2004. – 126 с.

ПРИЛОЖЕНИЕ А
Пример оформления титульного листа

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СЕВАСТОПОЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт Информационных технологий
Кафедра «Информационные технологии и компьютерные системы»

Пояснительная записка
к курсовой работе по дисциплине
«Функциональное и логическое программирование»

на тему ***Разработка учебной экспертной системы обнаружения компьютерных атак***

Выполнил: студент ***2*** курса, группы ***ИВТПИН/б-21-1-о***

направления подготовки 09.03.01 «Информатика и вычислительная техника»/09.03.04 «Программная инженерия»

Фамилия, имя, отчество студента

Руководитель ***Брюховецкий Алексей Алексеевич, к.т.н, доцент,***
(фамилия, инициалы, степень, звание, должность)

заведующей кафедрой ИТКС

Дата допуска к защите «____»_____ 2023_ г.

Зав. кафедрой _____ ***А.А. Брюховецкий***
(подпись) (инициалы, фамилия)

2023 г.

ПРИЛОЖЕНИЕ Б
Пример оформления листа задания

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СЕВАСТОПОЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт Информационных технологий
Кафедра «Информационные технологии и компьютерные системы»
Направление подготовки/специальность 09.03.01 «Информатика и вычислительная техника»/09.03.04 «Программная инженерия»
(код и название)
Профиль/специализация Электронно-вычислительные машины, системы и сети/Системное и прикладное программное обеспечение
Курс 2 Группа ИВТПИН/б-21-1-о Семестр 3

З А Д А Н И Е
НА КУРСОВОЙ ПРОЕКТ (РАБОТУ) СТУДЕНТА

Иванов Иван Иванович
(фамилия, имя, отчество)

1. Тема проекта (работы) Разработка учебной экспертной системы обнаружения компьютерных атак

2. Сроки сдачи студентом законченного проекта (работы) «31» мая 2023 г.

3. Исходные данные к проекту (работе) Вариант 1.

4. Содержание пояснительной записки (перечень вопросов, подлежащих разработке)
1. Постановка задачи (по варианту).
2. Описание структур данных.
3. Результаты построения решающего дерева в среде WEKA:
• иллюстрация режима выбора признаков (описание режима и параметров выбора, сканы экрана),
• оценки информативности выбранных признаков (скан экрана),
• иллюстрация режима работы классификатора (описание и сканы экрана),
• структура матрицы ошибок,
• описание метрик классификации, основные соотношения, полученные значения
• сформированное решающее дерево (структурная схема и система правил)
выполнения программ в виде сканов экрана.
4. Разработка программы построения решающего дерева:
• схема алгоритма построения решающего дерева,
• описание функций алгоритма (входные/выходные данные,
• структура программной системы.
5. Тестирование разработанной программы.
Предоставить результаты проверки построенной системы правил классификации:
• таблица нормированных значений,
• подсчетом числа векторов заданного класса, соответствующих указанному условию, для заданной вершины,

• частоту появления признака для заданной вершины дерева,
• формирование матрицы ошибок,
• расчет метрик классификации для листовых вершин,
6. Сравнительный анализ результатов качества классификации векторов.

5. Перечень графического материала (презентация)

1. СХЕМА ПРОГРАММЫ 2. СХЕМА АЛГОРИТМА 3. ДЕРЕВО ПРИНЯТИЯ РЕШЕНИЙ.

6. Содержание презентации: 1. ТИТУЛЬНЫЙ СЛАЙД. 2. ПОСТАНОВКА ЗАДАЧИ. 3. ОПИСАНИЕ СТРУКТУР ДАННЫХ. 4. РЕЗУЛЬТАТЫ ПОСТРОЕНИЯ РЕШАЮЩЕГО ДЕРЕВА В СРЕДЕ WEKA. 5. РАЗРАБОТКА ПРОГРАММЫ ПОСТРОЕНИЯ РЕШАЮЩЕГО ДЕРЕВА. 6. ТЕСТИРОВАНИЕ РАЗРАБОТАННОЙ ПРОГРАММЫ. 7. СРАВНИТЕЛЬНЫЙ АНАЛИЗ РЕЗУЛЬТАТОВ КАЧЕСТВА КЛАССИФИКАЦИИ ВЕКТОРОВ. ЗАКЛЮЧЕНИЕ.

7. Дата выдачи задания 06.02.2023 г.

КАЛЕНДАРНЫЙ ПЛАН

№ п/п	Наименование этапов выполнения курсовой работы	Срок выполнения этапов проекта (работы) – номер учебной недели	Примечания
1.	Получение задания и уточнение постановки задачи	1–2	
2.	Анализ структурной схемы дерева вывода	3–4	
3.	Разработка дерева принятия решений	5	
4.	Разработка последовательности запуска и подготовки файлов	6–7	
5.	Подготовка текста программы, входных и выходных файлов	8–10	
6.	Тестирование и отладка программы	11–14	
7.	Оформление пояснительной записки и презентации	15–16	
8.	Защита курсовой работы (в соответствии с графиком защиты)	17–18	

Студент Иванов И.И.
(подпись)

Руководитель проекта (работы) Брюховецкий А.А., зав. кафедрой
(подпись)

«_06_» 02 2023 г.

Р А З Р А Б О Т К А
УЧЕБНОЙ ЭКСПЕРТНОЙ
СИСТЕМЫ ОБНАРУЖЕНИЯ
КОМПЬЮТЕРНЫХ АТАК

Методические указания
к выполнению курсовой работы

Составители: **Брюховецкий** Алексей Алексеевич,
Ткаченко Кирилл Станиславович

В авторской редакции

Изд. № 16/2023. Объем 1,75 п.л. Усл. печ. л. 1,63. Уч.-изд. л. 1,715.
РИИЦМ ФГАОУ ВО «Севастопольский государственный университет»