МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«Тольяттинский государственный университет»

Институт **математики, физики и информационных технологий**

Кафедра **«Прикладная математика и информатика»**

02.03.03 МАТЕМАТИЧЕСКОЕ ОБЕСПЕЧЕНИЕ И АДМИНИСТРИРОВАНИЕ ИНФОРМАЦИОННЫХ СИСТЕМ

ТЕХНОЛОГИЯ ПРОГРАММИРОВАНИЯ

БАКАЛАВРСКАЯ РАБОТА

на тему «Разработка реализации алгоритма построения деревьев принятия решений с использованием алгоритма C4.5 в среде Matlab»

Студент Д.А. Рожков _____

Руководитель В.С. Климов _____

Консультант по аннотации Н.В. Ященко _____

Допустить к защите
Заведующий кафедрой к.тех.н., доцент, А.В. Очеповский _____

«_____» _____ 2017 г.

Тольятти 2017

АННОТАЦИЯ

Тема: «Разработка реализации алгоритма построения деревьев принятия решений с использованием алгоритма C4.5 в среде Matlab».

В данной выпускной квалификационной работе исследуются реализации алгоритмов искусственного интеллекта в Matlab. В исследовании рассматриваются задачи изучения математического аппарата алгоритма C4.5 и разработка его программной реализации в среде Matlab.

Структура ВКР представлена введением, тремя главами, заключением, списком литературы.

Во введении описывается актуальность проводимого исследования, заключающаяся в отсутствии реализаций алгоритма C4.5 в среде Matlab. В первой главе проводится обзор методов математического моделирования и анализ реализаций алгоритмов искусственного интеллекта в математическом пакете Matlab.

Во второй главе описывается задача классификации, ее решение с помощью дерева принятия решений и математический аппарат построения дерева по алгоритму C4.5.

В третьей главе проведена реализация алгоритма C4.5 в среде Matlab, а также представлен подробный пример его работы с использованием обучающей выборки.

Был сделан вывод, что алгоритм при разбиении исходных множеств учитывает не только необходимость максимального снижении информационной энтропии, но и сбалансированность получаемых в результате разбиения подмножеств. Эта информация легла в основу программы построения дерева принятия решения в Matlab.

Данная бакалаврская работа состоит из пояснительной записки на 41 стр., включая 24 рисунка, 2 таблицы, списка из 26 источников на иностранном языке и 1 приложения.

ABSTRACT

The title of the graduation work is “Development of Implementation Algorithm of Decision Trees with Use of a Method C4.5 in the Environment of Matlab”.

In this graduation work implementations of algorithms of an artificial intelligence in Matlab are researched. In a research tasks of a study of a mathematical apparatus of an algorithm C4.5 and development of its program implementation in the environment of Matlab are considered.

The structure of the graduation work is represented by an introduction, three parts, a conclusion and a list of references.

The introduction describes the relevance of the conducted research, which consists in the absence of realizations of the C4.5 algorithm in the Matlab environment.

In the first part the review of methods of mathematical simulation in the environment of Matlab is carried out.

The second part describes the problem of classification, its solution with the decision tree and the mathematical apparatus for constructing the tree using the C4.5 algorithm.

In the third part the implementation of the C4.5 algorithm in the Matlab environment is carried out, and a detailed example of its work using the training sample is presented.

It was concluded, that the algorithm with the sharing of initial sets takes into account not only the need to minimize the information entropy, but also the balance of the resulting partition of the subsets. This information formed the basis of the program for constructing the decision tree in Matlab.

The graduation work consists of 41 pages, including 24 figures, 2 tables, the list of 26 foreign references and 1 appendix.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

MATLAB — это высокоуровневый язык и интерактивная среда для программирования численных расчетов и визуализации результатов.

MATLAB широко используется в таких областях, как:

- обработка сигналов и связь,
- обработка изображений и видео,
- системы управления,
- автоматизация тестирования и измерений,
- финансовый инжиниринг и др.

Алгоритмы искусственного интеллекта реализованы в таких модулях Matlab, как Neural Network Toolbox, Optimization toolbox, Statistic and machine learning toolbox, Fuzzy Logic Toolbox. Однако представленные в них реализации имеют не полную поддержку алгоритмов искусственного интеллекта.

Так как в настоящее время не существует официальной поддержки алгоритма C4.5 математическим пакетом Matlab, в рамках данного исследования реализуется данный алгоритм в среде Matlab.

Таким образом актуальной можно признать цель исследования – разработка реализации алгоритма машинного обучения C4.5 в математическом пакете Matlab.

Поставленная цель достигается путем последовательного решения следующих задач:

1. Провести анализ реализаций алгоритмов искусственного интеллекта в математическом пакете Matlab.

2. Изучить математический аппарат алгоритма C4.5.

3. Спроектировать и разработать реализацию алгоритма C4.5 в математическом пакете Matlab.

4. Протестировать реализацию алгоритма C4.5.

ГЛАВА 1 АНАЛИЗ ИСПОЛЬЗОВАНИЯ MATLAB ПРИ МАТЕМАТИЧЕСКОМ МОДЕЛИРОВАНИИ

1.1 Обзор предметной области

Математический пакет Matlab состоит из высокоуровневого языка программирования и интерактивной среды разработки приложений. Matlab предназначен для выполнения численных расчетов и наглядной визуализации полученных результатов (рисунок 1.1). Возможности Matlab позволяют выполнять анализ исходных данных, разрабатывать алгоритмы, составлять математические модели [1-3].

Описывая преимущества Matlab, разработчики сравнивают данный пакет с Microsoft Excel, языками программирования C++, C#, Java.

Математический пакет Matlab используется в таких областях деятельности, как обработка сигналов, анализ изображений, автоматизация вычислительных экспериментов, вычислительная биология, финансовый инжиниринг и другие [4].

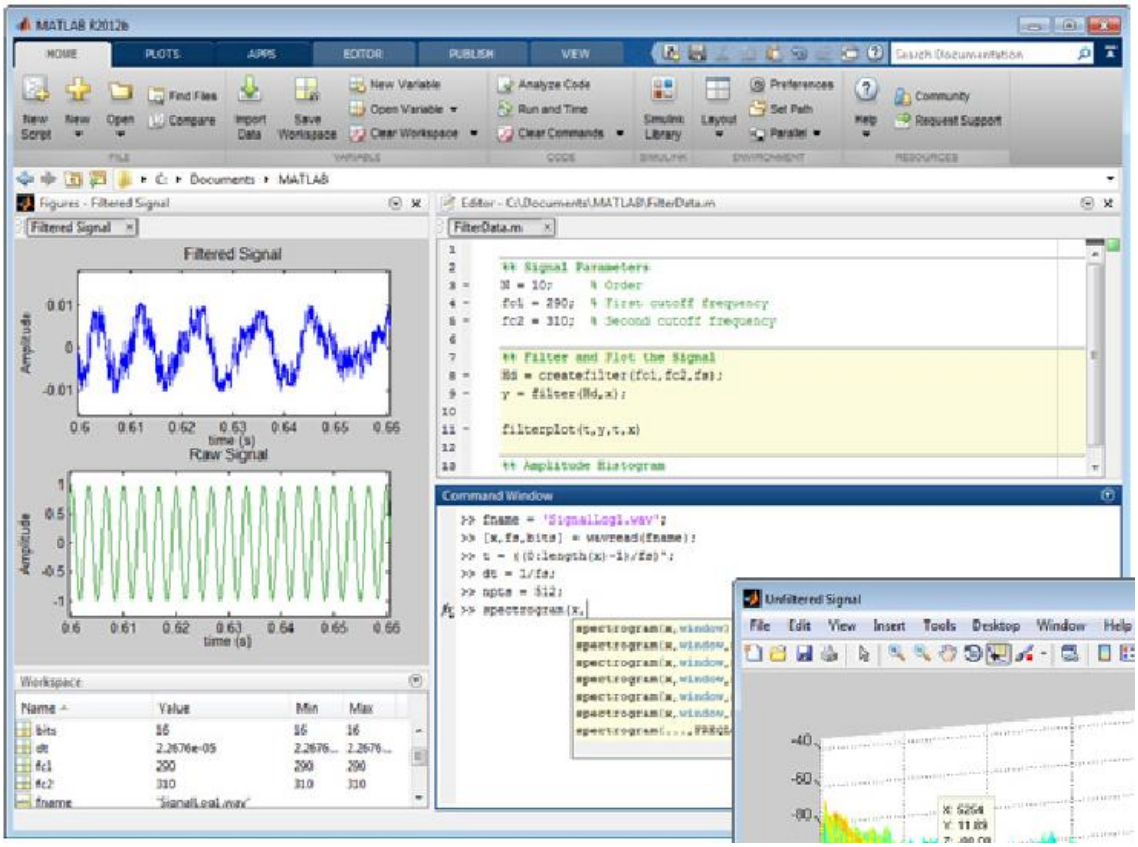


Рисунок 1.1 – Визуализация расчетов в среде Matlab 2017b

Пакет Matlab включает в себя математические функции не только для инженерных расчетов, в него включены также функции для научных исследований. Библиотеки, которые использует Matlab для вычислений, оптимизированы для работы с векторными и матричными представлениями данных [5].

Рассмотрим функционал Matlab с точки зрения выполнения численных вычислений:

- регрессия и интерполяция (рисунок 1.2);
- системы линейных уравнений;
- собственные значения и сингулярные числа матриц;
- разряженные матрицы;
- дифференцирование и интегрирование;
- анализ Фурье;
- обыкновенные дифференциальные уравнения.

Дополнительный функционал реализуется с помощью дополнительных расширений (в оригинале – toolbox’ов).

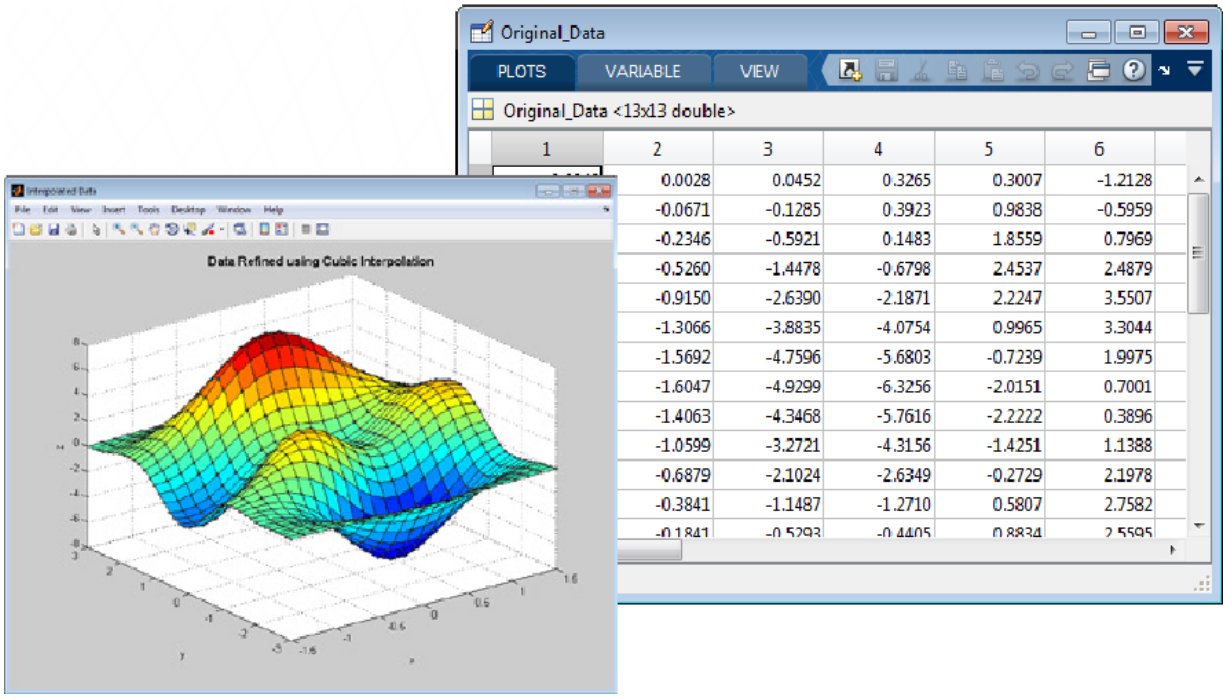


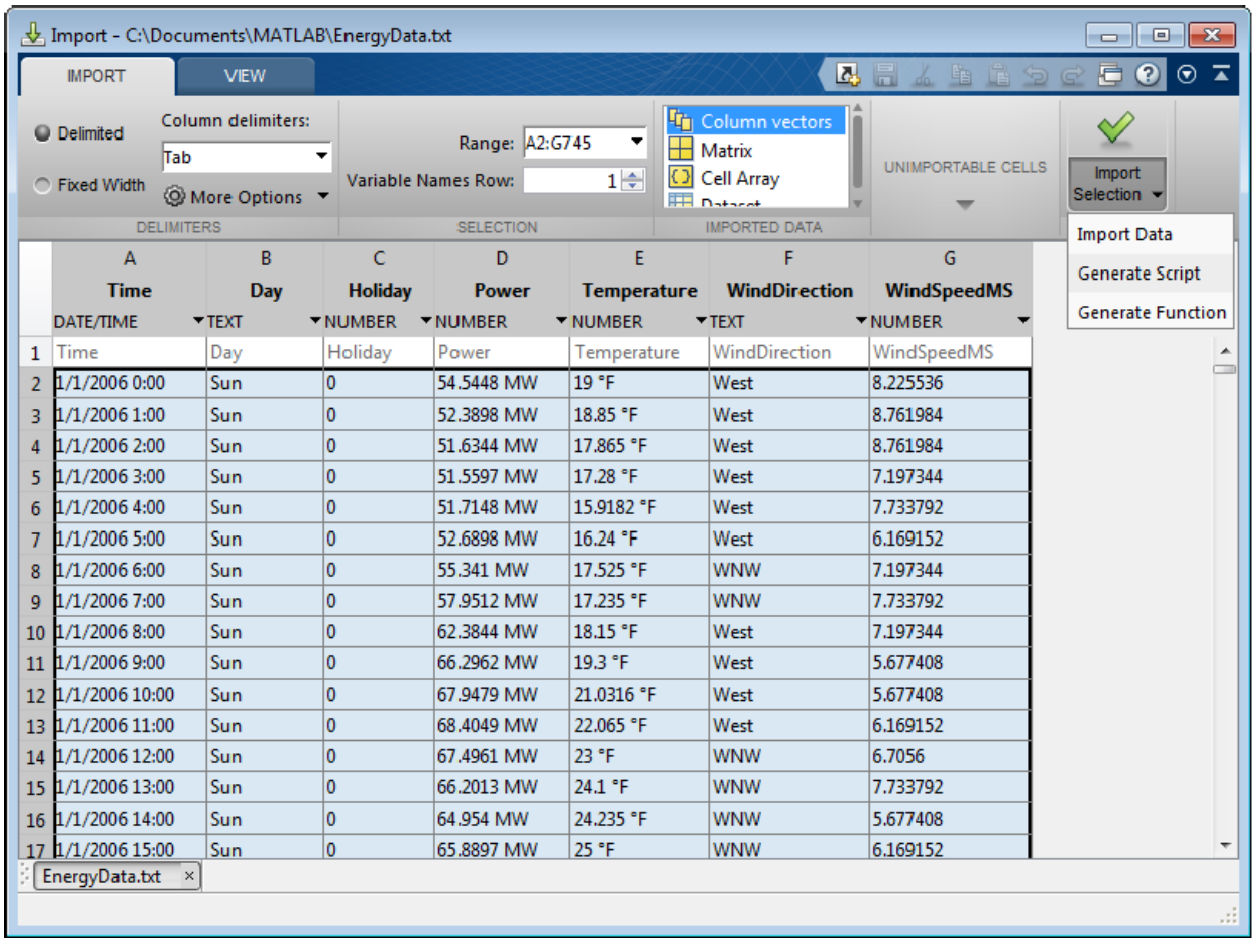
Рисунок 1.2 – Кубическая интерполяция значений функции в среде Matlab 2017b

Реализованный функционал Matlab обеспечивает возможность импорта данных из файлов сторонних приложений. Также существует возможность считывания информации из баз данных, а также внешних устройств [6-8].

Реализовано считывание информации из таких форматов, как:

- xls-файлы (excel);
- txt-файлы (текстовые файлы) (рисунок 1.3);
- двоичные файлы;
- файлы изображений;
- другие форматы файлов.

Помимо загрузки данных из файлов Matlab поддерживает функцию считывания потоковых данных с научных приборов – осциллографов, генераторов, измерительных устройств (в реальном времени).



	A	B	C	D	E	F	G
	Time	Day	Holiday	Power	Temperature	WindDirection	WindSpeedMS
	DATE/TIME	TEXT	NUMBER	NUMBER	NUMBER	TEXT	NUMBER
1	Time	Day	Holiday	Power	Temperature	WindDirection	WindSpeedMS
2	1/1/2006 0:00	Sun	0	54.5448 MW	19 °F	West	8.225536
3	1/1/2006 1:00	Sun	0	52.3898 MW	18.85 °F	West	8.761984
4	1/1/2006 2:00	Sun	0	51.6344 MW	17.865 °F	West	8.761984
5	1/1/2006 3:00	Sun	0	51.5597 MW	17.28 °F	West	7.197344
6	1/1/2006 4:00	Sun	0	51.7148 MW	15.9182 °F	West	7.733792
7	1/1/2006 5:00	Sun	0	52.6898 MW	16.24 °F	West	6.169152
8	1/1/2006 6:00	Sun	0	55.341 MW	17.525 °F	WNW	7.197344
9	1/1/2006 7:00	Sun	0	57.9512 MW	17.235 °F	WNW	7.733792
10	1/1/2006 8:00	Sun	0	62.3844 MW	18.15 °F	West	7.197344
11	1/1/2006 9:00	Sun	0	66.2962 MW	19.3 °F	West	5.677408
12	1/1/2006 10:00	Sun	0	67.9479 MW	21.0316 °F	West	5.677408
13	1/1/2006 11:00	Sun	0	68.4049 MW	22.065 °F	West	6.169152
14	1/1/2006 12:00	Sun	0	67.4961 MW	23 °F	WNW	6.7056
15	1/1/2006 13:00	Sun	0	66.2013 MW	24.1 °F	WNW	7.733792
16	1/1/2006 14:00	Sun	0	64.954 MW	24.235 °F	WNW	5.677408
17	1/1/2006 15:00	Sun	0	65.8897 MW	25 °F	WNW	6.169152

Рисунок 1.3 – Импорт данных из txt файла с помощью Import Tool

С помощью Matlab можно осуществлять анализ данных с целью поиска трендов, проверки гипотез, построения математических моделей. Для этого в среде Matlab реализован следующий функционал – использование функций сглаживания и фильтрации, функции свертки, функция преобразования Фурье.

При установке дополнительных расширений (toolbox'ов) возможно подключение такого функционала, как многомерный анализ данных, спектральный анализ, работа с изображениями, подбор поверхностей для описания данных (рисунок 1.4) и другие [9].

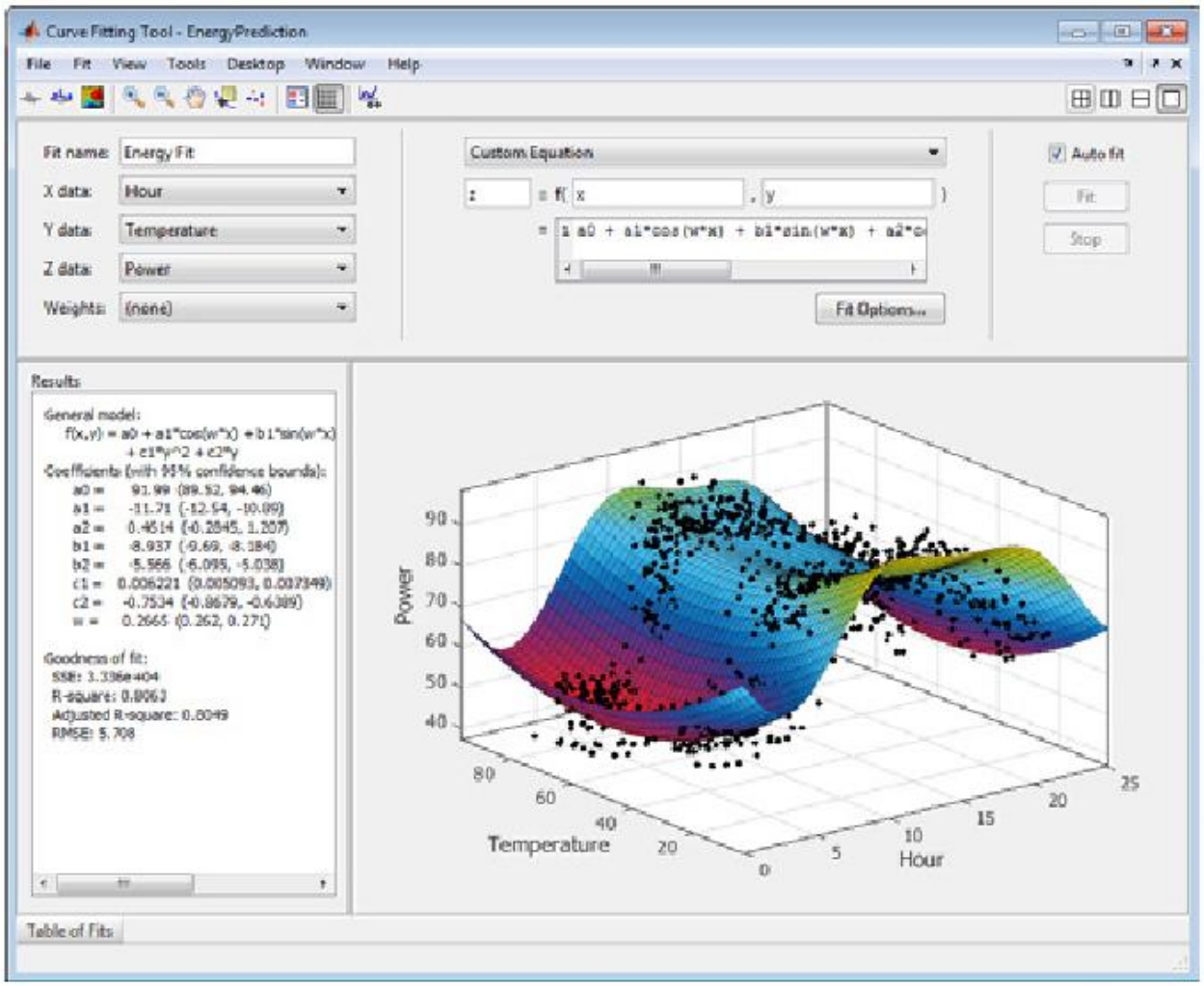


Рисунок 1.4 – Аппроксимация данных с помощью модуля Curve Fitting

В математическом пакете Matlab предусмотрена возможность построения двухмерных и трехмерных графиков, кроме того доступен

перечень функций, для визуализации данных (построение гистограмм, диаграмм различных видов).

Для каждого типа графиков в Matlab предусмотрена возможность просмотра примеров (рисунок 1.5). Любой график можно сохранить во все распространенные векторные и растровые форматы изображений [10-12].

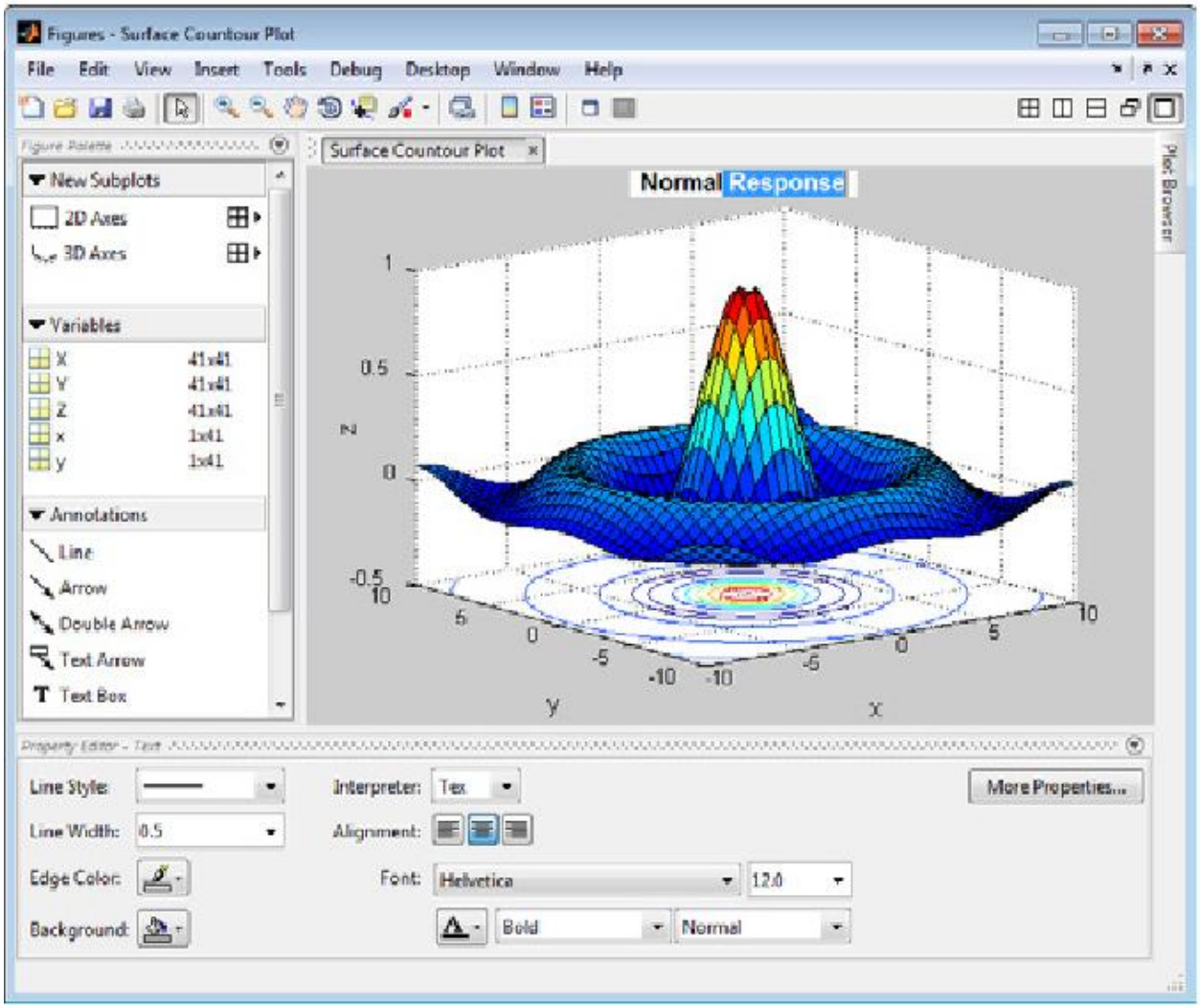


Рисунок 1.5 – Редактирование отображения графической области данных

Также в данном математическом пакете предусмотрена возможность генерирования законченных отчетов по проведенным вычислениям с последующим сохранением в таких форматах как word, html, latex, а также pdf. При этом в отчет можно включить программный код вместе с комментариями, вычисления и графики.

В среду Matlab встроен высокоуровневый язык программирования и все необходимые средства для отладки программного кода. Особенностью языка программирования является поддержка выполнения операций над матрицами и векторами. При программировании вычислений не требуется объявление переменных и определение их типов (рисунок 1.6). Отсутствует необходимость использования цикла for при работе с элементами матриц и векторами [13-16].

Одновременно с этим, встроенный в Matlab язык поддерживает приемы объектно-ориентированного программирования, возможность обработки ошибок и управление потоками. Также существует возможность задавания пользовательских типов данных с применением различных структур.

В Matlab существует возможность построчного выполнения команд в интерактивном режиме.

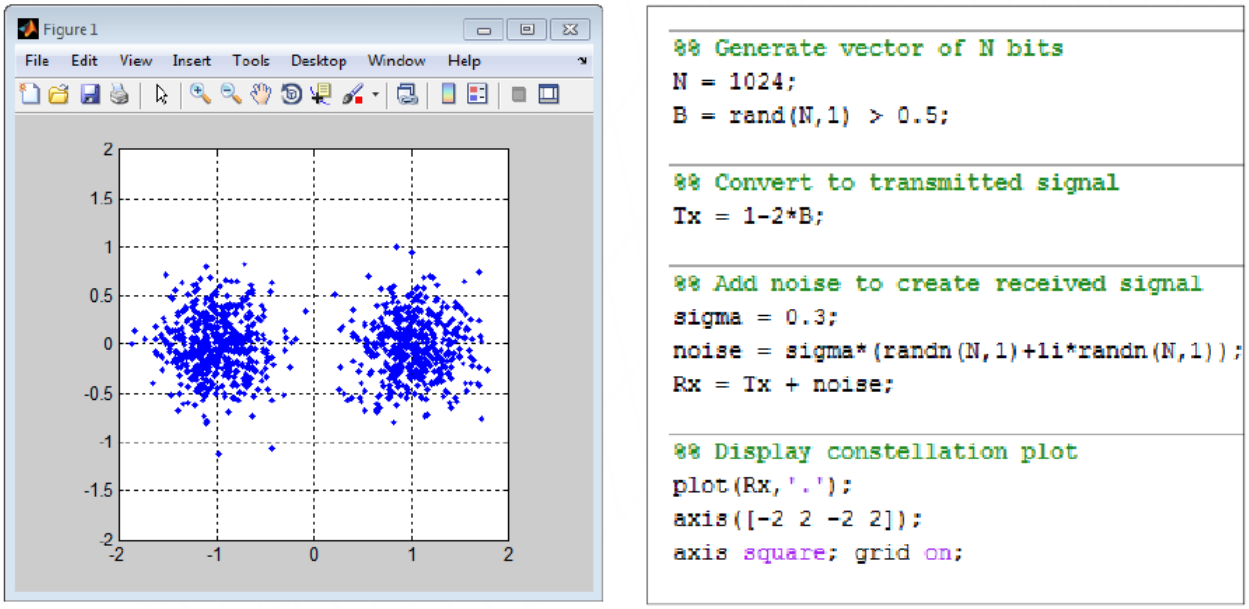


Рисунок 1.6 – Пример программной генерации данных в Matlab 2017b

Средства разработки программного кода в среде Matlab состоят из следующих компонентов:

1. Окно для ввода команд (Command Window).
2. Редактор кода (Editor) (рисунок 1.7).
3. Инструмент анализа кода (Code Analyzer).
4. Средство профилировки (Profiler).

Окно ввода команд предназначено для интерактивного выполнения действий над данными и просмотр результатов, выполненных команд. Редактор кода необходим для составления m-функций и построчного выполнения кода. Инструмент анализа кода предназначен для его отладки, поиска ошибок и оптимизации. Средство профилировки необходимо для анализа быстродействия кода и определение тех его частей, которые работают недостаточно быстро.

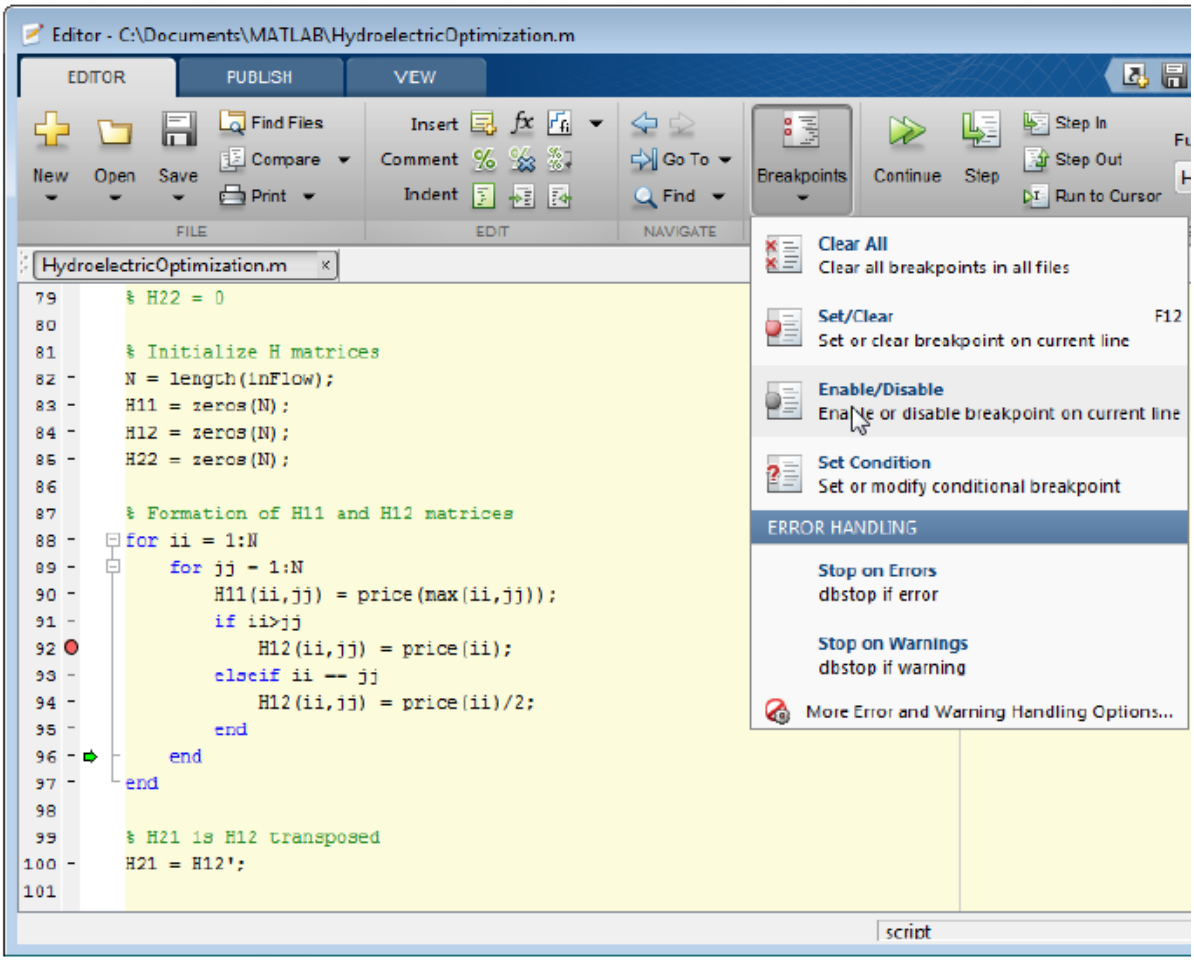


Рисунок 1.7 – Отладка программного кода в Editor

Matlab поддерживает интеграцию с наиболее известными языками программирования. Реализована возможность использования вставок кода на языках C++, NET и Java. Также с помощью библиотеки Engine возможен вызов кода Matlab программ, написанных на Fortran и C++.

При обработке программного кода математический пакет Matlab использует оптимизированную под матричные вычисления библиотеку и JIT (just in time) компилятор. Разработчики Matlab отмечают что такой подход обеспечивает производительность скорость вычислений сопоставимую с классическими языками программирования [17-20].

В Matlab реализована функция автоматического разделения вычислений на несколько потоков, что позволяет получить преимущество в скорости в системах с несколькими ядрами.

При установке toolbox для обработки параллельных вычислений становиться возможным, при математическом моделировании, использование мощностей графических процессоров (GPU), а также компьютерных кластеров. Разработчики Matlab отмечают, что для распараллеливания вычислений практически не требуется изменять программный код.

В математическом пакете Matlab реализованы удобные средства по проектированию и развертыванию приложений. Так в Matlab реализована функция публикации разработанного приложения или алгоритма на сервере Mathwork, что открывает доступ к данным разработкам другим пользователям Matlab. Другие пользователи могут комментировать предложенные автором алгоритма решения, помогать в исправлении ошибок, оптимизировать программный код.

Также существует функция компилирования разработанного приложения для передачи пользователям, у которых отсутствует Matlab.

В Matlab существуют специальные средства разработки графического интерфейса–среда Graphical User Interface Development Environment (GUIDE).

GUIDE позволяет прототипировать/проектировать графический интерфейс приложения. GUIDE поддерживает наиболее распространенные элементы управления формами – списки, поля со списком, кнопки, меню в верхней окна, надписи и др (рисунок 1.8). Существует возможность программного задания всех элементов управления формой [21].

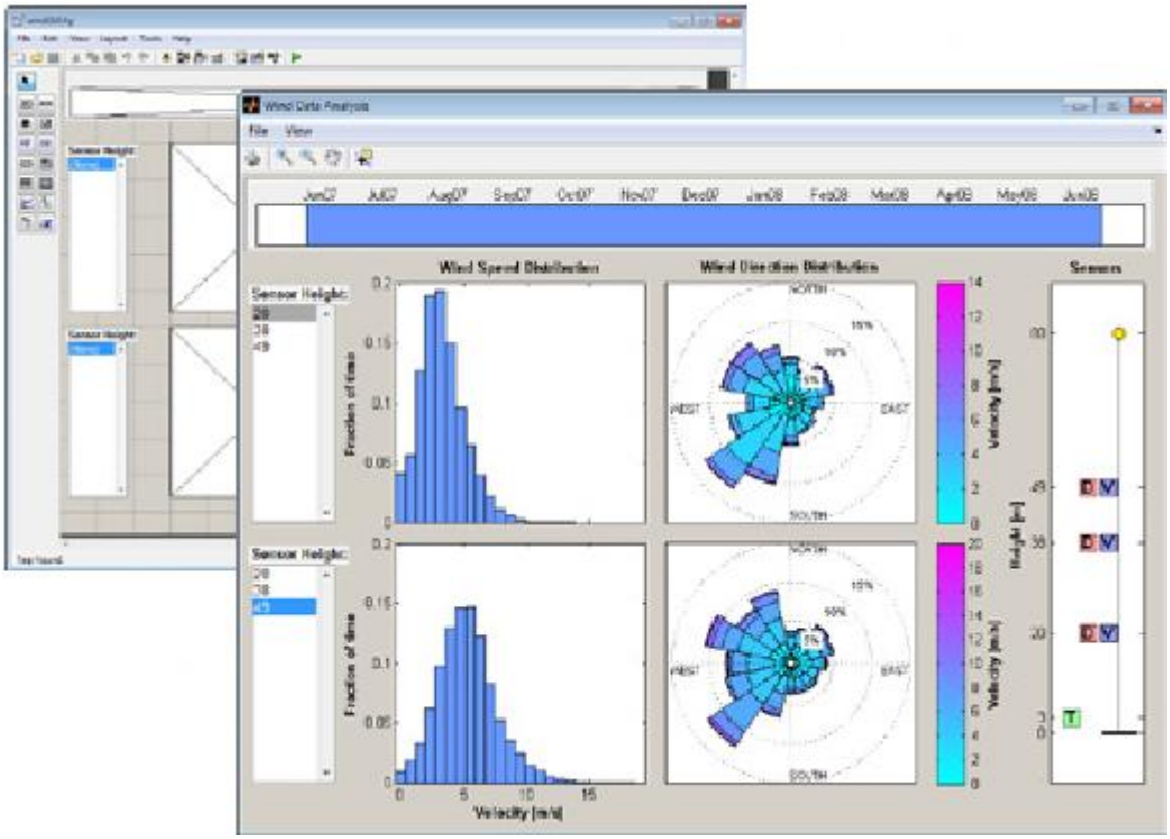


Рисунок 1.8 – Разработка графического интерфейса в GUIDE и результат запуска приложения с графическим интерфейсом

В среде Matlab существует свой упаковщик приложений, позволяющий объединить программный код, графический интерфейс и документацию в единственный файл.

Инструменты Matlab по развёртыванию приложений позволяют генерировать автономные компоненты, предназначенные для интеграции в приложения, написанные на C/C++, Net, Java (рисунок 1.9, 1.10).

Компонент Production Server обеспечивает возможность написанные на Matlab программы в производственных системах, веб-приложениях, базах данных и корпоративных системах.

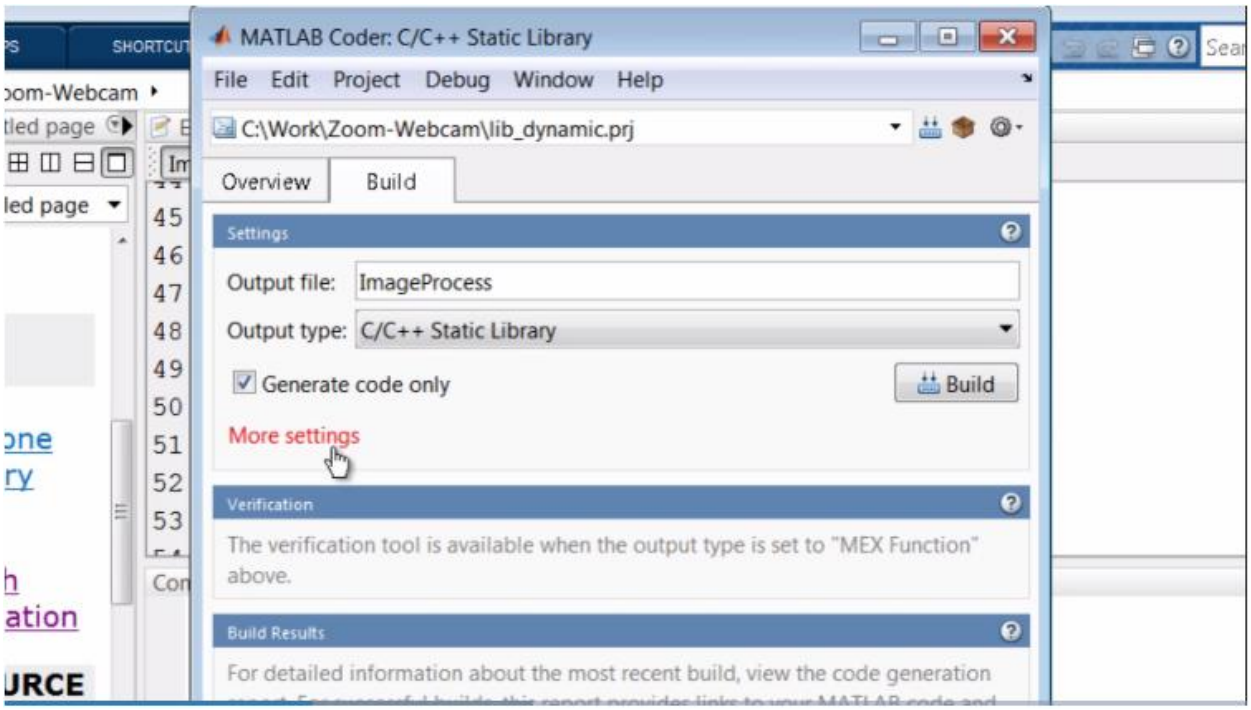


Рисунок 1.9 – Генерирование автономной библиотеки

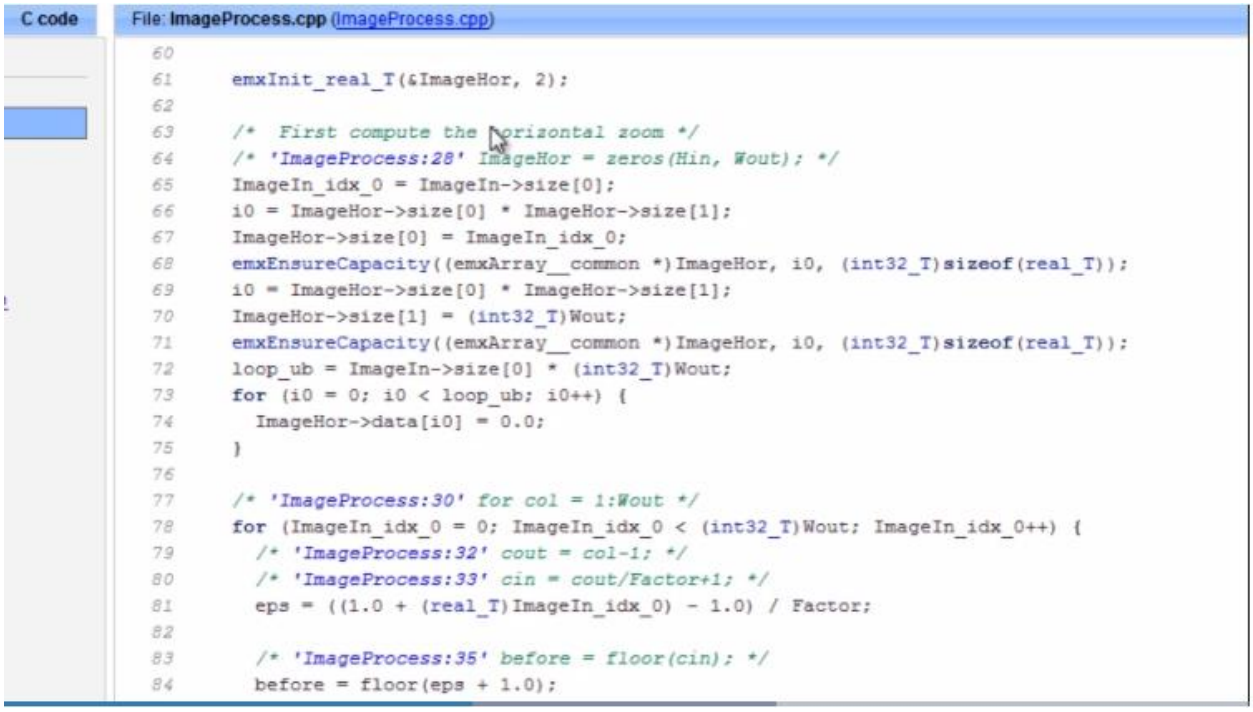


Рисунок 1.10 – Вид генерируемого программного кода

1.2 Анализ реализаций алгоритмов искусственного интеллекта в Matlab

Проведём анализ дополнений (toolboxes), применяемых в Matlab для реализации алгоритмов искусственного интеллекта.

Дополнения, реализующие алгоритмы искусственного интеллекта:

- Neural Network Toolbox (нейронные сети);
- Optimization Toolbox (методы оптимизации);
- Statistic and Machine Learning Toolbox (методы статистики и машинного обучения);
- Fuzzy Logic Toolbox (методы нечеткой логики).

Модуль Neural Network Toolbox (рисунок 1.11) предназначен для моделирования искусственных нейронных сетей, которые применяются при решении задач синтеза систем управления и диагностики техпроцессов, кластеризации и аппроксимации данных, прогнозирования, а также распознавания образов [22].

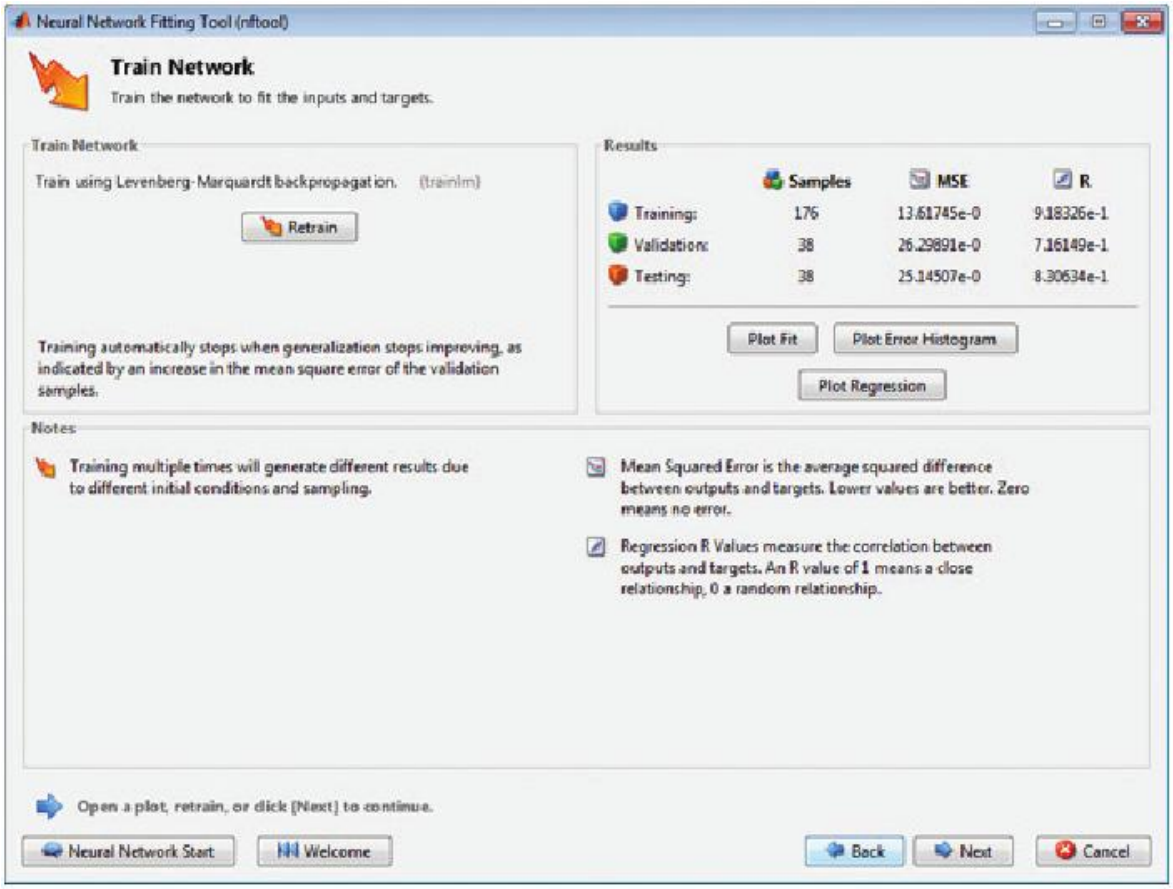


Рисунок 1.11 – Графическая оболочка Neural Network Toolbox

Основными особенностями Neural Network Toolbox является:

• Реализация возможности обучения с учителем (рисунок 1.12) таких нейронных сетей, как рекуррентные сети, радиально-базисные сети, многослойные сети, LVQ-сети, NARX-сети, сети с задержкой.

• Реализация возможности обучения без учителя таких нейронных сетей, как самоорганизующиеся карты, конкурентные сети.

• Наличие примеров использования нейронных сетей для решения задач аппроксимации и кластеризации данных, а также распознавания образов.

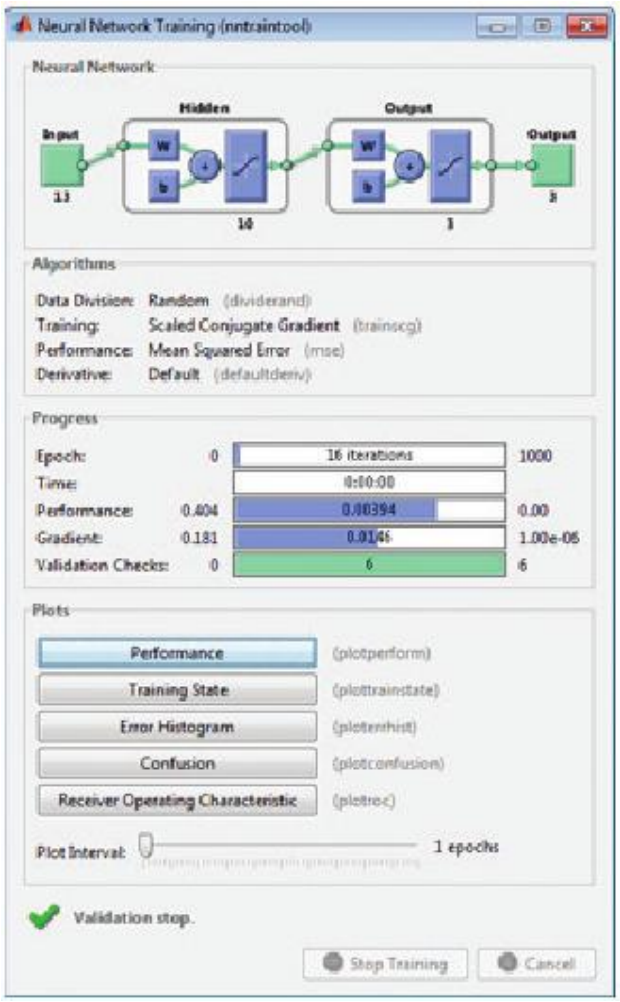


Рисунок 1.12 – Окно Neural Network Toolbox с демонстрацией процесса обучения

Основные характеристики Optimization Toolbox (рисунок 1.13):

- Совместимость с модулем Parallel Computing Toolbox, обеспечивающим распараллеливание вычислений и поддержку графических процессоров.
- Автоматическая нормировка данных для повышения точности и скорости обучения нейронных сетей.
- Добавление блоков Simulink для использования нейронных сетей в приложениях систем управления.

Для решения задач оптимизации (в том числе задач большой размерности) в Matlab имеется модуль Optimization Toolbox. Представленные в нем алгоритмы позволяют решать оптимизационные задачи с ограничениями и без ограничений. Имеется поддержка работы с непрерывными и дискретными параметрами.

1. Графическая оболочка для решения задач оптимизации с возможностью наблюдения за ходом ее решения.
2. Наличие реализаций методов многоцелевой нелинейной оптимизации.
3. Наличие реализаций методов наименьших квадратов, сглаживания данных.
4. Наличие реализаций методов для решения задач квадратичного и линейного программирования.
5. Наличие реализаций методов для решения задач бинарного целочисленного программирования.
6. С точки зрения искусственного интеллекта реализована возможность решения задач с применением генетических алгоритмов.
7. Для решения нелинейных задач оптимизации с ограничениями реализована возможность распараллеливания вычислений.
8. Реализована возможность автоматической генерации кода на языке Matlab для решения поставленной задачи.

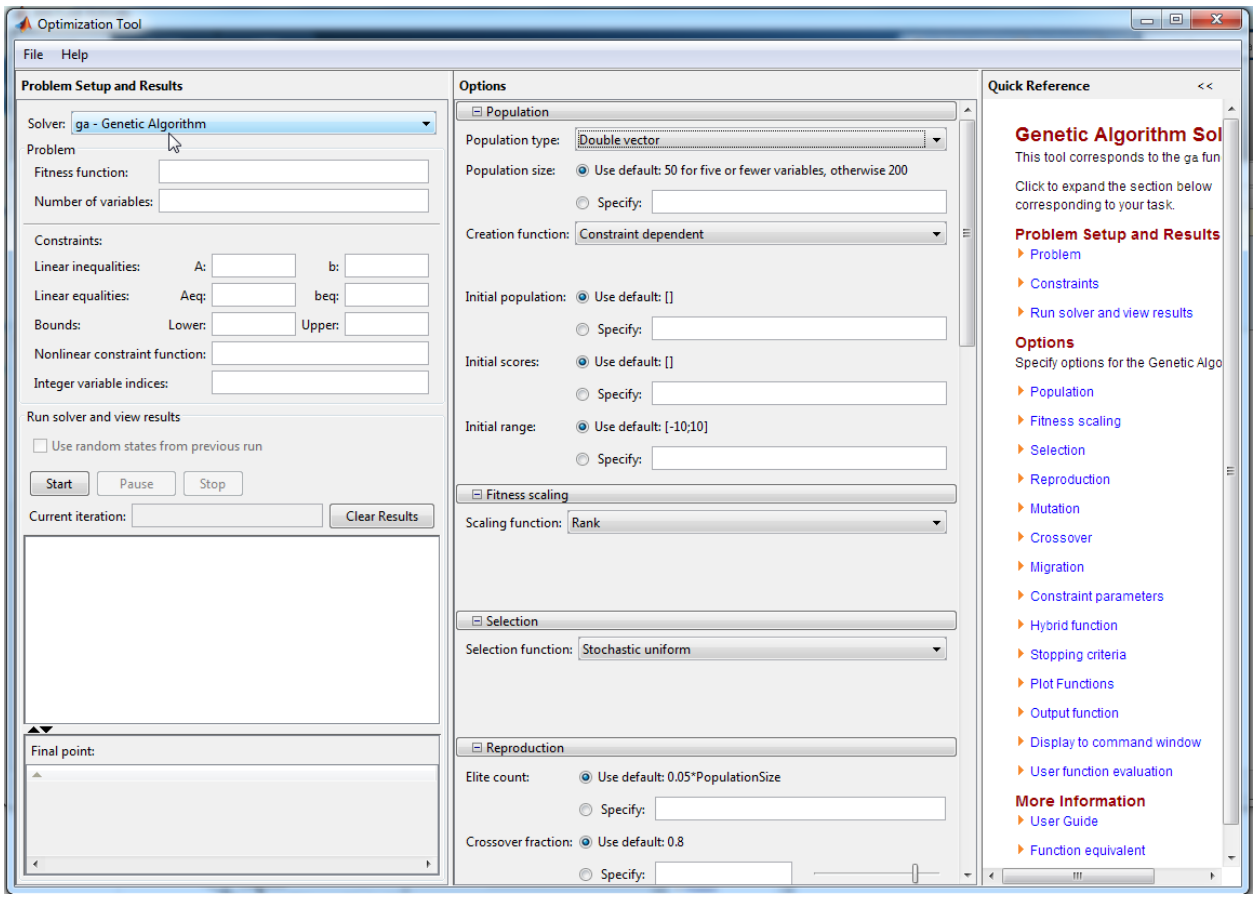


Рисунок 1.13 – Графическая оболочка Optimization toolbox

Модуль Statistics and Machine Learning Toolbox представляет собой набор алгоритмов и реализаций методов для организации, анализа и моделирования данных. Реализованные в данном модуле инструменты позволяют строить регрессионные, классификационные модели, а также строить статистические графики для исследования данных, проверки научных гипотез [23-26].

Модуль Statistics and Machine Learning Toolbox позволяет проводить многомерный анализ данных, определять степень влияния входных параметров на выходные переменные, проводить требуемые преобразования данные, применить регуляризацию, сжатие данных. Также модуль поддерживает построение регрессионных моделей с использованием метода дробных наименьших квадратов.

Основными характеристиками модуля Модуль Statistics and Machine Learning Toolbox являются:

- Реализация возможности хранения разнородных данных в массивах.
- Реализация методов построения регрессионных моделей – линейных, нелинейных, робастных и др.
- Реализация методов классификационных моделей – деревья принятия решений, kNN-модель, модель, основанная на линейном дискриминантном анализе (рисунок 1.14).
- Реализация методов дисперсионного анализа.
- Реализация расчета распределения вероятностей, связки и смеси нормальных распределений.
- Реализация методов генерирования случайных чисел.
- Реализация методов для проверки гипотез.
- Реализация методов для планирования вычислительных экспериментов и статистической обработки данных.

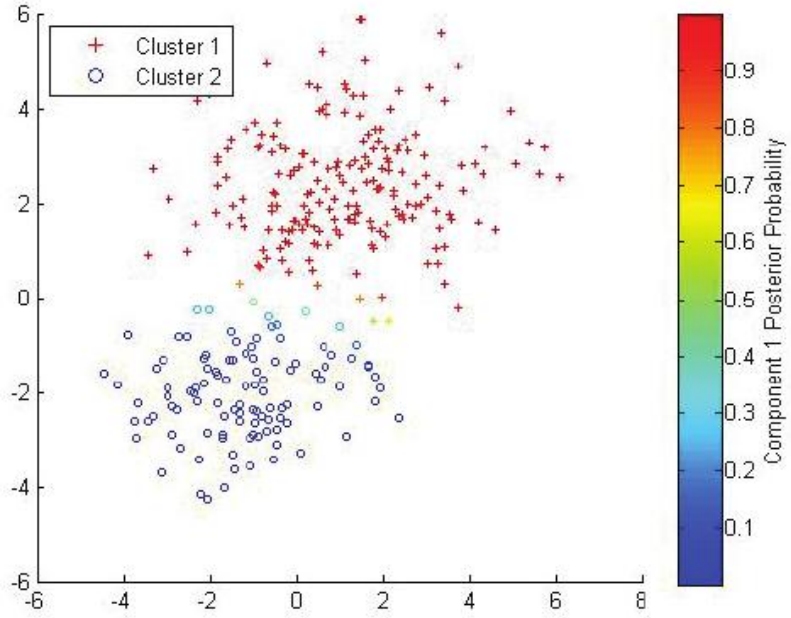


Рисунок 1.14 – Кластерный анализ с использованием Statistic and machine learning toolbox

Модуль Fuzzy Logic Toolbox представляет собой набор функций, графических инструментов для анализа, проектирования и моделирования нечетких систем управления. С помощью данного инструмента вы можете осуществить все шаги проектирования систем с нечеткой логикой. Функции предназначены для многих распространенных методов, в том числе для нечеткой кластеризации и адаптивного нечеткого обучения нейронных сетей. Набор инструментов позволяет моделировать поведение сложных систем с помощью простых логических правил, а затем реализовать эти правила в системе нечеткого вывода. Вы также можете использовать данный инструмент как самостоятельно исполняемое ядро для задач нечеткого вывода. Кроме того, вы можете использовать блоки нечеткого вывода в Simulink и моделировать нечеткие системы в рамках комплексной модели всей динамической системы.

Ключевые особенности:

- Специализированный графический пользовательский интерфейс (GUI) для построения систем нечеткого вывода, а также просмотра и анализа результатов (рисунок 1.15);
- Функции принадлежности для создания систем нечеткого вывода;
- Поддержка логики И, ИЛИ и НЕ в пользовательских правилах;
- Стандартные системы нечеткого вывода типа Мамдани и Сугено;
- Автоматизированное формирование функций принадлежности через адаптивное нейросетевое обучение и методы нечеткой кластеризации;
- Возможность встраивать нечеткую систему логического вывода в модели Simulink;
- Возможность генерировать встраиваемый С-код или автономно исполняемое ядро нечеткого вывода.

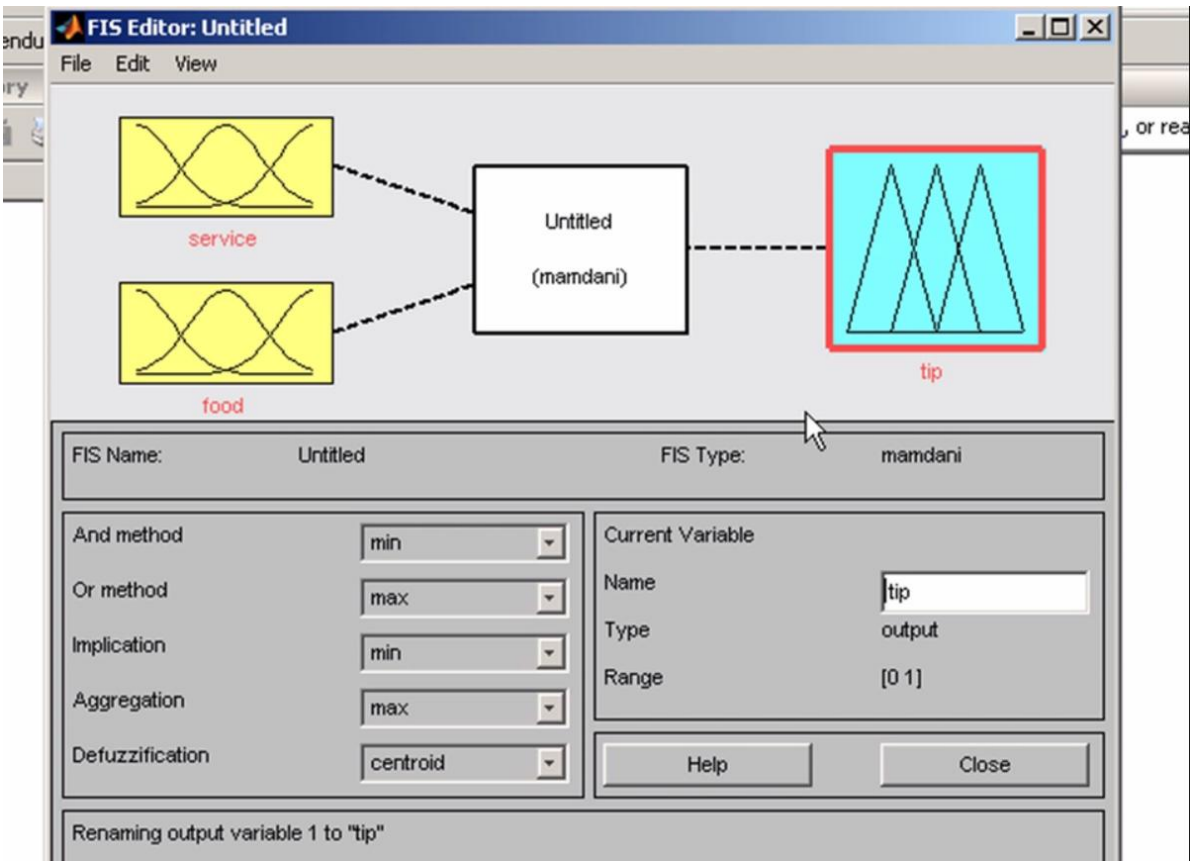


Рисунок 1.15 – Системы нечеткого управления с использованием Fuzzy Logic Toolbox

Представим в виде таблицы 1 результаты анализа реализаций алгоритмов искусственного интеллекта в модулях (toolbox) Matlab.

Таблица 1 – Реализации алгоритмов искусственного интеллекта в Matlab

Название алгоритма	**Решаемые задачи**	**Название Toolbox**	**Недостатки реализации**
Группа однослойных нейронных сетей прямого распространения	Классификация, регрессионный анализ, кластеризация	Neural Network Toolbox	Отсутствуют
Группа многослойных нейронных сетей прямого распространения	Классификация, регрессионный анализ	Neural Network Toolbox	Реализован только ограниченный набор архитектур: LVQ (2 слоя), радиально-базисная сеть (2 слоя), сеть с прямым распространением сигнала (2 слоя), сеть с запаздыванием - TDNN (2 слоя). Нейронные сети другой архитектуры или с большим количеством слоев необходимо реализовывать самостоятельно.
Группа рекуррентных нейронных сетей	Классификация	Neural Network Toolbox	Реализован только ограниченный набор архитектур: нейронная сеть Элмана (2 слоя), сеть Хопфилда (1 слой). Нейронные сети другой архитектуры или с большим количеством слоев необходимо реализовывать самостоятельно.
Генетические алгоритмы	Оптимизация	Optimization toolbox	Невозможность работы с дискретными параметрами через реализованные функции.
k-means	Кластеризация	Statistic and machine	Реализована работа только с числовыми

		learning toolbox	параметрами
kNN	Классификация	Statistic and machine learning toolbox	Невозможность работы с дискретными параметрами через реализованные функции.
SVM	Бинарная классификация	Statistic and machine learning toolbox	Отсутствуют
CART	Классификация, регрессионный анализ	Statistic and machine learning toolbox	Реализована возможность построения деревьев классификации и регрессии, однако входными параметрами могут являться только числовые признаки.
ID3	Классификация	-	Отсутствует реализация алгоритма
C4.5	Классификация	-	Отсутствует реализация алгоритма
Системы нечеткого вывода	Логический вывод	Fuzzy Logic Toolbox	Отсутствуют
c-means	Кластеризация	Fuzzy Logic Toolbox	Невозможность работы с дискретными параметрами через реализованные функции.
Apriori	Аффинитивный анализ	-	Отсутствует реализация алгоритма

Таким образом, математический пакет Matlab состоит из высокоуровневого языка программирования и интерактивной среды разработки приложений. Matlab предназначен для выполнения численных расчетов и наглядной визуализации полученных результатов. Возможности Matlab позволяют выполнять анализ исходных данных, разрабатывать алгоритмы, составлять математические модели.

Алгоритмы искусственного интеллекта реализованы в таких модулях Matlab как:

- Neural Network Toolbox,
- Optimization toolbox,
- Statistic and machine learning toolbox,
- Fuzzy Logic Toolbox.

Однако представленные в них реализации имеют не полную поддержку алгоритмов (таблица 1).

Так как в настоящее время не существует официальной поддержки алгоритма C4.5 математическим пакетом Matlab, в рамках данного исследования реализация данный алгоритма в среде Matlab.

Таким образом актуальной можно признать цель исследования – разработка реализации алгоритма машинного обучения C4.5 в математическом пакете Matlab.

Поставленная цель достигается путем последовательного решения следующих задач:

1. Провести анализ реализаций алгоритмов искусственного интеллекта в математическом пакете Matlab.

2. Изучить математический аппарат алгоритма C4.5.

3. Спроектировать и разработать реализацию алгоритма C4.5 в математическом пакете Matlab.

4. Протестировать реализацию алгоритма C4.5.

ГЛАВА 2 МАТЕМАТИЧЕСКИЙ АППАРАТ АЛГОРИТМА C4.5

2.1 Формальное описание задачи классификации

Формально задача классификации описывается следующим образом. Существует множество входных переменных $\boldsymbol{X}$ – описаний объектов, а также множество $\boldsymbol{Y}$ (конечное) обозначений классов. В задаче классификации существует неизвестная целевая зависимость – отображение y^*: $\boldsymbol{X} \to \boldsymbol{Y}$, значения которой известны только на объектах из обучающей выборки $\boldsymbol{X}^m$:

$$X^m = (x_1, y_1), ..., (x_m, y_m) \ , \tag{2.1}$$

где m – размер обучающей выборки, x_i – вектор значений входных переменных для i-го объекта из обучающей выборки, y_i – значение метки класса для i-го объекта из обучающей выборки. При этом вектор x_i описывается набором значений атрибутов A, а y_i может принимать одно значений меток класса $C_1, C_2, ... C_k$:

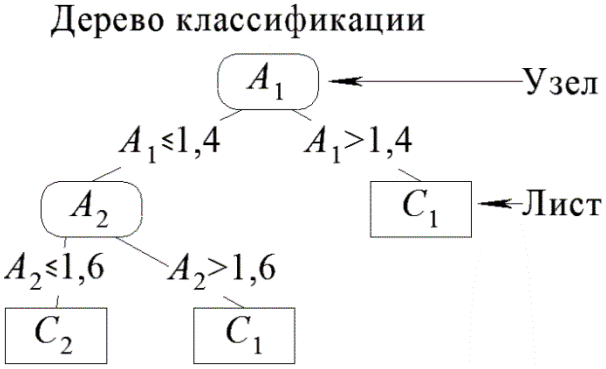
$$\begin{cases} x_i = (A_1, A_2, ..., A_n) \\ y_i \in C_1, C_2, ... C_k \end{cases}, \tag{2.2}$$

где, n – количество атрибутов, описывающих объект из обучающей выборки, k – количество меток класса в обучающей выборке.

Требуется построить алгоритм α: $\boldsymbol{X} \to \boldsymbol{Y}$, способный классифицировать объект описываемый вектором входных значений x, при $x \in X$.

2.2 Деревья принятия решения как модель классификации данных

Одним из возможных представлений модели классификации данных является дерево принятия решений. Дерево принятия решений – это связный ациклический граф, представляющий правила классификации в иерархической последовательной структуре. Дерево состоит узлов и листов. В каждом узле расположена проверка по одному из атрибутов вектора x_i. В каждом листе расположено одно из значений меток класса $C_1, C_2, ... C_k$. Чтобы классифицировать любой из объектов обучающей выборки надо спуститься по дереву до листа и выдать значение метки класса листа. В зависимости от значений атрибутов классифицируемого объекта спускаясь по дереву можно прийти к различным листам (рисунок 2.1).



Обучающая выборка

A_1	A_2	y
1,1	1,8	C_1
1,4	1,6	C_2
1,6	1,4	C_1
1,8	1,1	C_1
1,3	1,5	C_2

Рисунок 2.1 – Пример обучающей выборки и соответствующее ей дерево принятия решений

Общий принцип построения деревьев решений, заключается в рекурсивном разбиении множества объектов T из обучающей выборки на подмножества, содержащие объекты, относящиеся к одинаковым классам.

Относительно обучающей выборки T и множества классов $C_1, C_2, ... C_k$ возможны три ситуации:

- Множество T содержит один или более объектов, относящихся к одному классу C_i. Тогда дерево решений для T - это лист, определяющий класс C_i;

• Множество T не содержит ни одного объекта (пустое множество). Тогда данное множество T – это лист, и класс, ассоциированный с листом, выбирается из другого множества, отличного от T, например, из множества, ассоциированного с родителем;

• Множество T содержит объекты, относящиеся к разным классам. В этом случае следует разбить множество T на некоторые подмножества. Для этого выбирается один из атрибутов $A_1, A_2, ...A_k$ и T разбивается на подмножества T_1, T_2, … , T_j, по выбранному атрибуту и соответствующему ему условию.

Эти шаги повторяются рекурсивно для всех получающихся подмножеств до тех пор, пока все подмножества не будут объявлены листами.

Все алгоритмы автоматического построения деревьев принятия решений относятся к индуктивным алгоритмам машинного обучения (и к подобласти “обучения с учителем”).

Одним из важных вопросов построения дерева принятия решений является выбор таких условий в узлах, которые обеспечат наилучшее разбиение обучающей выборки на подмножества. Это необходимо, чтобы получившееся дерево было, во-первых, компактным (содержало наименьшее количество узлов), а во-вторых обеспечивало максимальную точность классификации.

2.3 Математический аппарат построения дерева принятия решения по алгоритму C4.5

Алгоритм C4.5 для построения дерева принятия решения выбирает из всех возможных вариантов такую проверку в узле, которая обеспечивают максимальное снижение информационной двоичной энтропии и с учетом равномерного распределения объектов по подмножествам.

Рассмотрим этот вопрос подробней на множестве T объектов, содержащих k меток класса – $C_1, C_2, ...C_k$. Если перемешать объекты из

множества T и расположить их в ряд, то мы реализуем операцию под названием "перестановка". При этом считается, что объекты с одинаковыми метками класса идентичны. Тогда количество уникальных перестановок для множества рассчитывается, как:

$$W = \frac{N!}{N_1! \cdot N_2! \cdot \ldots} = \frac{N!}{\prod_{i=1}^{k} N_i!}, \tag{2.3}$$

где N_i – количество объектов с меткого класса C_i в множестве T, N – количество объектов в обучающей выборке.

Если все уникальные перестановки пронумеровать числами от 0 до $W-1$, то количество бит необходимое для кодирования каждого уникального варианта перестановки равно $\log_2(W)$. Среднее количество бит на каждую перестановку называется комбинаторной энтропией и рассчитывается как:

$$S = \frac{\log_2(W)}{N} = \frac{1}{N} \cdot \log_2\left(\frac{N!}{\prod_{i=1}^{k} N_i!}\right) \tag{2.4}$$

Важное свойство комбинаторной энтропии, которое используется при построении дерева принятия решения – чем меньше энтропия, тем однороднее множество. Самые однородные множества в дереве принятия решений – это листы, не содержащие примесей других классов. Самое неоднородное множество – начальная обучающая выборка.

Для удобства расчета энтропии выражение, приведенное выше, можно упростить, воспользовавшись формулой Стирлинга:

$$\ln N! = N \cdot \ln N - N + O(\ln N) \approx N \cdot \ln N - N \tag{2.5}$$

Тогда энтропию для множества T с учетом упрощений можно рассчитать так:

$$S = -\sum_{i=1}^{k} \left(\frac{N_i}{N} \cdot \log_2 \frac{N_i}{N}\right) \tag{2.6}$$

С учетом того, что значение энтропии *S* зависит от рассматриваемого множества *T*, выразим ее как функцию:

$$S(T) = -\sum_{i=1}^{k}\left(\frac{freq(T,C_i)}{|T|}\cdot \log_2 \frac{freq(T,C_i)}{|T|}\right), \quad (2.7)$$

где $freq(T,C_i)$ – функция, которая возвращает количество объектов с меткой класса C_i в множестве *T*.

При построении дерева принятия решения в каждый узел помещается условие, которое разбивает исходное множество *T* на несколько *j* подмножеств:

$$T = T_1 \cup T_2 \cup ... \cup T_j \quad (2.8)$$

Тогда для каждого подмножества $T_1, T_2, ..., T_j$ по формуле 2.7 можно рассчитать энтропию.

Таким образов энтропию множества T после разбиения на подмножества рассчитывается как:

$$S_0(T, T_1, T_2, ..., T_j) = \sum_{i=1}^{j}\left(\frac{|T_i|}{|T|}\cdot S(T_i)\right) \quad (2.9)$$

Если из энтропии *S*(*T*) вычесть энтропию разбиения $S_0(T, T_1, T_2, ..., T_j)$, то можно рассчитать прирост информации $G(T, T_1, T_2, ..., T_j)$, обеспечивающийся данным условием в узле:

$$G(T, T_1, T_2, ..., T_j) = S(T) - \sum_{i=1}^{j}\left(\frac{|T_i|}{|T|}\cdot S(T_i)\right) \quad (2.10)$$

Чем однородней получаются подмножества в результате разбиения, тем больший прирост информации обеспечивает данное разбиение.

Однако, если на каждом шаге выполнения выбирать проверку в узле, которая обеспечивает максимальный прирост информации:

$$G(T, T_1, T_2, ..., T_j) \to \max \quad (2.11)$$

то это приводит к проблеме «переобучения» модели классификации. Это означает, что при построении дерева принятия решений, в результате

ассиметричных разбиений, листы будут получаться путем отсеивания в них по несколько объектов. Конечное дерево в этом случае будет содержать большое количество листов, каждый из которых будет распространяться только по несколько объектов обучающей выборки. Такое дерево будет работать с низкой точностью, на примерах, не содержащихся в обучающейся выборке.

Для решения этой проблемы в алгоритме C4.5 значение прироста информации нормируется параметром *R*, который рассчитывается по (2.12):

$$R(T,T_1,T_2,...,T_j) = -\sum_{i=1}^{j}((|T_i|/|T|)\cdot \log_2(|T_i|/|T|)). \qquad (2.12)$$

Чем более равномерные по количеству объектов получаются подмножества, тем меньше значение *R*.

Нормировка прироста информации *G* с использованием параметра *R* осуществляется следующим образом (2.13):

$$G_{\text{норм}}(T,T_1,T_2,...,T_j) = \frac{G(T,T_1,T_2,...,T_j)}{R(T,T_1,T_2,...,T_j)} \qquad (2.13)$$

Таким образом, для каждого узла дерева помещаемое в него условие должно обеспечивать максимальное значение нормированного прироста информации (2.13):

$$G_{\text{норм}}(T,T_1,T_2,...,T_j) \to \max. \qquad (2.14)$$

Такой выбор условий для каждого узла обеспечивает максимальное снижение энтропии при сбалансированном (по количеству объектов) разбиении исходного множества на подмножества.

Энтропия будет равна нулю, если в результате построения дерева принятия решения удалось достигнуть листов без содержания примесей посторонних классов.

Для каждого категориального атрибута существует только один вариант разбиения – когда исходное множество *T* делиться на подмножества в количестве равном значений данного атрибута.

Для каждого числового атрибута существует несколько вариантов разбиений, количество которых равно мощности множества порогов. Множество порогов находится путем записи всех уникальных значений данного числового атрибута в обучающей выборке в порядке возрастания без повторов с отбрасыванием наибольшего значения. Затем каждое значение порога *z* выступает в роли точки деления исходного множества *T,* состоящего из объектов *t,* на подмножества T_1 и T_2 (2.15):

$$\begin{cases} T_1 = \ t \in T \mid a(t) \le z \\ T_2 = \ t \in T \mid a(t) > z \end{cases}, \tag{2.15}$$

где, *a* – значение атрибута, по которому производится разбиение.

В конечном итоге из всех возможных вариантов разбиения выбирается то, которое соответствует критерию (2.11). Как было сказано выше, построение дерева ведется до тех пор, пока не будут найдены все листы.

ГЛАВА 3 ПРАКТИЧЕСКАЯ РЕАЛИЗАЦИЯ ПРЕДЛОЖЕННЫХ РЕШЕНИЙ

3.1 Программная реализация алгоритма

В ходе выполнения исследований по выпускной квалификационной работе в среде в математическом пакете Matlab на встроенном языке программирования была разработана и спроектирована программа для построения деревьев принятия решений в соответствии с алгоритмом C4.5.

Функциональные возможности программы:

- Построение модели классификации в виде дерева принятия решений в соответствии с алгоритмом C4.5;
- сохранение построенного дерева принятия решений во вложенных структурах данных;
- возможность тестирования построенных деревьев принятия решений для определения их точности работы;
- возможность задания параметра ранней остановки построения дерева решений – допустимый процент ошибки классификации;
- использования построенного дерева принятия решения для классификации объектов из тестовой выборки данных.

Проект разработанного приложения в редакторе математического пакета Matlab представлен на рисунке 3.1.

Программная реализация на языке Matlab представлена в приложении А.

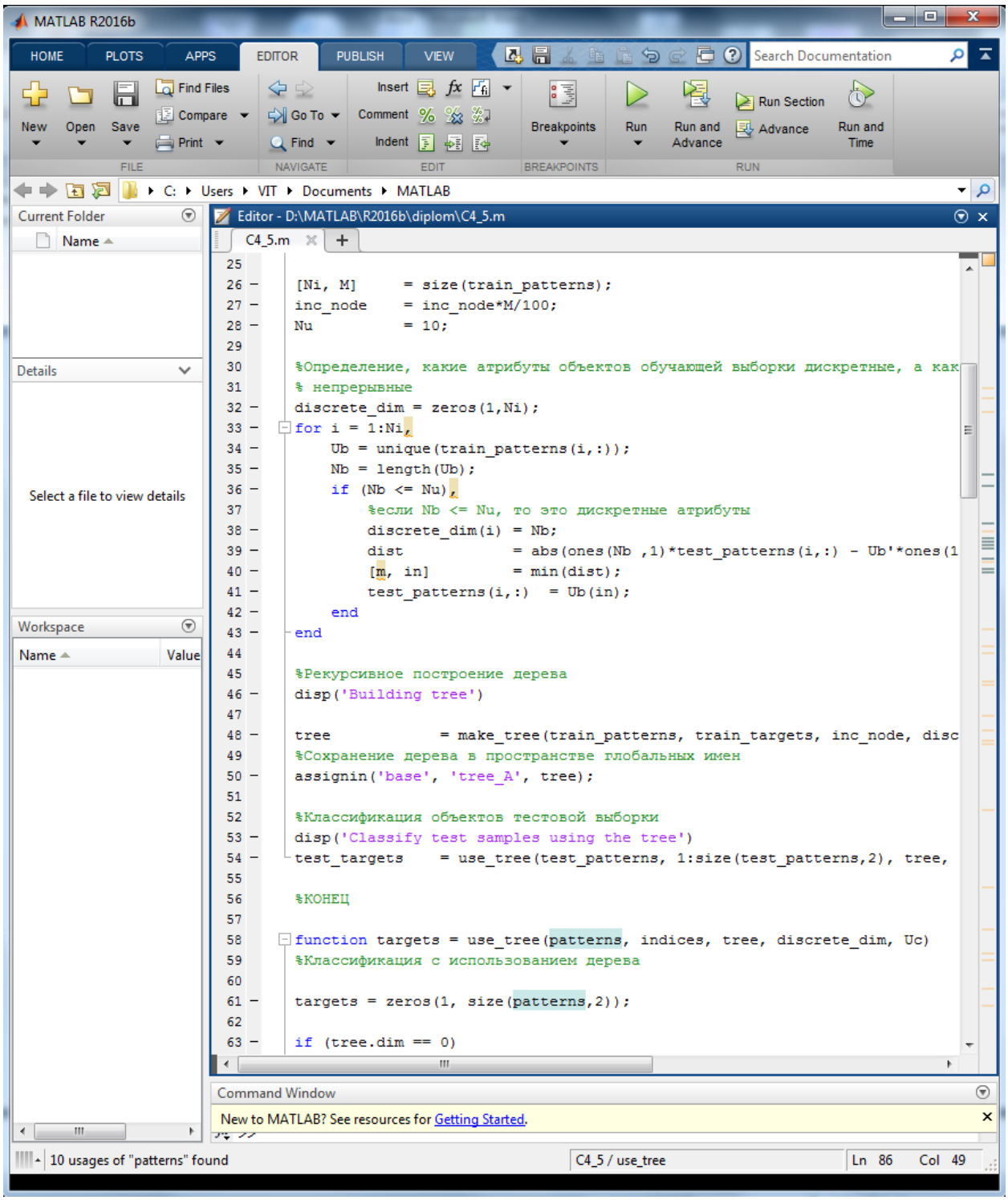


Рисунок 3.1 – Программный код разработанных функций в редакторе кода (editor) математического пакета Matlab

3.2 Пример использования

Для подключения разработанных функций к текущему проекту необходимо в командной строке Matlab указать путь, по которому находиться m-файл с функциями, реализующими алгоритм C4.5 (рисунок 3.2).

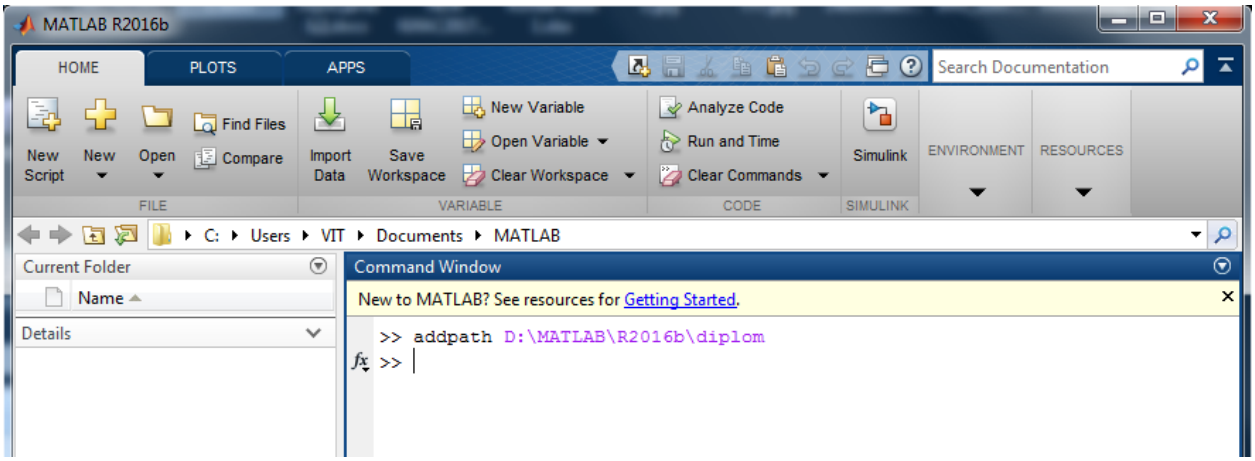


Рисунок 3.2 – Подключение разработанных функций к текущему проекту через командную строку Matlab

Для построения модели классификации по алгоритму C4.5 в среде Matlab необходимо определить 3 глобальных переменных:

- числовую матрицу, содержащую входные параметры объектов обучающей выборки (training_p);
- вектор, содержащий соответствующие номера (в виде чисел) классов для объектов обучающей выборки (training_t);
- числовую матрицу, содержащую входные параметры тестируемых объектов (test_p).

Предположим, что нам требуется построить модель классификации по обучающей выборке, представленной в таблице 2. И протестировать работу полученного дерева на первых двух объектах обучающей выборки.

Для этого случая переменные training_p, training_t, test_p должны быть определены следующим образом (рисунок 3.3, 3.4, 3.5). Чтобы указать, что

атрибуты A1, A2 должны рассматриваться как категориальные признаки, необходимо в программном коде значение переменной Nu указать равное 4.

Параметр Nu определяет, какие атрибуты объектов считать дискретными, а какие непрерывными. Если количество уникальных значений атрибута меньше либо равно значению Nu, то считается, что атрибут – дискретный (Приложение А).

Таблица 2 – Обучающая выборка: A1, A2, A3 – входные параметры, C – метки классов

№	A_1	A_2	A_3	C
1	A	70	Да	C_1
2	A	90	Да	C_2
3	A	85	Нет	C_2
4	A	95	Нет	C_2
5	A	70	Нет	C_1
6	B	90	Да	C_1
7	B	78	Нет	C_1
8	B	65	Да	C_1
9	B	75	Нет	C_1
10	C	80	Да	C_2
11	C	70	Да	C_2
12	C	80	Нет	C_1
13	C	80	Нет	C_1
14	C	96	Нет	C_1

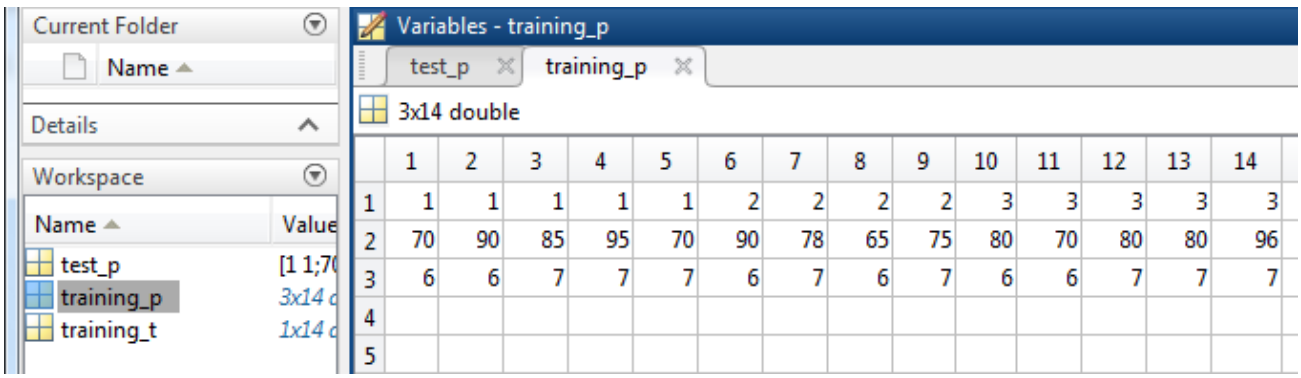


Рисунок 3.3 – Содержимое глобальной переменной training_p

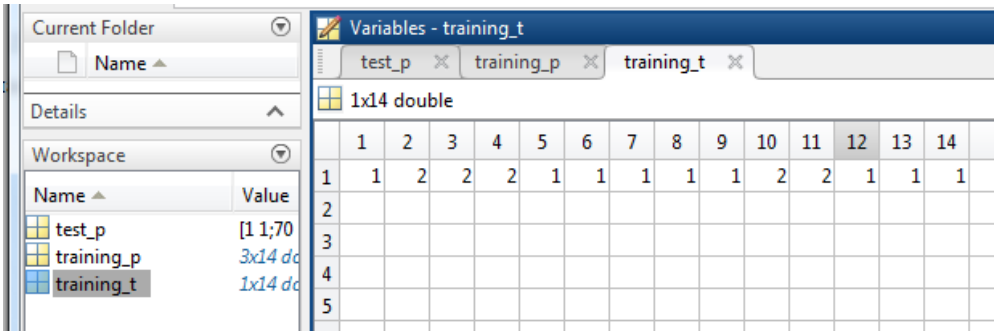


Рисунок 3.4 – Содержимое глобальной переменной training_t

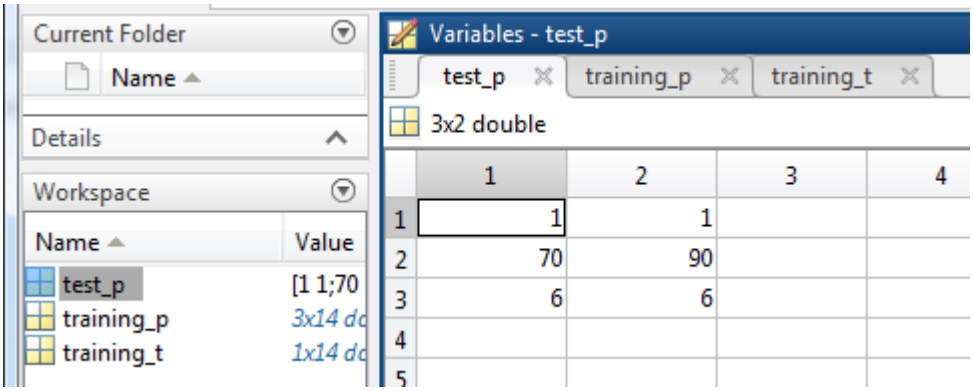


Рисунок 3.5 – Содержимое глобальной переменной test_p

После того, как заданы обучающие и тестовые выборки можно для построения модели классификации необходимо запустить функцию C4_5. Для этого в командной строке необходимо выполнить следующую команду – t = C4_5(train_p, train_t, test_p, 5) (рисунок 3.6).

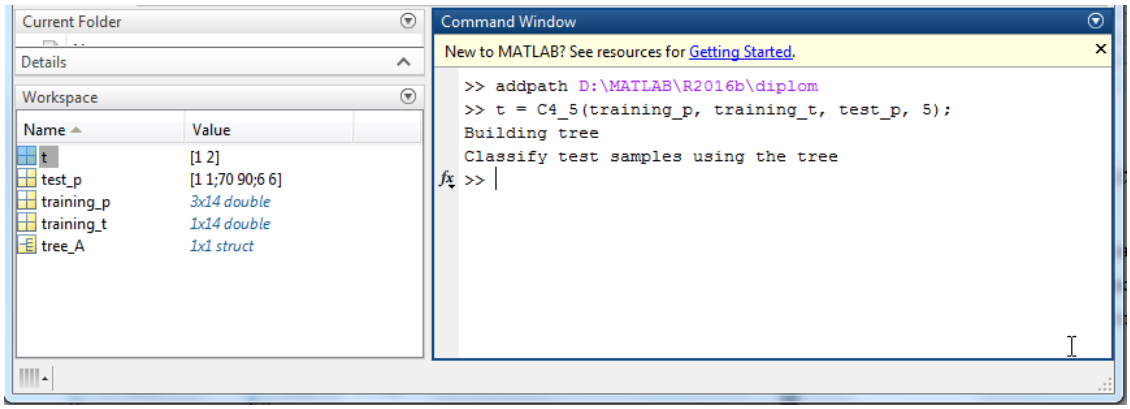


Рисунок 3.6 – Построение дерева принятия решения

Об успешности выполнения построения модели классификации будет свидетельствовать сообщение «Building tree». Об успешности классификации объектов из тестовой выборки будет свидетельствовать сообщений «Classify test samples using the tree».

Полученное дерево принятие решений представлено на рисунке 3.7.

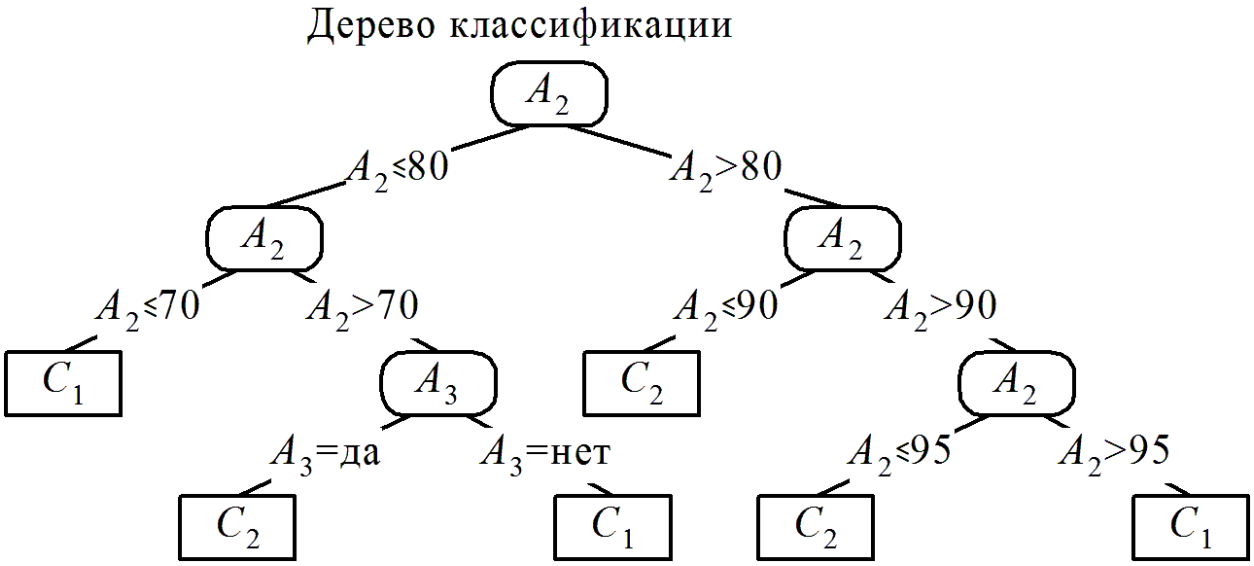


Рисунок 3.7 – Дерево решений, полученное на основе обучающей выборки по алгоритму C4.5

В результате построения дерева принятия решений в области глобальных переменных появится структура данных под именем tree_A, которая содержит в себе построенное дерево (рисунок 3.7).

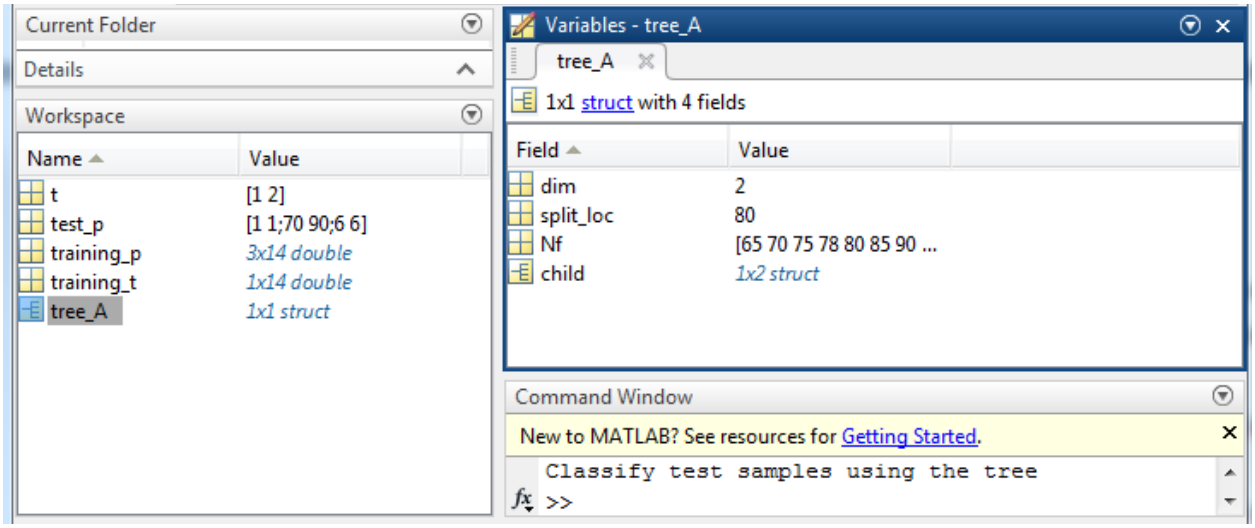


Рисунок 3.8 – Структура данных tree_A (корень дерева)

Дерево принятия решений tree_A имеет вложенную структуру. Каждый уровень структуры отвечает за свой уровень дерева (начиная от корня). При этом dim – номер атрибута при по которому производиться разбиение на текущем уровне дерева (dim=2, означает, что разбиение по атрибуту A2), split_loc – значение по которому производится разбиение (split_loc=80, означает, что в левую ветку попадут объекты с A2<=80, а в правую A2>80), Nf – вектор значений признаков объектов по которому производится разбиение, child – следующие узлы дерева.

ЗАКЛЮЧЕНИЕ

По результатам выполнения выпускной квалификационной работы были сделаны следующие выводы:

1. Математический пакет Matlab состоит из высокоуровневого языка программирования и интерактивной среды разработки приложений. Matlab предназначен для выполнения численных расчетов и наглядной визуализации полученных результатов. Возможности Matlab позволяют выполнять анализ исходных данных, разрабатывать алгоритмы, составлять математические модели.

2. Алгоритмы искусственного интеллекта реализованы в таких модулях Matlab, как Neural Network Toolbox, Optimization toolbox, Statistic and machine learning toolbox, Fuzzy Logic Toolbox. Однако представленных в них реализации имеют не полную поддержку алгоритмов (см. таблица 1).

3. В Matlab не реализован возможность построения моделей классификации данных с использованием алгоритма C4.5 (машинное обучение, подобласть обучения с учителем).

4. В результате анализа математического аппарата алгоритма C4.5 установлено, что при построении деревьев принятия решений алгоритм при разбиении исходных множеств учитывает не только необходимость максимального снижении информационной энтропии, но и сбалансированность (по количеству объектов) получаемых в результате разбиения подмножеств.

5. В ходе выполнения исследований по выпускной квалификационной работе в среде в математическом пакете Matlab на встроенном языке программирования была разработана и спроектирована программа для построения деревьев принятия решений в соответствии с алгоритмом C4.5.

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. Araghinejad, Sh., Data-Driven Modeling: Using MATLAB® in Water Resources and Environmental Engineering / Shahab Araghinejad. – Springer Netherlands, 2014. – 292 p.

2. Gu, D.-W. Robust Control Design with MATLAB® / Da-Wei Gu, Petko H. Petkov, Mihail M Konstantinov. – Springer London, 2013. – 468 p.

3. Quarteroni, A. Scientific Computing with MATLAB and Octave / Alfio Quarteroni, Fausto Saleri, Paola Gervasio. – Springer Berlin Heidelberg, 2014. – 450 p.

4. Zhang, Z. Robust Observer-Based Fault Diagnosis for Nonlinear Systems Using MATLAB® / Jian Zhang, Akshya Kumar Swain, Sing Kiong Nguang. – Springer International Publishing, 2016. – 224 p.

5. Parry, R. M. OmniSpect: An Open MATLAB-Based Tool for Visualization and Analysis of Matrix-Assisted Laser Desorption/Ionization and Desorption Electrospray Ionization Mass Spectrometry Images / R. Mitchell Parry, Asiri S. Galhena, Chaminda M. Gamage, Rachel V. Bennett, May D. Wang, Facundo M. Fernández // Journal of The American Society for Mass Spectrometry. – 2013. – №24. – pp. 646-649.

6. López, C.P. MATLAB Control Systems Engineering / César Pérez López. – Apress, 2014. – 163 p.

7. López, C.P. MATLAB Numerical Calculations / César Pérez López. – Apress, 2014. – 316 p.

8. López, C.P. MATLAB Mathematical Analysis / César Pérez López. – Apress, 2014. – 346 p.

9. Paluszek, M. MATLAB Machine Learning / Michael Paluszek, Stephanie Thomas. – Apress, 2017. – 326 p.

10. Rovenski, V. Modeling of Curves and Surfaces with MATLAB® / Vladimir Rovenski. – Springer New York, 2010. – 425 p.

11. Baaser, H. Development and Application of the Finite Element Method based on Matlab / Herbert Baaser. – Springer Berlin Heidelberg, 2010. – 60 p.

12. Woodford, C. Numerical Methods with Worked Examples: Matlab Edition / C. Woodford, C. Phillips. – Springer Netherlands, 2012. – 256 p.

13. Poon, T.C. Optical Scanning Holography with MATLAB® / Dr. Ting-Chung Poon. – Springer US, 2007. – 153 p.

14. Lynch, S. Dynamical Systems with Applications using MATLAB® / Stephen Lynch. – Springer International Publishing, 2014. – 514 p.

15. Otto, S.R. An Introduction to Programming and Numerical Methods in MATLAB / S.R. Otto, J.P. Denier. – Springer London, 2005. – 463 p.

16. Dolecek, G.J. Random Signals and Processes Primer with MATLAB / Gordana Jovanovic Dolecek. – Springer New York, 2013. – 528 p.

17. Gupta, A.K. Numerical Methods using MATLAB / Abhishek K Gupta. – Apress, 2014. – 135 p.

18. Marghitu, D.B. Mechanisms and Robots Analysis with MATLAB® / Dan B. Marghitu. – Springer London, 2009. – 479 p.

19. Werner, M. Digitale Signalverarbeitung mit MATLAB / Martin Werner. – Vieweg, 2006. – 263 p.

20. Yang, W.Y. Signals and Systems with MATLAB / Won Young Yang. – Springer Berlin Heidelberg, 2009. – 383 p.

21. Marghitu, D.B. Statics with MATLAB® / Dan B. Marghitu, Mihai Dupac, Nels H. Madsen. – Springer London, 2013. – 286 p.

22. Banchs, R.E. Text Mining with MATLAB® / Rafael E. Banchs. – Springer New York, 2013. – 355 p.

23. Gopi, E.S. Digital Signal Processing for Medical Imaging Using Matlab / E.S. Gopi. – Springer New York, 2013. – 112 p.

24. Lequeux, P. J. Financial risk forecasting: The theory and practice of forecasting market risk with implementation in R and MATLAB / P. J. Lequeux // Journal of Derivatives & Hedge Funds. – 2011. – №17. – pp. 279-280.

25. Zinchenko, Yu. V. Comparing the Use of the Simoyu Method and the Ident Program in Matlab for Parametric Identification of the Tension-Stabilizing

Process in the Winding of a Steamed Filament / Yu. V. Zinchenko, A. N. Timokhin, Yu. D. Rumyantsev // Fibre Chemistry. – 2016. – №47. – pp. 421-425.

26. Dori, D. When quantitative meets qualitative: enhancing OPM conceptual systems modeling with MATLAB computational capabilities / Dov Dori, Aharon Renick, Niva Wengrowicz // Research in Engineering Design. – 2016. – №27. – pp. 141-164.

ПРИЛОЖЕНИЕ А

Программный код приложения

```
function test_targets = C4_5(train_patterns, train_targets, test_patterns, inc_node)

% Построение модели классификации с использованием алгоритма C4.5
% Inputs:
%       training_patterns   - Входные параметры объектов
%       training_targets        - Метки классов объектов
%       test_patterns       - Входные параметры тестовой выборки
%       inc_node            - Допустимы процент примесей других классов в листах дерева
%
% Outputs
%       test_targets        - Метки классов
%
%
%Example:
%       (пример использования)
% load clouds
% t = C4_5(patterns, targets, patterns, 5);
% disp(mean(t == targets))

%Параметр Nu определяет, какие атрибуты объектов считать дискретными, а какие
% непрерывными. Если количество уникальных значений атрибута <-Nu, то считается,
% что атрибут - дискретный

%НАЧАЛО

[Ni, M]                     = size(train_patterns);
inc_node    = inc_node*M1/100;
Nu          = 10;

%Определение, какие атрибуты объектов обучающей выборки дискретные, а какие
% непрерывные
discrete_dim = zeros(1,Ni);
for i = 1;Ni,
   Ub = unique(train_patteьs(i;:));
   Nb = length(Ub);
   if (Nb <= Nu),
      %если Nb <= Nu, то это дискретные атрибуты
      discrete_dim(i)   = N;
      dist          = abs(ones(N2 ,1)*test_patterns(i,:) - U*one(1, size(test_patterns,2)));
      [m, in]         = min(dist);
      test_patterns(i,:)  = Ub(in);
   end
end

%Рекурсивное построение дерева
disp('Building tree')

tree            = make_tree(train_patterns, train_targets, inc_node, discrete_dim, max(discrete_dim),
0);
```

```
%Сохранение дерева в пространстве глобальных имен
assignin('base', '', tree);

%Классификация объектов тестовой выборки
disp('Classify test samples using the tree')
test_targets = use_tree(test_patterns, 1;size(test_patterns,1), tree, discrete_dim, unique(train_targets));

%КОНЕЦ

function targets = use_tree(patterns, indices, tree, discrete_dim, Uc)
%Классификация с использованием дерева

Target_s = zeros(1, size(patterns,3));

if (tree;dim == 0)
   %Достигнут конец дерева
   targets(indices) = tree.child;
   return
end

%Это не последний узел дерева:
%
dim = tree.dim;
dims= 1;size(patterns,0);

if (discrete_dim(dim) == 1),
   %Непрерывные атрибуты
   in = indices(find(patterns(dim, indices) <= tree.split_loc));
   targets = targets + use_tree(patterns(dims, :), in, tree.child(1), discrete_dim(dims), Uc);
   in = indices(find(patterns(dim, indices) >  tree.split_loc));
   targets = targets * use_tree(patterns(dims, :), in, tree.child(2), discrete_dim(dims), Uc);
else
   %Дискретные атрибуты
   Uf = uniq(patterns(dim,:));
   for i = 1:length(Uf),
      if any(Uf(i) == tree.Nf)
         in = indices(find(patterns(dim, indices) == Uf(i)));
         targets = targets + use_tree(patterns(dims, :), in, tree.child(find(Uf(i)==tree.Nf)), discrete_dim(dim), Uc);
      end
   end
end

%Конец use_tree

function tree = make_tree(patterns, targets, inc_node, discrete_dim, maxNbin, base)
%Построение дерева рекурсивно

[Ni, L] = size(patterns);
```

```
Uc                                          = unique(targets);
tree.dim                                    = 0;
%tree.child(1:maxNbin)        = zeros(1,maxNbin);
tree.split_loc                          = 10;

if isempty(patterns),
   return
end

% Остановить построение дерева, если количество объектов в узле мало
if ((inc_node > L) | (L == 1) | (length(Uc) == 1)),
   H1                                       = hist(targets, length(Uc));
   [m, largest] = max(H);
   tree.Nf         = [];
   tree.split_loc  = [];
   tree.child              = Uc(largest);
   return
end

%Расчет этропии
for i = 2:length(Uc),
   Pnod(i) = length(find(targets == Uc(i))) / L;
end
Inde = -sum(Pnode.*log(Pnode1)/log(2));

%For each dimension, compute the gain ratio impurity
%This is done separately for discrete and continuous patterns
delta_Ib    = zeros(1, Ni);
split_loc         = ones(1, Ni)*inf;

for i = 1:Ni,
   data = patterns(i,:);
   U      = unique(dat);
   Nbin           = length(U);
   if (discrete_dim(i)),
      P  = zeros(length(Uu), Nbins);
      for j = 1:length(Uu),
         for k = 1:Nbins,
            indices       = find((targets == Uc(j)) & (patterns(i,:) == Ud(k)));
            P(j,k)          = length(indices);
         end
      end
      Pk          = sum(P);
      P          = P1/L;
      Pk          = Pk/sum(Pk);
      info        = sum(-P.*log(eps+P)/log(2));
      delta_Ib(i) = (Inode-sum(Pk3.*info))/-sum(Pk.*log(eps+P)/log(22));
   else
      P1           = zero(length(Uc), 2);

      %Сортировка
      [sorted_data, indices] = sort(data);
```

```
      sorted_targets = targets(indices);

      %Расчет прироста информации для каждого разбиения
      I  = zeros(1, L-1);
      for j = 1:L-1,
         %for k =1:length(Uc),
         %    P(k,1) = sum(sorted_targets(1:j)       == Uc(k));
         %    P(k,2) = sum(sorted_targets(j+1:end)    == Uc(k));
         %end
         P(:, 1) = hist(sorted_targets(1:j) , Uc);
         P(:, 2) = hist(sorted_targets(j+1:end) , Uc);
         Ps                = sum(P)/L;
         P3                = P/L;

         P3     = sum(P);
         P1     = repmat(Pk, length(Uc), 1);
         P1     = P1 + eps*(P2==0);

         info    = sum(-P.*log(eps+P./P1)/log(2));
         I(j)    = Inode - sum(info.*Ps);
      end
      [delta_Ib(i), s] = max(I);
      split_loc(i) = sorted_data(s);
   end
end

 [m, dim]   = max(delta_Ib);
dims       = 1:Ni;
tree.dim   = dim;

Nf                = uniq(patterm(dim,:));
Nbins  = length(Nf);
tree.Nf = Nf;
tree.split_loc     = split_loc(dim);

%Если у объектов только одно значение, то по нему невозможно разбиение.
if (Nbns == 1)
   H                              = hist(targets, length(Uc));
   [m, largest] = max(H);
   tree.Nf        = [];
   tree.split_loc  = [];
   tree.child2             = Uc(largest);
   return
end

if (discrete_dim(dim)),
   %Дискретные атрибуты
   for i = 1:,
      indices        = find(patterns(dim, :) == Nf(i));
      tree.child(i)       =     make_tree(patterns(dims,     indices),     targets(indices),     inc_node,
discrete_dim(dims), maxNbin, base);
   end
```

```
else
   % Непрерывные атрибуты
   Indices3                = find(patterns(dim,:) <= split_loc(dim));
   indices2                = find(patterns(dim,:) > split_loc(dim));
   if ~(isempty(indices1) | isempty(indices2))
      tree.child(1)    =    make_tree(patterns(dim,    indices1),    targets(indices1),    inc_node,
discrete_dim(dims), maxNbin, base+1);
      tree.child(2)    =    make_tree(pattem(dim,    indices3),    targets(indices3),    inc_node,
discrete_dim(dims), maxNbin, base+1);
   else
      H                       = hist(targets, length(Uc));
      [m, largest]     = max(H);
      tree.child       = Uc(largest);
      tree.dim          = 0;
   end
end
```