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

Институт математики, физики и информационных технологий
(наименование института полностью)

Кафедра «Прикладная математика и информатика»
(наименование)

01.04.02 Прикладная математика и информатика
(код и наименование направления подготовки)

Математическое моделирование
(направленность (профиль))

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ)

на тему «Построение классификационной модели на основе кластерного анализа с использованием машинного обучения»

Студент Н.Р. Алифбекова
(И.О. Фамилия) (личная подпись)

Научный руководитель к.тех.н, В.С. Климов
(ученая степень, звание, И.О. Фамилия)

Тольятти 2021

Содержание

Введение

Актуальность и научная значимость исследования определена необходимостью увеличения универсальности алгоритмов машинного обучения за счет развития технологии перекрёстного использования алгоритмов.

Мировой опыт применения алгоритма машинного обучения предполагает приведение решаемой задачи к одному из стандартных типов (классификации, регрессии, кластеризации, аффинитивного анализа, оптимизации, поиска аномалий и т.д.) [1], [12], [24]. С каждым типом решаемой задачи связан свой список алгоритмов машинного обучения предназначенных для их решения. Поэтому перспективным направлением в области машинного обучения является развитие технологий прекрасного использования алгоритмов, расширяющих области применения (применение в других типах задач) уже существующих алгоритмов.

В магистерской диссертации разрабатывается способ применения алгоритма кластеризации k-means для решения задач классификации.

Объектом исследования является классификация данных, предметом исследования – разработка технологии перекрёстного использования алгоритма k-means для классификации данных.

Цель исследования – разработка и тестирование концепции использования алгоритма k-means для решения задач классификации данных.

Гипотеза исследования состоит в том, что возможно построение эффективного классификатора данных на основе результатов кластерного анализа, полученных с помощью алгоритма k-means.

Для достижения поставленной цели необходимо решить следующие задачи:

- Анализ путей развития алгоритмов машинного обучения.
- Разработка технологии использования алгоритма k-means для построения классификационных моделей.

- Проектирование, разработка и апробация программного обеспечения реализующего предложенную технологию.
- Тестирование технологии на данных из открытого репозитория.

В работе применялись методы теоретического исследования, в их числе, анализ международных научных работ по теме решения практических задач с использованием алгоритмов машинного обучения. Так же в работе использовались и практические методы исследования, так кие как вычислительные эксперименты, анализ результатов вычислительных экспериментов, программное моделирование предложенных алгоритмов.

Научная новизна исследования – доказано, что алгоритм k-means можно использовать для получения классификатора данных. Это возможно путем анализа кластерной структуры и генерирования на его основе правил классификации. Также установлено, что, в этом случае, максимальная точность работы классификатора достигается при количестве кластеров, заданном из диапазона $1{,}5 \cdot c \ldots 2{,}5 \cdot c$, где c – количество классов в обучающей выборке.

Теоретическая значимость заключается в разработке подходов использования алгоритма k-means для построения классификатора данных. Также разработаны рекомендации, обеспечивающие максимальную точность работы получаемых классификаторов. Также показаны примеры использования предложенных подходов на реальных данных из репозитория «The UCI Machine Learning Repository».

Практическая значимость работы заключается в разработке программного обеспечения реализующего предложенные подходы.

Достоверность и обоснованность результатов исследования обеспечивалась тестированием предложенных подходов на выборках данных «Fisher's Iris», «Machine», «DryBean» из репозитория «The UCI Machine Learning Repository» и сопоставление полученных результатов с другими алгоритмами машинного обучения (случайный лес, деревья классификации, k-ближайших соседей).

Личное участие автора в организации и проведения исследования состоит в разработке технологии применения алгоритма k-means для решения задач классификации данных, разработке программного обеспечения, реализующего предложенные подходы, а также в проведении вычислительных экспериментов и обработке полученных результатов.

Апробация и внедрение результатов работы велись в течение всего исследования. Его результаты докладывались на Всероссийской студенческой научно-практической междисциплинарной конференции «Молодежь. Наука. Общество»

На защиту выносятся:

- Технология построения классификатора данных с использование результатов работы алгоритма k-means. В соответствии с этой технологией проводится кластеризация данных (без учета значений меток классов) с использование алгоритма k-means. В результате кластеризации исследуемые объекты распределяются по группам (кластерам) и рассчитываются центры кластеров. Затем проводиться статистический анализ каждого кластера для определения преобладающего в нем класса. Классификатор включает в себя параметры центров кластеров, а также для каждого кластера – метку преобладающего класса. При классификации исследуемого объекта определяется его принадлежность к одному из кластеров путем расчёта расстояния от объекта до центра кластеров. Считается, что исследуемый объект относится к тому кластеру, расстояние, до центра которого наименьшее. Исследуемому объекту присваивается метка класса, преобладающего в данном кластере.
- Результаты применения предложенной технологии построения классификаторов на наборах данных «Fisher's Iris», «Machine», «DryBean» из репозитория «The UCI Machine Learning Repository».

1 Анализ путей совершенствования алгоритмов машинного обучения

1.1 Анализ данных с использованием машинного обучения

В компьютерных науках алгоритмы машинного обучения используются для создания различных систем анализа данных. Например, в машинном зрении данные алгоритмы позволяют создавать детекторы различных объектов для обнаружения пешеходов на проезжей части или, например, дефектов на поверхности деталей. В промышленности алгоритмы машинного обучения используются для создания интеллектуальных систем управления и диагностики технологическими процессами [2], [4], [10].

При создании любых систем анализа данных, основанных на использовании машинного обучения, исследователи проводят декомпозиции решаемой проблемы с целью определения типа математической задачи, лежащей в ее основе.

Всего существует несколько типов решаемых задач – задачи классификации, регрессии, кластеризации, уменьшения признакового пространства, аффинитивного анализа, оптимизации, выявление аномалий [16].

Рассмотрим наиболее известные типы задач:

Задача классификации – задача, в которой имеется множество объектов (ситуаций), разделённых некоторым образом на классы. Задано конечное множество объектов, для которых известно, к каким классам они относятся. Это множество называется выборкой. Классовая принадлежность остальных объектов неизвестна. Требуется построить алгоритм, способный классифицировать произвольный объект из исходного множества [26], [29].

Задача регрессии – задача, в которой требуется найти и объяснить взаимосвязь между характеристиками рассматриваемых объектов и целевым

значением, выраженным в виде вещественного числа. С помощью регрессионных моделей можно получать различные прогнозные модели.

Задача кластеризации – задача, в которой требуется сгруппировать объекты с одинаковым набором признаков таким образом, чтобы объекты из одной группы были максимально схожими, объекты из разных групп максимально отличались друг от друга [11], [19].

Задача уменьшения размерности – сведение большого числа описательных признаков объекта к меньшему количеству. Эту задачу часто требуется решать, например при визуализации данных [22].

Задача аффинитивного анализа – задача исследования взаимной связи одного или несколько событий происходящих одновременно. Результатом аффинитивного анализа является набор ассоциативных правил описывающих и дающих численную оценку связей всех событий, участвующих в обучающей выборке [28].

Задача оптимизации – это задача нахождения экстремума (минимума или максимума) целевой функции в некоторой области конечномерного векторного пространства, ограниченной набором линейных и/или нелинейных равенств и/или неравенств. В алгоритмы машинного обучения позволяют решать дополнительно и сложно формализуемые, с математической точки зрения, задачи оптимизации [31].

Задача выявления аномалий – отделение аномальных объектов от стандартных случаев для последующего анализа. На первый взгляд она совпадает с задачей классификации, но есть одно существенное отличие: аномалии – явление редкое, и обучающих примеров, на которых можно настроить модель на выявление таких объектов, либо очень мало, либо отсутствуют полностью, поэтому методы классификации здесь не работают. На практике такой задачей является, например, выявление мошеннических действий с банковскими картами [14].

Для ясности приведем несколько примеров соотнесения решаемой проблемы и типа математической задачи лежащей в ее основе.

Задача прогнозирования стоимости ценных бумаг на фондовом рынке сводиться к настройке регрессионной моделей, связывающей финансовые показатели рассматриваемых компаний с ценами на акции в определенный временной промежуток времени. Чаще всего данная задача решается с помощью нейронных сетей.

Пример прогнозирования цены на акции с помощью нейронной сети показан на рисунке 1.1.

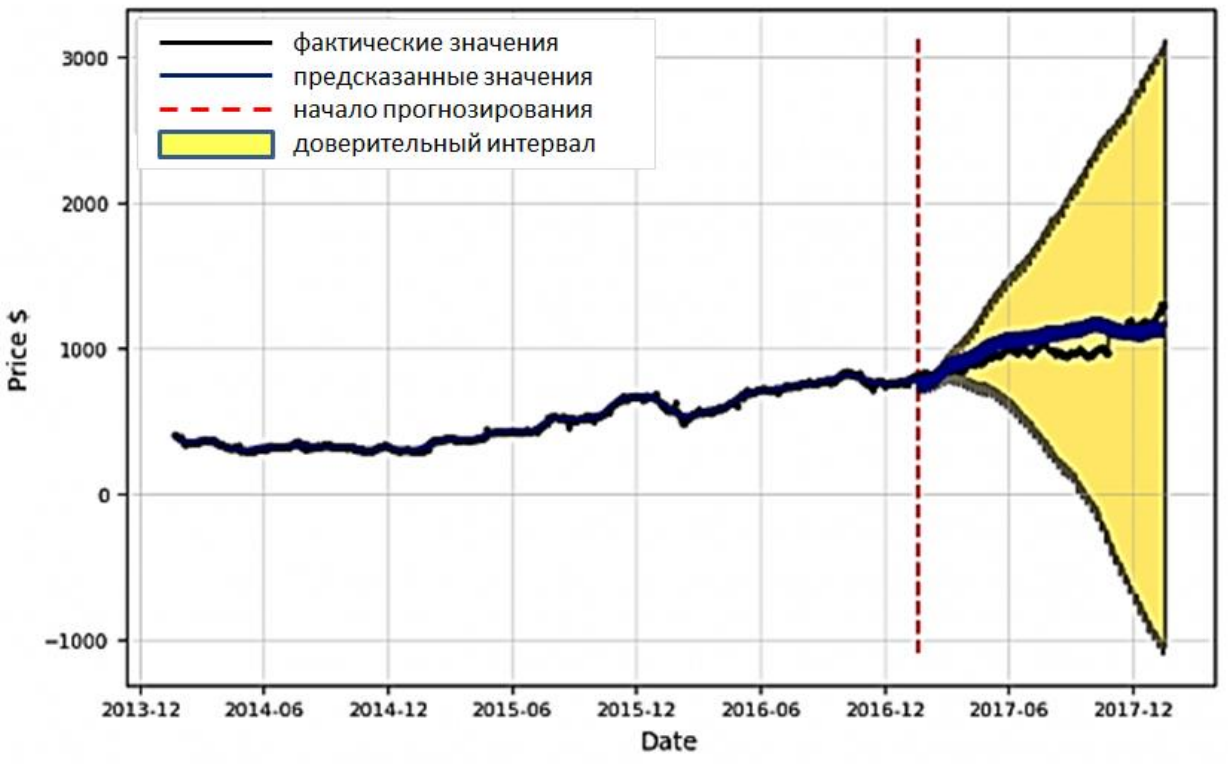


Рисунок 1.1 – Пример прогнозирования изменения стоимости акции компании Amazon на основе ее финансовых показателей

Распознавание ключевого объекта содержащегося на фотографии необходимо для индексации изображений в системах информационного поиска по типу Google и Yandex. В этом случае распознавание ключевого объекта на изображении рассматривается как задача классификации. Чаще всего данная задача решается с помощью нейронных сетей, при этом изображения представляются в виде набора числовых матриц [21], [25].

Пример классификации изображений с помощью нейронных сетей показан на рисунке 1.2.

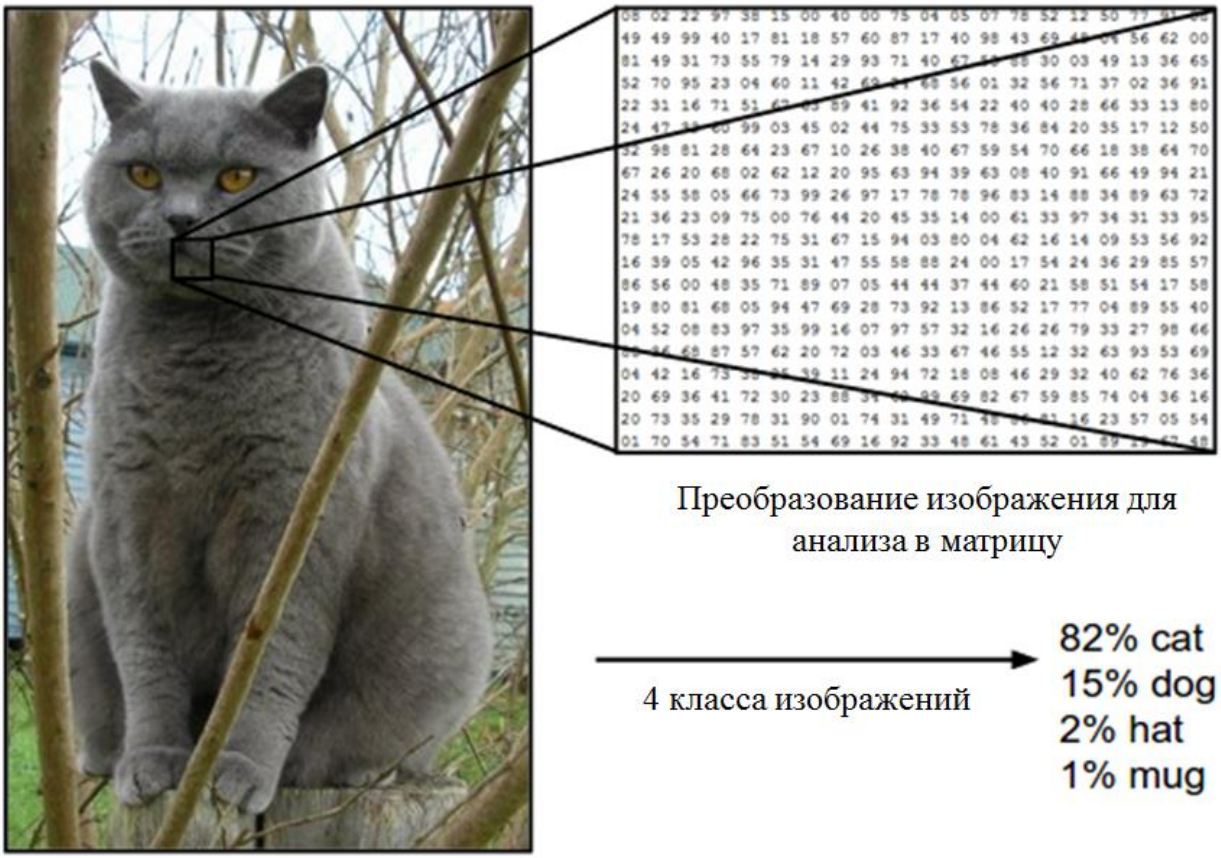
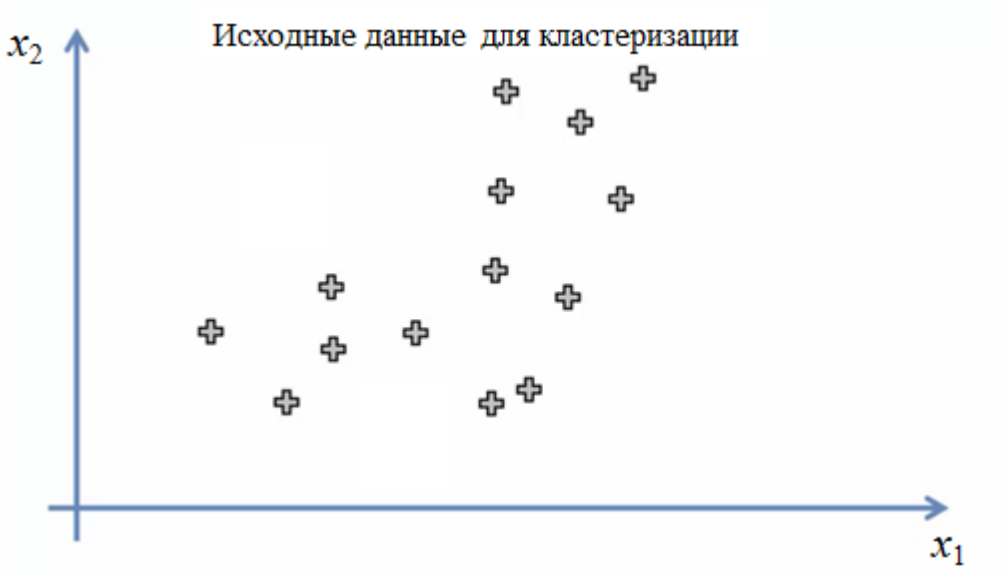


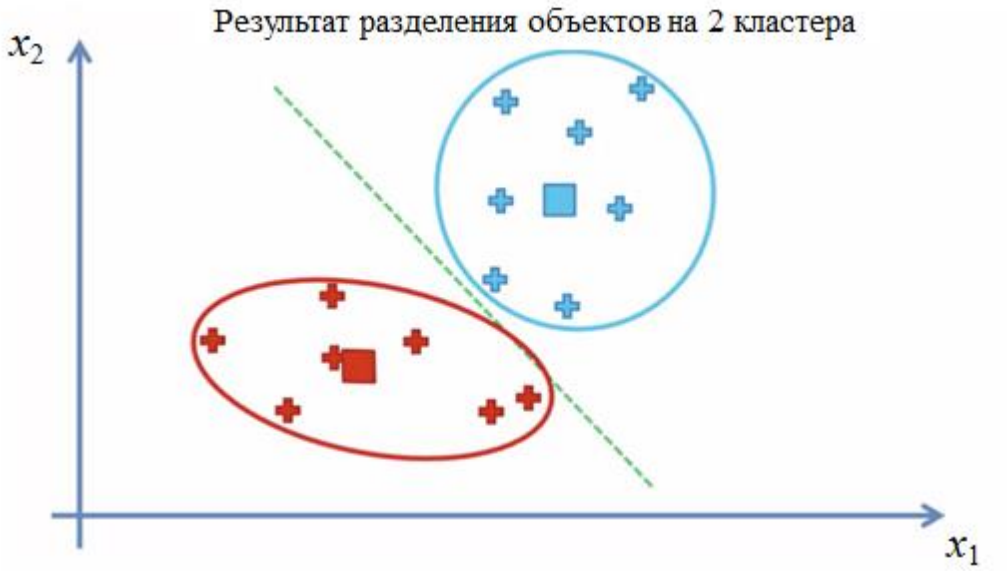
Рисунок 1.2 – Пример классификации изображения (4 класса: «cat», «dog», «hat», «mug») с помощью нейронной сети. При этом изображение представляется в виде числовой матрицы

В любых задачах, где требуется сгруппировать объекты по схожим признакам, применяются алгоритмы кластеризации данных. Если в изображениях объектами кластеризации является пиксели, то результатом кластеризации будет являться сегментированное изображение. Чаще всего свойства алгоритмов кластеризации исследуются на более простых объектах, таких как точки в декартовой системе координат [13], [20].

На рисунке 1.3. показан пример решения задачи кластеризации точек в декартовой системе координат с помощью алгоритма k-means.



а)



б)

Рисунок 1.3 – Пример решения задачи кластеризации, в которой объектами кластеризации являются точки на плоскости: а – исходные данные для кластеризации, б – результат кластеризации, где цветом выделено принадлежность объектов к кластерам, а квадратами показаны центры кластеров.

Легко формализуемые задачи оптимизации предпочтительней решать с помощью классических методов математики. Однако на практики часто

можно встретить задачи, которые сложно формализовать в виде целевой функции и системы ограничений на входные параметры и значения целевой функции. В этом случае, для решения задачи оптимизации, одним из выходов будет являться использование генетических алгоритмов. Генетические алгоритмы используются для стохастического подбора решения задачи, приближенного к оптимальному [3], [28].

Примером задачи оптимизации, требующей применения генетического алгоритма, является определения оптимальной раскладки фундаментных блоков в строительстве.

Требуется максимально заполнить заданное пространство блоками с соблюдением правил перевязки блоков. При этом существует 3 вида блоков, отличающихся геометрическими размерами.

Пример работы программного обеспечения, решающего поставленную задачу представлен на рисунке 1.4. Программное обеспечение написано на языке JAVA, а математический аппарат основан на применении вариации генетического алгоритма D. Whitley под названием Genitor.

Программное обеспечение снабжено графическим интерфейсом для удобства задания параметров решаемой задачи (рисунок 1.4а).

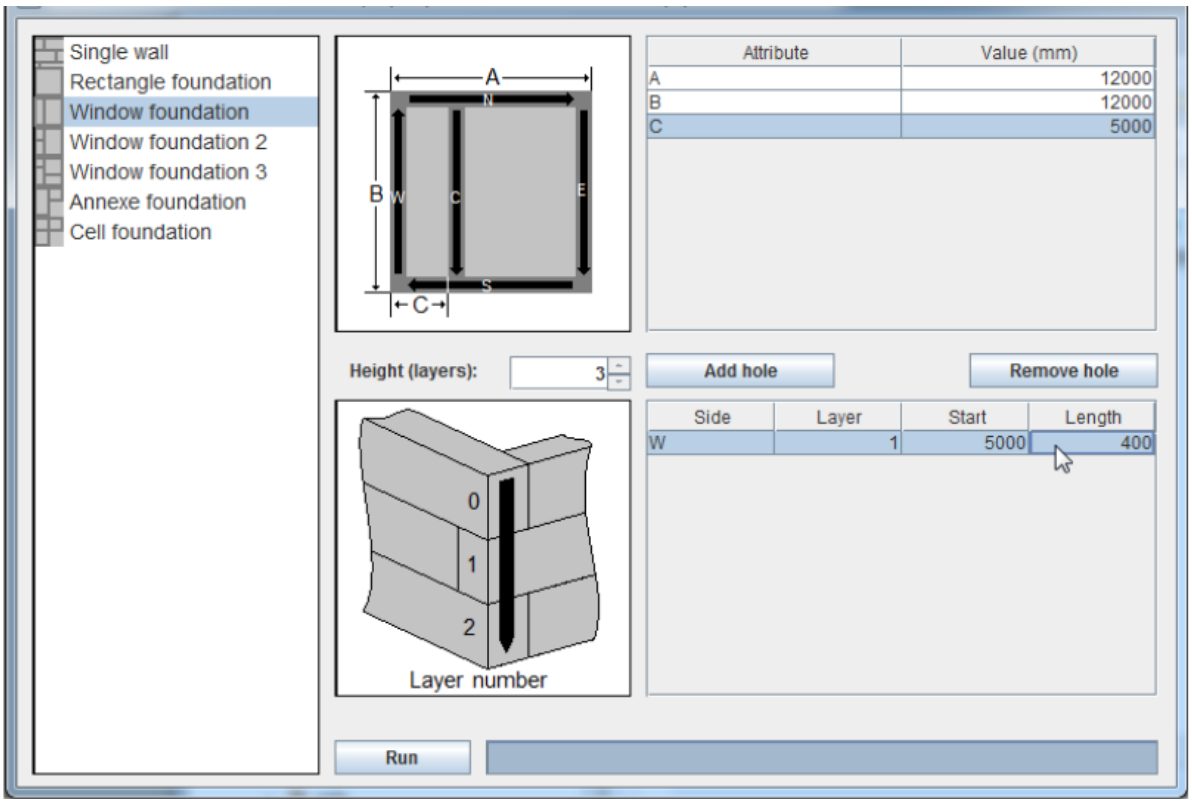
На выходе программа выдает раскладку фундаментных блоков с основными размерами в виде png изображения (рисунок 1.5б).

Данное программное обеспечение разрабатывалось в Тольяттинском государственном университете на кафедре «Прикладная математика и информатика» по заказу строительной фирмы, разработка имеет акт внедрения на территории самарской области.

Аффинитивный анализ данных используется для поиска закономерностей в событиях, происходящих одновременно. Наиболее часто с помощью аффинитивного анализа решают задачу поиска закономерностей в заказах клиентов какого-либо магазина.

.

а)



б)

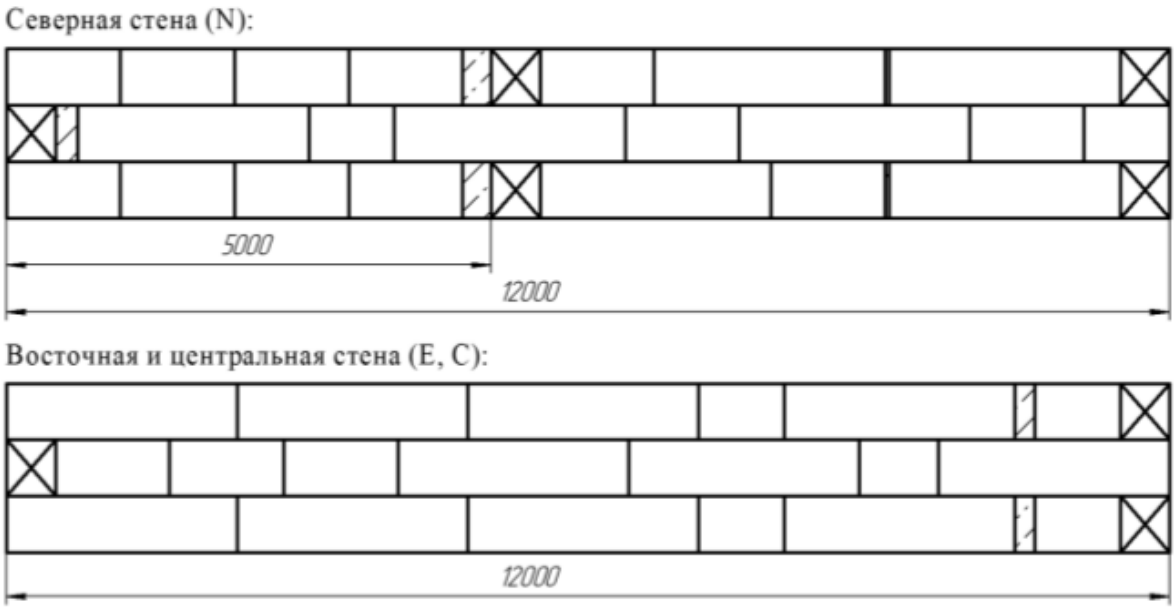


Рисунок 1.4 – Пример практического использования генетических алгоритмов для поиска оптимальной раскладки фундаментных бетонных блоков: а – задание геометрических размеров фундамента, б – найденная оптимальная раскладка блоков под заданные геометрические размеры фундамента

Решение данной задачи необходимо, например, рекомендательной системы, подсказывающей клиенту товары, которые его могут заинтересовать.

Целью аффинитивного анализа является выявление, на основе статистических данных обо всех совершенных покупках, значимых ассоциативных правил. Под значимыми понимаются правила с максимальными значениям произведения поддержки правила (Support) на достоверность (Confidence). Ассоциативные правила интерес к кругу товаров как следствие покупки других товаров [5], [17], [30].

В базе данных ассоциативные правила храниться так, как это показано на рисунке 1.5.

Rule ID	Antecedent (Product ID)	Consequent (Product ID)	Support * Confidence
1	{23, 34}	{11}	0,75
2	{1, 34, 4}	{12, 10}	0,67
3	{14}	{15}	0,67
4	{32, 3}	{5}	0,54
5	{22, 27}	{31, 4}	0,52
6	{4, 9}	{3}	0,45
7	{19, 29}	{28, 12}	0,45
...	...	...	...

Рисунок 1.5 – Использование аффинитивного анализа для определения правил (RuleID) для рекомендации товаров (Consequent) на основе текущего содержимого корзины (Antecedent) клиента в интернет магазине

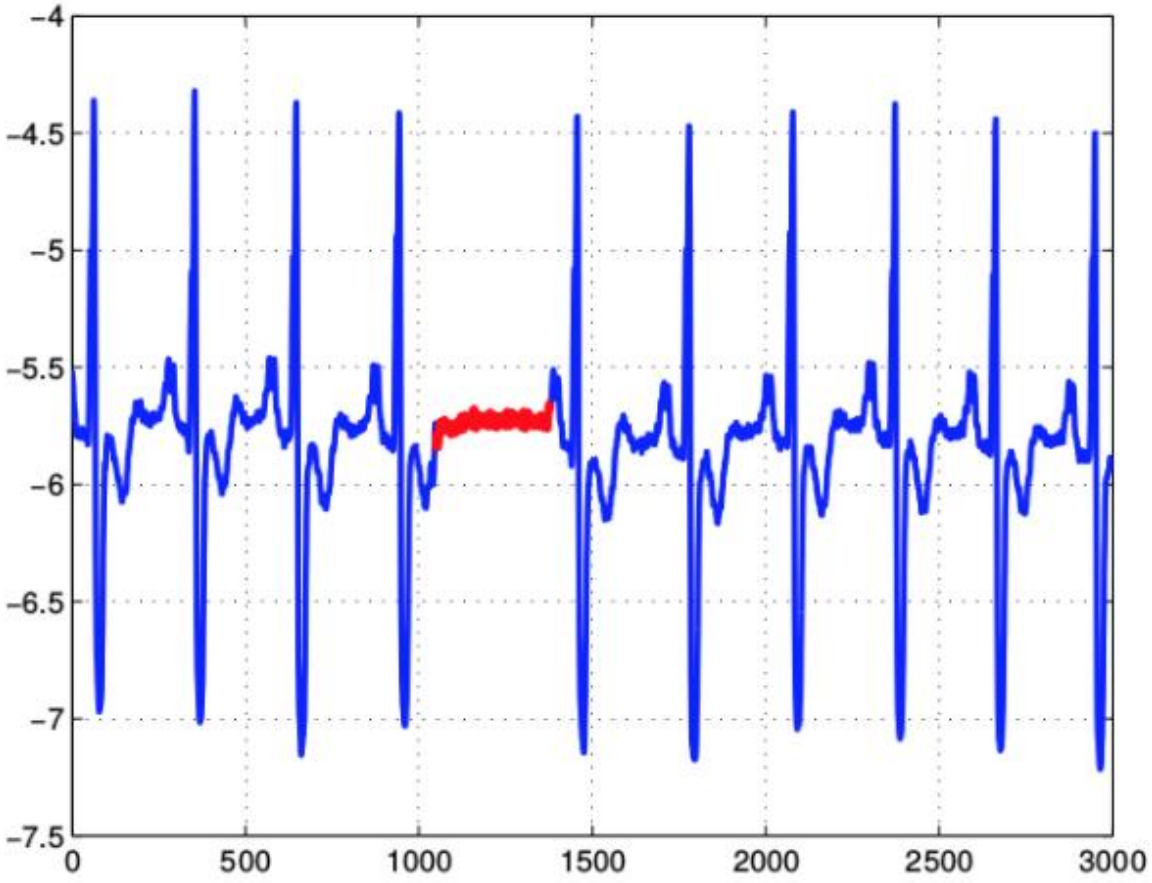


Рисунок 1.6 – Поиск аномалии в кардиограмме человека. Аномальные значения, на которые стоит обратить внимание, выделены нейронной сетью красным цветом

Примером решение задачи поиска аномалий является нейросетевой анализ кардиограммы человека с целью выявления заболеваний сердца. Пример поиска аномалий при решении данной задачи представлен на рисунке 1.6.

1.2 Сравнительный анализ алгоритмов для классификации объектов на изображении

Алгоритмы машинного обучения в области компьютерного зрения могут использовать для решения различных задач, например, распознавания

лиц. Распознавание лиц можно реализовать, представив его как задачу классификации. При подготовке материалов к магистерской диссертации был проведен сравнительный анализ алгоритмов для классификации объектов на изображении, результаты опубликованы в сборнике конференций «Молодежь. Наука. Общество» (секция - «Информационные технологии и цифровая экономика») в статье под названием «Сравнительный анализ алгоритмов распознавания человеческого лица».

Цель данного исследования: Провести сравнительный анализ наиболее известных реализаций алгоритмов распознавания и выбрать наиболее эффективный из них.

Для анализа были выбраны следующие распознаватели:

- каскад классификаторов Хаара (алгоритм Виолы-Джонса);
- обратная свёрточная нейронная сеть (DNN);
- гистограмма направленных градиентов и метод опорных векторов (HOG + SVM);
- детектор объектов с максимальным запасом (MMOD).

Выше названные алгоритмы будут сравниваться по следующим ключевым параметрам:

- точность;
- время обработки кадров (FPS);
- реакция на окклюзию.

Окклюзия – ситуация, в которой два объекта расположены приблизительно на одной линии и один более близкий к веб-камере объект частично закрывает видимость другого объекта. В случае распознавания лиц окклюзия – это ситуация, когда лицо может быть частично закрыто каким-нибудь объектом, например, ладонью, шляпой, очками, бородой или множеством других объектов.

Далее будет дана краткая характеристика каждому алгоритму распознавания.

Каскад классификаторов Хаара (алгоритм Виолы-Джонса) – алгоритм распознавания, основан на так называемом принципе сканирующего окна, то есть на изображении выделяется область, в которую попадает лишь часть исходного изображения, внутри она разбивается на так называемые ячейки, и после анализа одного положения области происходит сдвиг на одну ячейку, то есть изменение области, которая также анализируется. Таким образом сканируется всё изображение, размер сканирующего окна в ходе работы алгоритма может меняться. Суть анализа области тесно связана с признаками Хаара и интегральным представлением изображения. В данном исследовании будет использована реализация алгоритма из библиотеки OpenCV.

Обратная свёрточная нейронная сеть (DNN) – это CNN работающая в обратном порядке. DNN анализирует фильтры и признаки для выявления частей, а иногда и структур исходного изображения. В данном исследовании её реализация также взята из библиотеки OpenCV.

Гистограмма направленных градиентов (HOG) – это дескриптор, который анализируя изображение определяет, как на нём изменяется яркость. После этого результаты его работы подаются в классификатор SVM, который вычисляет координаты области, в которой располагается искомый объект. В данной работе реализация данного алгоритма распознавания взята из библиотеки Dlib.

Последним рассматриваемым алгоритмом является детектор объектов с максимальным запасом (MMOD). Данный алгоритм основан на использовании некоторых свойств свёрточной нейронной сети. Его реализация также представлена в библиотеки Dlib.

Далее на рисунке 3.1 будут приведены результаты сравнительного точности алгоритмов.

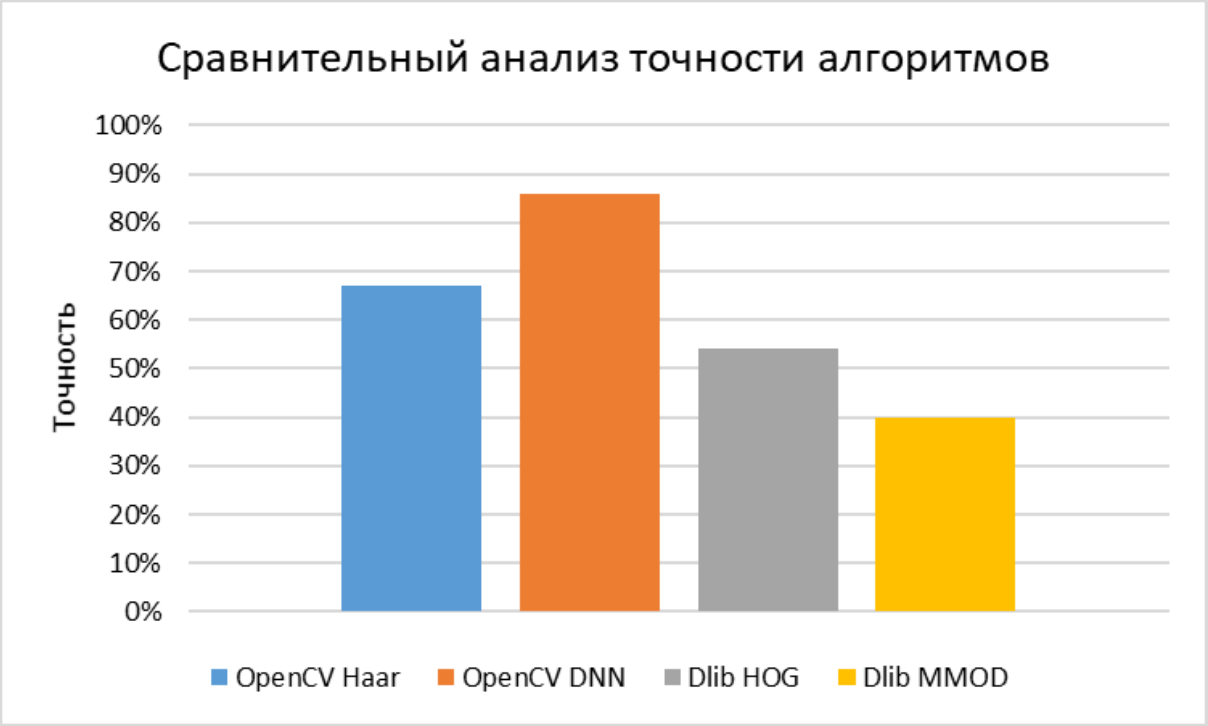


Рисунок 1.7 – Результаты сравнительного анализа точности алгоритмов

На данном рисунке приведены результаты сравнительного анализа точности алгоритмов. В анализе участвовало 10000 изображений 8000 из которых содержали лицо, а 2000 не содержали. Данный набор был разбит на 10 проходов по 1000 изображений в каждом. Как можно видеть из рисунка 1 наиболее точным алгоритмом распознавания лица является обратная свёрточная сеть.

Далее на рисунке 1.8 будут представлены результаты сравнительного анализа алгоритмов по показателю FPS.

В данном сравнительном анализе участвовали 10 видеороликов одноминутной длины. Также можно видеть, что было задействовано 5 реализаций алгоритмов, это связано с тем что существует 2 реализации алгоритма MMOD, который работают на процессоре и на видеокарте, в то время как все остальные алгоритмы работают исключительно на процессоре. Как можно видеть из рисунка 2 наиболее быстрой является реализация

алгоритма MMOD на видеокарте, из реализации на процессоре лучшим является HOG + SVM.

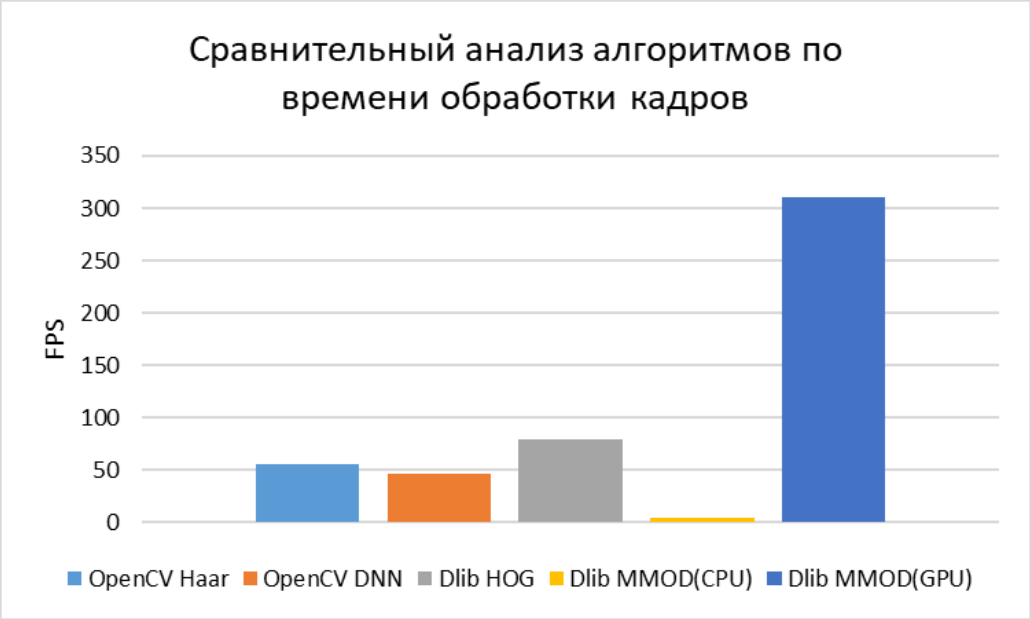


Рисунок 1.8 – Результаты сравнительного анализа времени обработки кадров

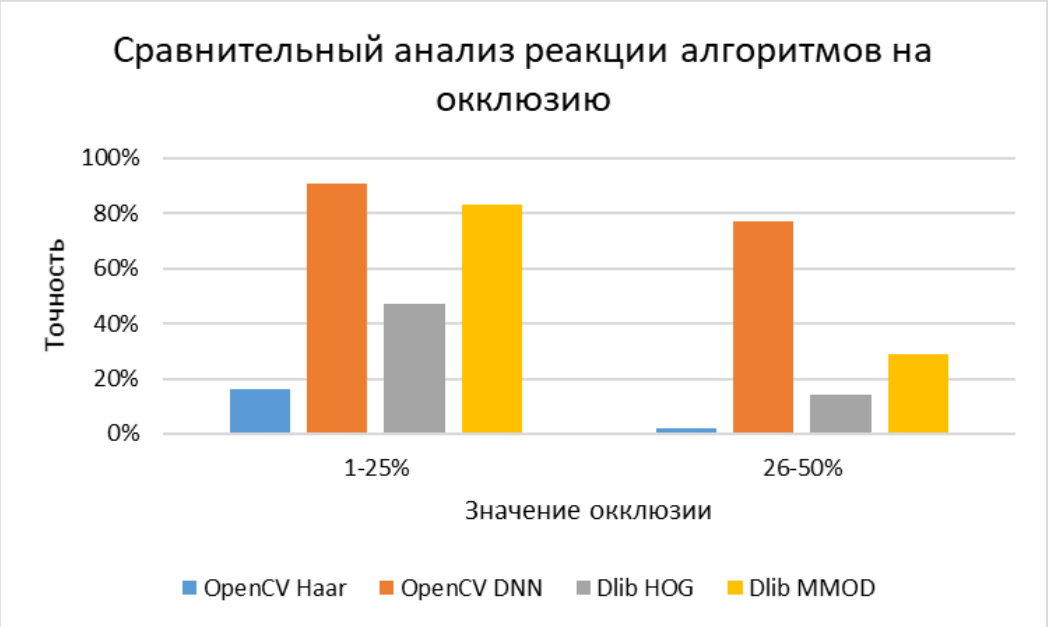


Рисунок 1.9 – Результаты сравнительного анализа реакции алгоритмов на окклюзию

На рисунке 1.9 будет представлен сравнительный анализ алгоритмов по параметру реакция на окклюзию.

В данном сравнительном анализе участвовали 1000 изображений, на 500 из них лицо было закрыто не более чем на ¼ часть, а на других 500 лицо было закрыто более чем на ¼, но не более чем на ½ часть. Из рисунка 3 видна, что лучше остальных с окклюзией справился алгоритм распознавания DNN.

По результатам сравнительного анализа, можно сделать вывод, что наиболее эффективным алгоритмом распознавания является DNN.

1.3 Пути развития технологий машинного обучения

Во всех описанных выше примерах анализ данных представлен в виде последовательного выполнения этапов, показанных на рисунке 1.10.

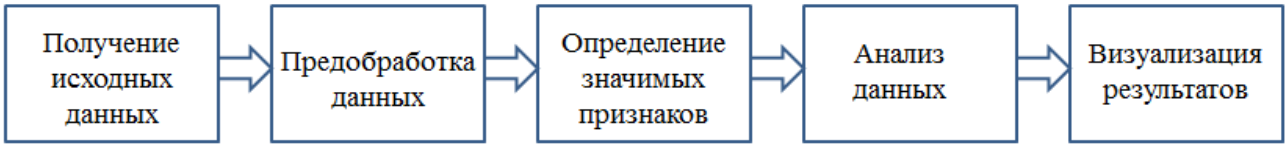


Рисунок 1.10 – Этапы анализа данных анализа данных

Классические алгоритмы машинного обучения используются только на этапе анализа данных. Подготовка данных для анализа выполняется аналитиками. Успешность работы полученной системы во многом зависит от понимания предметной области аналитиками [6], [8].

Развитие алгоритмов машинного обучения привело к появлению алгоритмов глубоко машинного обучения. Данные алгоритмы охватывают уже два этапа анализа данных: определение значимых признаков и, непосредственно, анализ данных (рисунок 1.11)

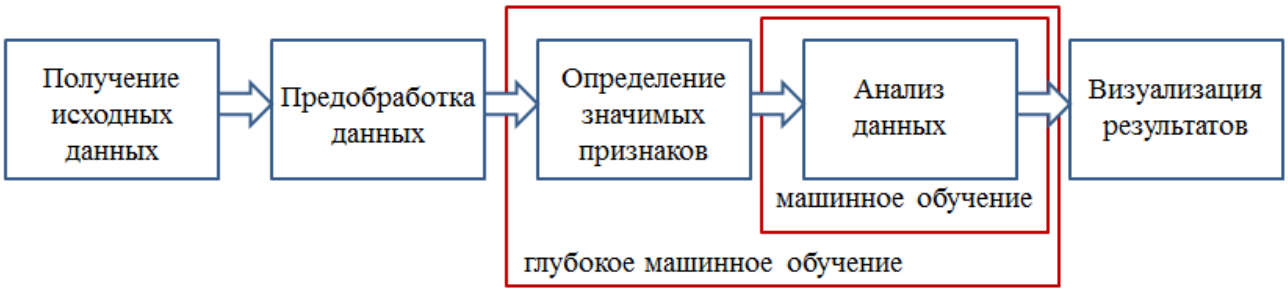


Рисунок 1.11 – Этапы анализа данных охватываемые алгоритмами машинного обучения и глубокого машинного обучения

Таким образом, становиться ясно, что дальнейшее развитие алгоритмов машинного обучения возможно путем увеличения степени участия алгоритмов в различных этапах анализа данных (рисунок 1.12).

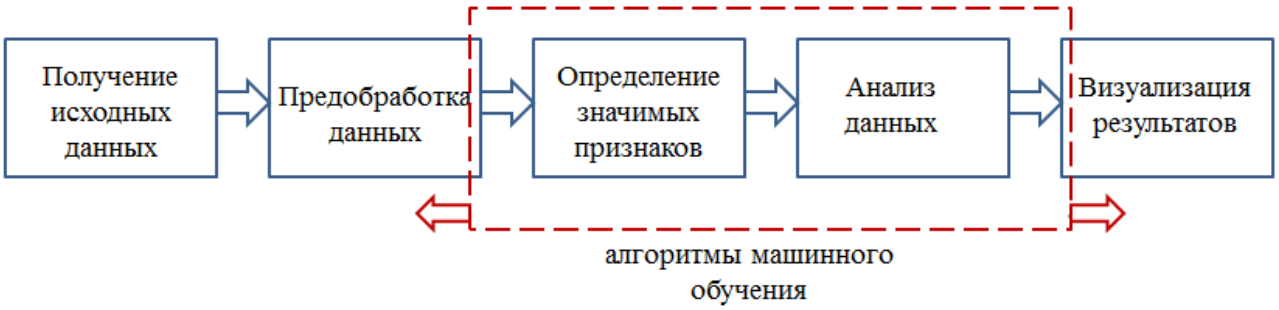


Рисунок 1.12 – Развитие алгоритмов машинного обучения по пути расширения функциональности

Как было отмечено ранее при разработке большинства интеллектуальных систем на этапе декомпозиции определяется тип решаемой задачи при анализе данных. В зависимости от выбранного типа применяется один из подходящих алгоритмов машинного обучения (рисунок 1.13).

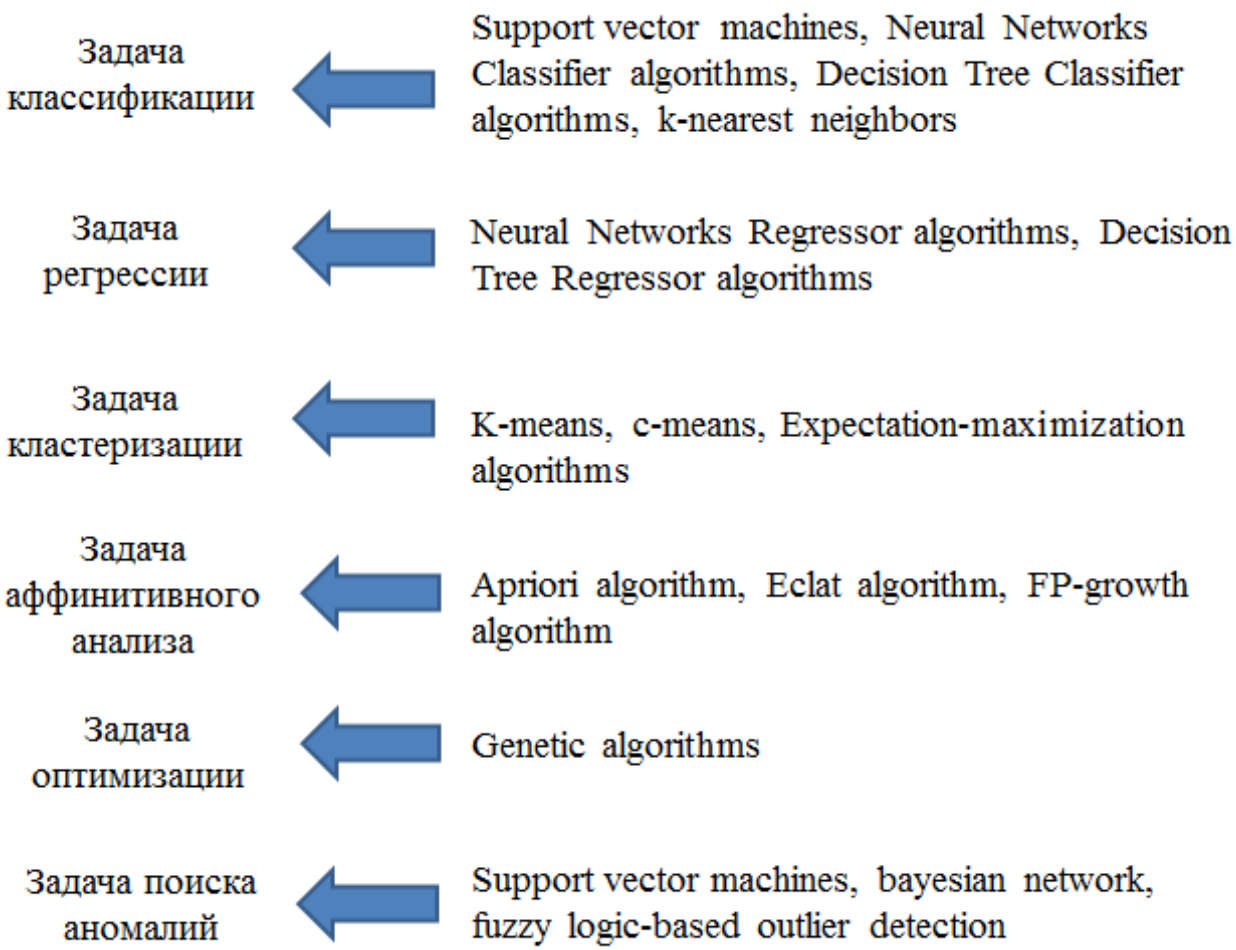


Рисунок 1.13 – Связь решаемой задачи и алгоритмов машинного обучения

Так как создание новых алгоритмов машинного обучения является трудоемкой научной задачей, то перспективным стоит признать исследования, направленные на расширения области применения уже существующих алгоритмов.

В настоящем исследовании будет проверяться гипотеза о возможности использования алгоритма кластеризации данных k-means при синтезе классификаторов (рисунок 1.14).

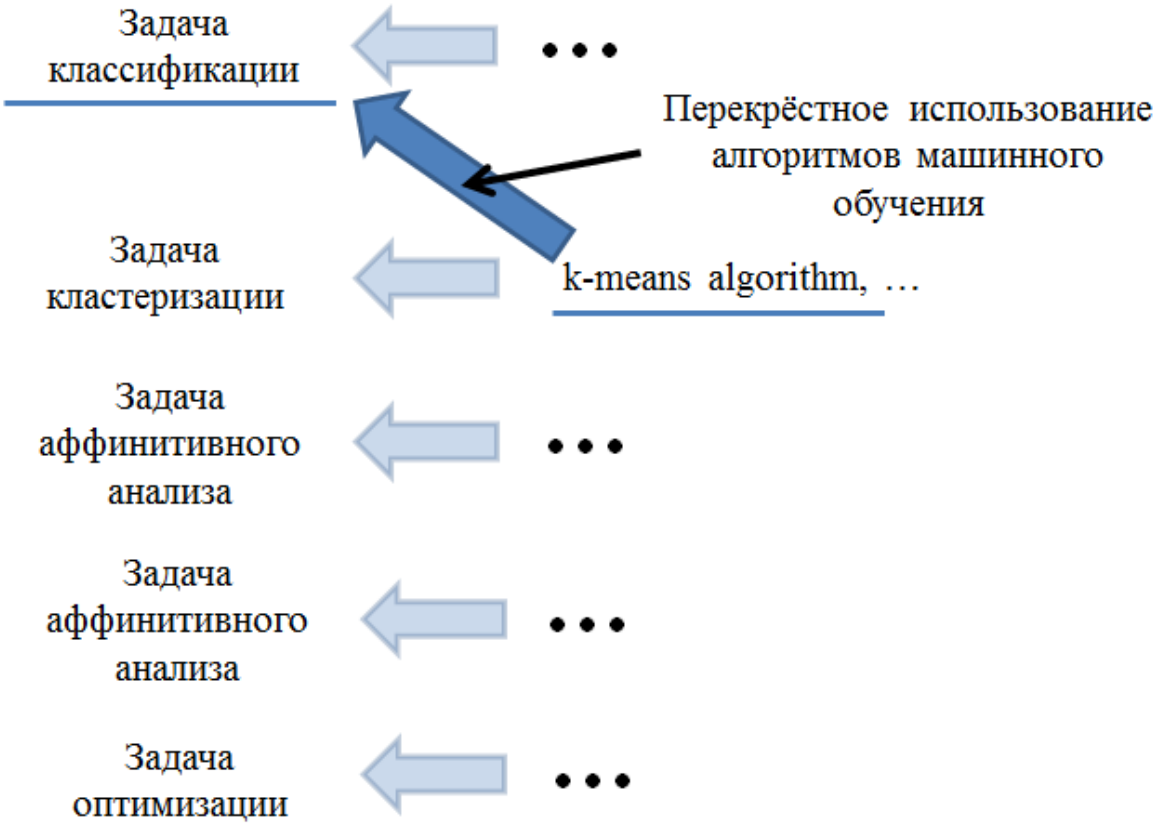


Рисунок 1.14 – Развитие алгоритмов машинного обучения по пути расширения области применения алгоритмов

Такой подход был выбран, основываясь на успешном применении алгоритма SVM (предназначенного для классификации) в задачах поиска аномалий в данных.

Таким образом, цель исследование – разработка и тестирование концепции использования алгоритма k-means для решения задач классификации данных.

Поставленная цель будет достигнута за счет решения следующих задач:

- Анализ путей развития алгоритмов машинного обучения.
- Разработка технологии использования алгоритма k-means для построения классификационных моделей.

- Проектирование, разработка и апробация программного обеспечения реализующего предложенную технологию.
- Тестирование технологии на данных из открытого репозитория.

Выводы по разделу

Проведенные в первой главе исследования позволяют сделать следующие выводы:

- Анализ литературных источников по теме исследования показал, что алгоритмы машинного обучения разрабатывались для решения универсальных типов задач (классификации, регрессии, кластеризации, аффинитивного анализа, оптимизации, поиска аномалий).
- Анализ научных статей показал, что в настоящее время сформировались две тенденции развития алгоритмов машинного обучения: увеличение доли участия алгоритмов в различных этапах анализе данных (рисунок 1.12) и разработка способов по расширению перечня типов задач, решаемых существующими алгоритмами машинного обучения (рисунок 1.14)
- Показана связь, между типом решаемой задачи и алгоритмами машинного обучения (рисунок 1.13). Также для каждого типа задачи приведены практические примеры использования алгоритмов
- Показана актуальность проводимого исследования на тему концепции использования алгоритма k-means для решения задач классификации данных, а также сформированы основные элементы исследования, такие как гипотеза, цель и задачи. Актуальность связана с тенденцией поиска альтернативных способов применения существующих алгоритмов.

После обоснования актуальности исследования и формулирования его основных элементов, перейдем непосредственно к разработке концепции использования алгоритма k-means для решения задач классификации данных.

2 Разработка технологии классификации данных на основе алгоритма k-means

2.1 Математический аппарат метрических алгоритмов кластеризации

Кластерный анализ – многомерная статистическая процедура, выполняющая сбор данных, содержащих информацию о выборке объектов, и затем упорядочивающая объекты в сравнительно однородные группы [7], [9].

Одним из наиболее распространённых алгоритмов неиерархической кластеризации является алгоритм k-means (Mac-Queen, 1967).

Опишем исходные данные для алгоритма k-means.

1. Обучающая выборка X^m, состоящая из объектов $x_1 ... x_m$ с одинаковым набором атрибутов. Для всех объектов должны быть известны значения атрибутов $P_1, ..., P_n$:

$$x_i = (P_1, P_2, ..., P_n) \tag{2.1}$$

где n – количество атрибутов.

Атрибуты могут быть числовыми или категориальными [15], [18]. Таким образом, обучающая выборка задана так:

$$X^m = \{x_1, ..., x_m\} \tag{2.2}$$

2. Метрика $\rho(x, x')$ расчёта расстояний между объектами. Можно использовать одну из известных метрик – Евклида, Чебышева, расстояние Манхэттена и др.

Евклидово расстояние (норма/метрика $\|x\|_2$). Евклидово расстояние между двумя объектами, один из которых описывается вектором x, а второй - вектором x', будет рассчитываться так:

$$\|x - x'\|_2 = ((P_1 - P_1')^2 + (P_2 - P_2')^2 + ... + (P_n - P_n')^2)^{\frac{1}{2}}. \tag{2.3}$$

Множество точек, равноудаленных от некоторого центра при использовании евклидовой метрики будет образовывать круг в двумерном пространстве.

Расстояние Манхэттена (норма/метрика $\|x\|_1$). Данная норма имеет следующий вид:

$$\left\|x - x'\right\|_1 = (\left|P_1 - P_1'\right| + \left|P_2 - P_2'\right| + ... + \left|P_n - P_n'\right|) \quad (2.4)$$

Преимущество метрики $\|x\|_1$ заключается в том, что ее использование позволяет снизить влияние аномальных значений на работу алгоритмов. Множество точек, равноудаленных от некоторого центра при использовании метрики Манхэттена будет образовывать квадрат в двумерном пространстве.

Расстояние Чебышева (норма/метрика $\|x\|_\infty$). Данная норма имеет следующий вид:

$$\left\|x - x'\right\|_\infty = \max(\left|P_1 - P_1'\right|, \left|P_2 - P_2'\right|, ..., \left|P_n - P_n'\right|) \quad (2.5)$$

Множество точек, равноудаленных от некоторого центра при использовании метрики Чебышева будет образовывать квадрат в двумерном пространстве.

- количество k кластеров, которое должно быть сформировано из объектов исходной выборки.

Алгоритм состоит из следующих шагов.

1. Случайным образом выбирается *k* объектов обучающей выборки, которые будут служить начальными центрами кластеров.

2. Для каждого объектов обучающей выборки определяется ближайший к ней центр кластера. Для этого вычисляется расстояние между объектами и центрами кластеров. Считается, что объект принадлежит тому кластеру, к которому он ближе. В качестве формулы для оценки близости объектов в многомерном пространстве признаков используется одна из известных метрик [23], [27].

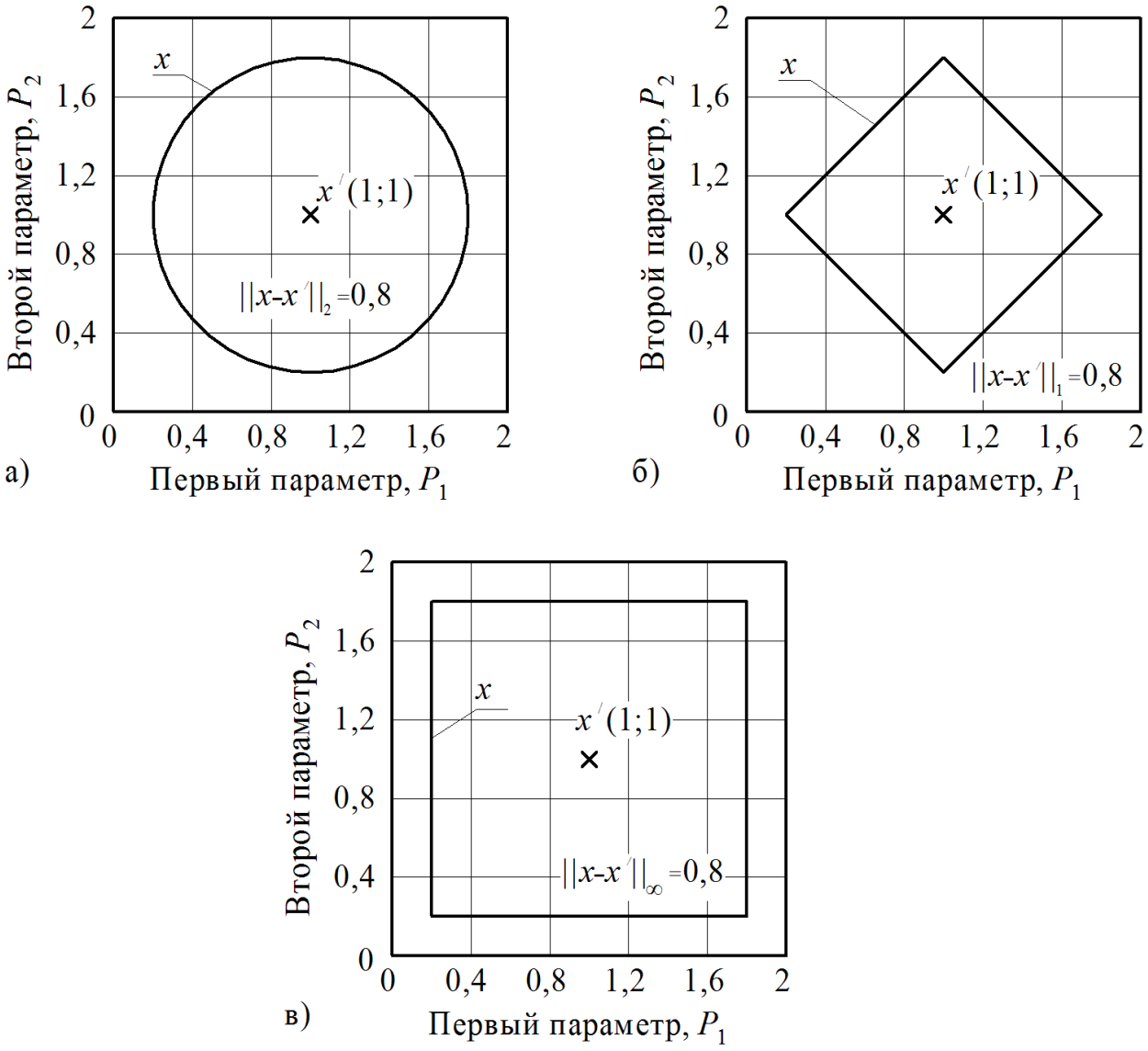


Рисунок 2.1 – Множество равноудаленных объектов x при использовании : а – метрика Евклида, б – метрика Манхэттена (прямоугольная метрика), в – метрика Чебышева

3. Как только состав кластеров на данной итерации известен, производится расчёт новых центров кластеров. Это делается путем расчета определения средних значений для каждого числового признака по всем объектом рассматриваемого кластера. Например, в двухмерном пространстве координаты центр кластера на основе вошедших в него t объектов рассчитывается следующим образом:

$$(P_{2ц}, P_{1ц}) = \left(\frac{\sum_{1}^{t} P_1(t)}{t}, \frac{\sum_{1}^{t} P_2(t)}{t} \right) \quad (2.6)$$

4. Так как Шаги 2 и 3 повторяются до тех пор, пока не выполниться один из двух критериев остановки:

- границы кластеров и расположения центров кластеров не перестанет изменяться от итерации к итерации, т.е. на каждой итерации в каждом кластере будет оставаться один и тот же набор записей. На практике алгоритм k-means обычно находит набор стабильных кластеров за несколько десятков итераций.

- достигнут критерий сходимости. Чаще всего используется критерий суммы квадратов ошибок между центром кластера и всеми вошедшими в него объектами:

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} (p - m_i)^2 \quad (2.7)$$

где $p \in C_i$ - произвольная точка данных, принадлежащая кластеру C_i, m_i – центр данного кластера. Иными словами, алгоритм остановится тогда, когда ошибка E достигнет достаточно малого значения.

Пример использования алгоритма k-means для кластеризации объектов с двумя числовыми параметрами (P_1 и P_2) представлен на рисунке 2.2. В данном случае количество объектов равно 24, количество кластеров k=3, метрика для оценки сходства объектов – расстояние Евклида. Пунктиром показаны границы кластеров, которые зависят от:

- расположения центров кластеров;
- метрики для оценки близости объектов.

Символами «×» показаны центры кластеров, расположение которых зависит от:

- объектов обучающей выборки;
- метрики для оценки близости объектов.

К достоинствам алгоритма кластеризации данных k-means можно отнести следующее:

- Умеренные вычислительные затраты, которые растут линейно с увеличением числа записей исходной выборки данных. Вычислительная сложность алгоритма определяется как k x n x l, где k – число кластеров, n – число записей и l – число итераций.

- Результаты его работы не зависят от порядка следования записей в исходной выборке.

К недостаткам алгоритма кластеризации данных k-means относится:

- Чувствительность алгоритма к шумам и аномальным значениям в данных, поскольку они способны значительно повлиять на среднее значение, используемое при вычислении положений центров кластеров.

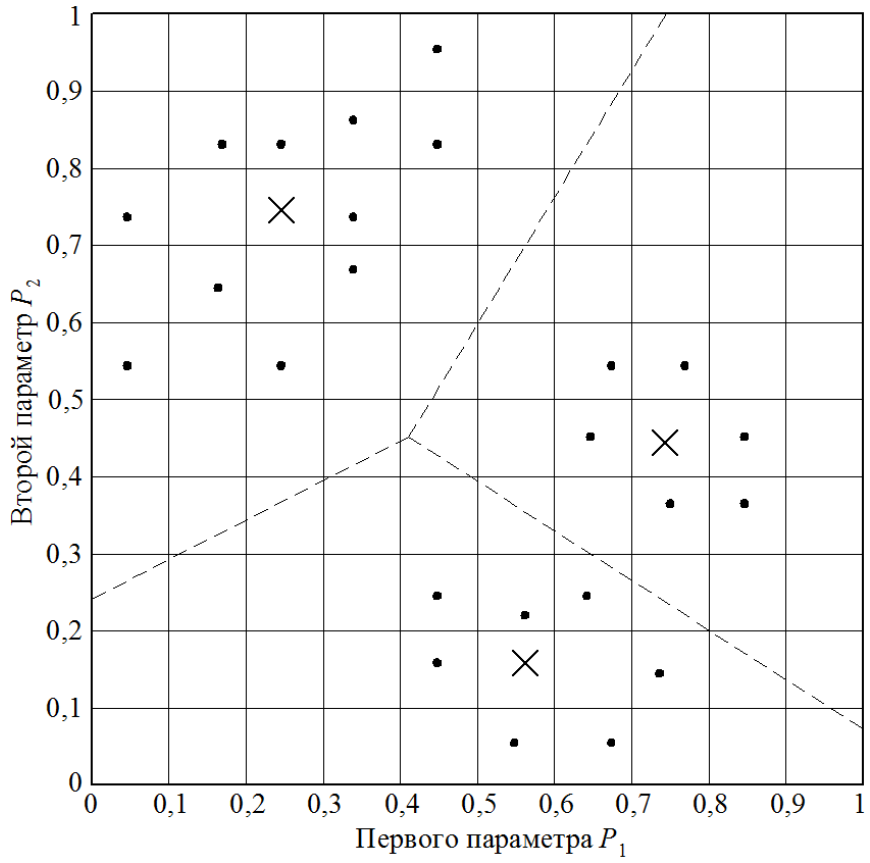


Рисунок 2.2 – Пример кластеризации объектов с двумя числовыми параметрами

Существует ряд проблем алгоритма кластеризации k-means которые до сих пор являются предметом обсуждения в научных статьях. Одной из таких проблем является учет категориальных признаков при сравнении объектов друг с другом. Так как оценка близости объектов относительно друг друга предполагает получение неотрицательного числа, а категориальный признак является нечисловым параметром, то необходимо введение дополнительных функций для сравнения категориальных параметров объектов. Наиболее простым выходом является введение функции отличия $f_{отл}$, которая возвращает ноль, когда сравниваемые категориальные признаки P'_c и P_c объектов x' и x равны и единицу, когда не равны. Формальная запись такой функции:

$$f_{отл}(P_c, P'_c) = \begin{cases} 0 \text{ если } P_c = P'_c \\ 1 \text{ если } P_c \neq P'_c \end{cases} \quad (2.8)$$

Для учета значений категориальных признаков функцией отличия модифицируются формулы расчета метрик. В нашем случае в обучающей выборке категориальные признаки отсутствуют, поэтому использование функции отличия не требуется.

Другой проблемой использование алгоритма k-means является необходимость выполнения нормировки числовых параметров объектов. Это необходимость связана с масштаб изменения числовых параметров объектов. Так при оценке близости объектов наибольшее влияние на конечный результат сравнения оказывает тот числовой параметр, масштаб которого самый большой. Для того чтобы уровнять степень влияние всех числовых признаков на результат сравнения их близости с использованием выбранной метрики, самым простым решением является нормировка всех числовых параметров путем линейного преобразования к диапазону [0, 1]. Так нормированные значение вектора i-ого числового параметра по всей обучающей выборке рассчитывается как:

$$P_i^{\text{н}} = \frac{P_i - \min(P_i)}{\max(P_i) - \min(P_i)} \quad (2.9)$$

В нашем случае необходимо проведение нормировки числовых параметров объектов, представленных в обучающей выборке, поэтому она будет проводиться по формуле 2.9.

Другой проблемой использование алгоритма k-means является проблема определения оптимального количества кластеров. Это связано с тем, что количество кластеров получаемой структуры является входным параметром алгоритма. Оптимальное количество кластеров определется путем анализа обучающей выборки с использованием стороннего математического аппарата. Другим вариантом определения оптимального количества кластеров – опытным путем (за счет проведения вычислительных экспериментов с заданием различных значений *k*). В нашем случае мы будем использовать второй способ, выбрав в качестве критерия оптимальности точность классификации объектов.

Другой особенностью использование алгоритма k-means является проблема выбора метрики для оценки сходства объектов. Суть проблемы заключается в том, что использование различных метрик влияет на результат каждой итерации алгоритма k-means. Различия в расчете расстояния между объектами приводят к тому, что при одних и тех же исходных данных объекты начинают относиться к разным кластерам. При этом разные метрики приводят к различному расположению центров кластеров. Чем больше итераций требуется для кластеризации данных, тем сильнее отличаются кластерные структуры, полученные при использовании различных метрик.

Таким образом, оптимальной метрика, позволяющая обеспечить наименьшую ошибку кластеризации, подбирает в зависимости от исходных данных опытным путем. Процедура определения оптимальной метрики является вопросом обсуждения в научных статьях. При этом исследователи сходятся во мнении, что в большинстве случаев достаточным является использование метрики Евклида.

Таким образом, для нашего случая, при кластеризации данных, для использования выбрана метрика Евклида.

2.2 Способ построения классификатора и классификации данных на основе результатов кластерного анализа

Как было сказано выше, алгоритм k-means предназначен для автоматизированной группировки объектов по кластерам. При этом кластерная структура подбирается таким образом, чтобы объекты внутри одного кластера были максимально похожи друг на друга.

Идея построения классификатора заключается в том, чтобы сначала поделить объекты обучающей выборки на кластеры и изучив состав каждого кластера сформировать стратегию классификации данных. Изучение состава кластеров необходимо для того, чтобы определить, как распределены метки классов по кластерам. Затем на основе изученных свойств кластеров предлагается классифицировать новые (ранее неизвестные) объекты. При классификации можно с помощью одной из известных метрик определить близость объекта до каждого из известных центров кластеров. Предположительно метка класса классифицируемого объекта будет та же, что и у большинства объектов находящихся в ближайшем центре кластера.

Таким образом, предложено 2 алгоритма:

- Алгоритм построения классификатора на основе кластерного анализа данных.
- Алгоритм классификации объектов с помощью получаемых классификаторов.

Алгоритм построения классификатора на основе кластерного анализа данных состоит из 3 этапов.

1. Подготавливается обучающая выборка – данные, на основе которых будет строиться (обучаться) классификатор (рисунок 2.4).

2. Выборка данных подвергается кластеризации с использованием алгоритма k-means. При этом задается количество кластеров *k*, выбирается метрика расчета расстояния между объектами. После выполнения данного этапа мы получаем распределение объектов по кластерам и координаты центров каждого кластера (рисунок 2.5).

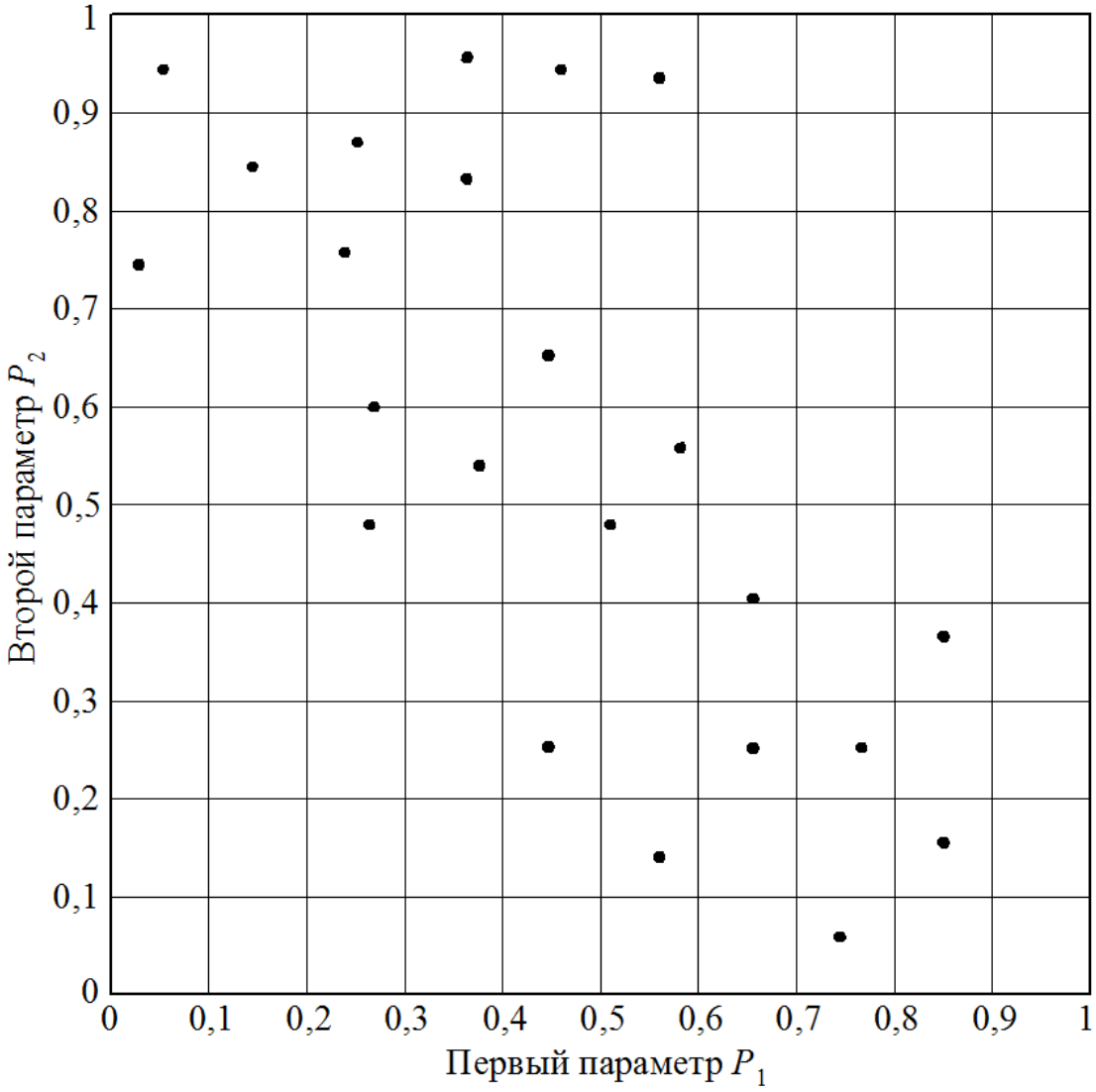


Рисунок 2.4 – Первый этап – подготовка данных для анализа (включая предобработку)

3. Производится статистический анализ каждого кластера. Целью анализа является исследование распределение классов по кластерам. При этом подсчитывается количество объектов в кластере, относящихся к

каждому из классов. Впоследствии для каждого кластера определяется доминирующий класс (рисунок 2.6).

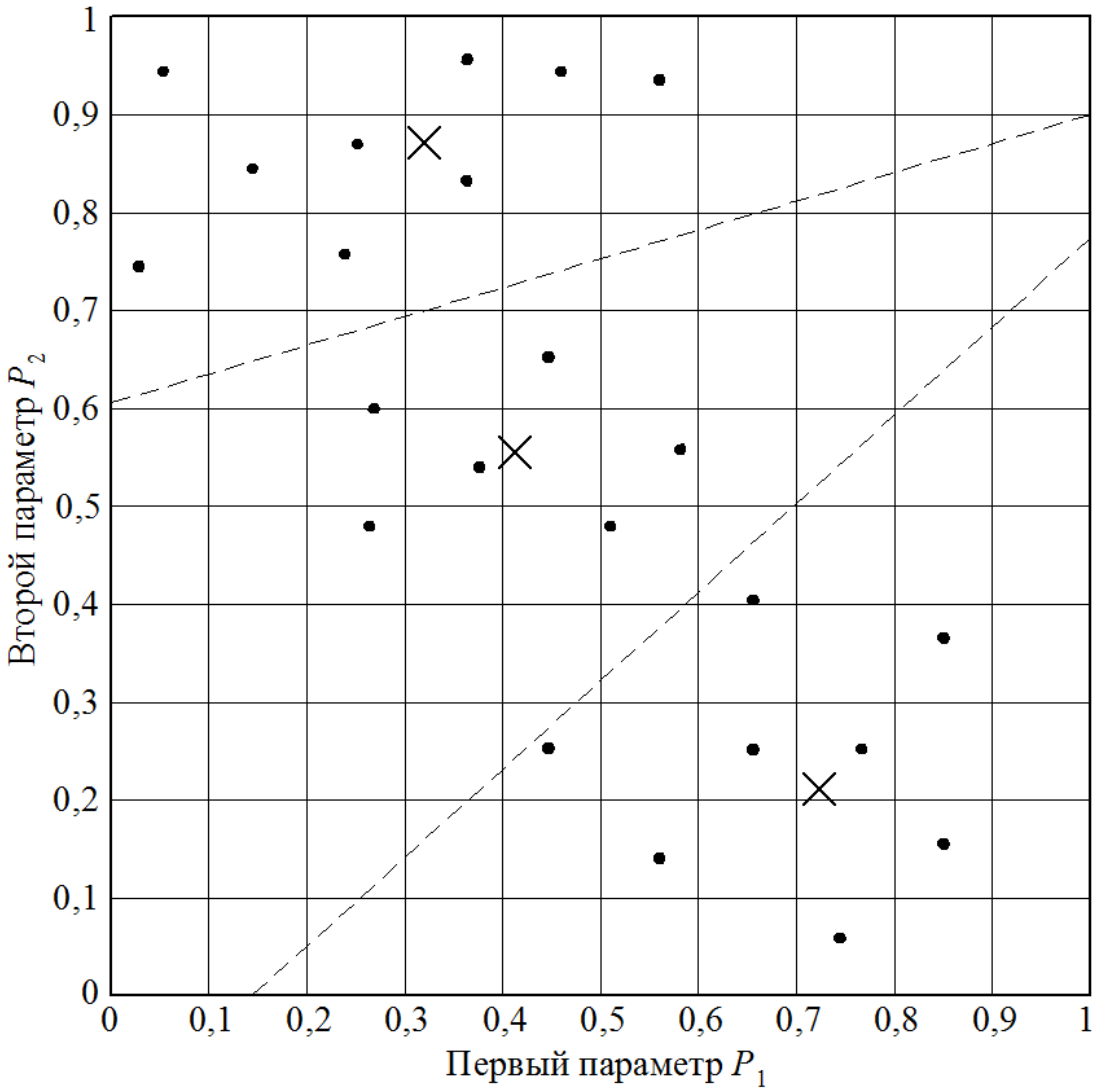


Рисунок 2.5 – Второй этап – кластеризация данный методом k-means с заданным значением *k* количества кластеров. Знаками «×» обозначены центры кластеров. Пунктирными линиями обозначены границы кластеров.

4. Производится объедение полученный на предыдущих этапах данных в классификатор. Таким образом, классификатор состоит из координат центров кластеров и метки класса, возвращаемой каждым центром кластера.

Алгоритм классификации объекта с помощью полученного классификатора состоит из 2 этапов.

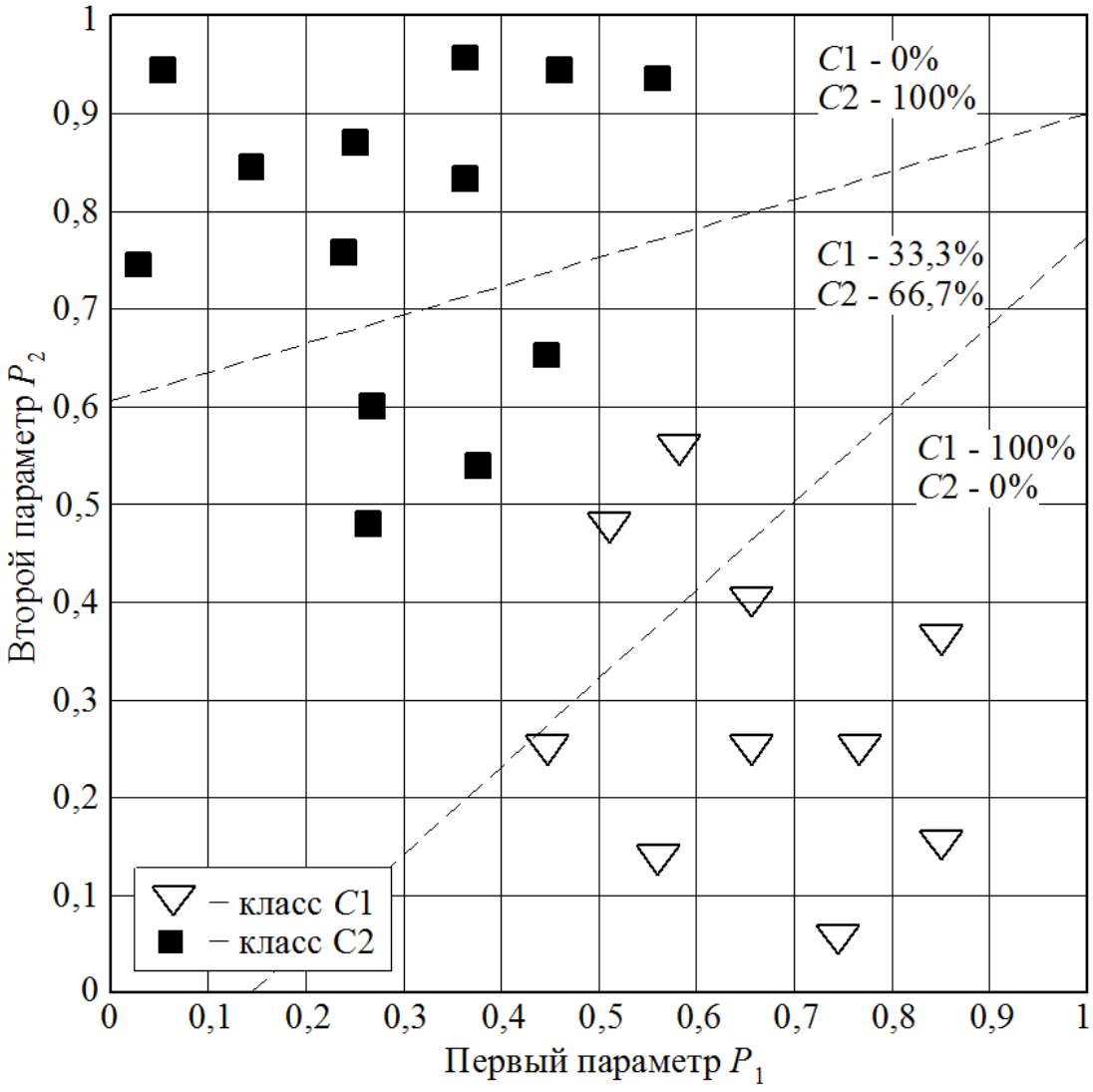


Рисунок 2.6 – Статистический анализ каждого кластера, определение преобладающего класса в каждом кластере

1. С помощью выбранной метрики рассчитывается расстояние между вектором входных параметров исследуемого объекта и центрами кластеров, хранящихся в классификаторе. Определяется ближайший центр кластера.

2. Исследуемой объект будет отнесен к тому классу, который доминирует в кластере с ближайшим центром.

Графическая интерпретация классификации объекта представлена на рисунке 2.7.

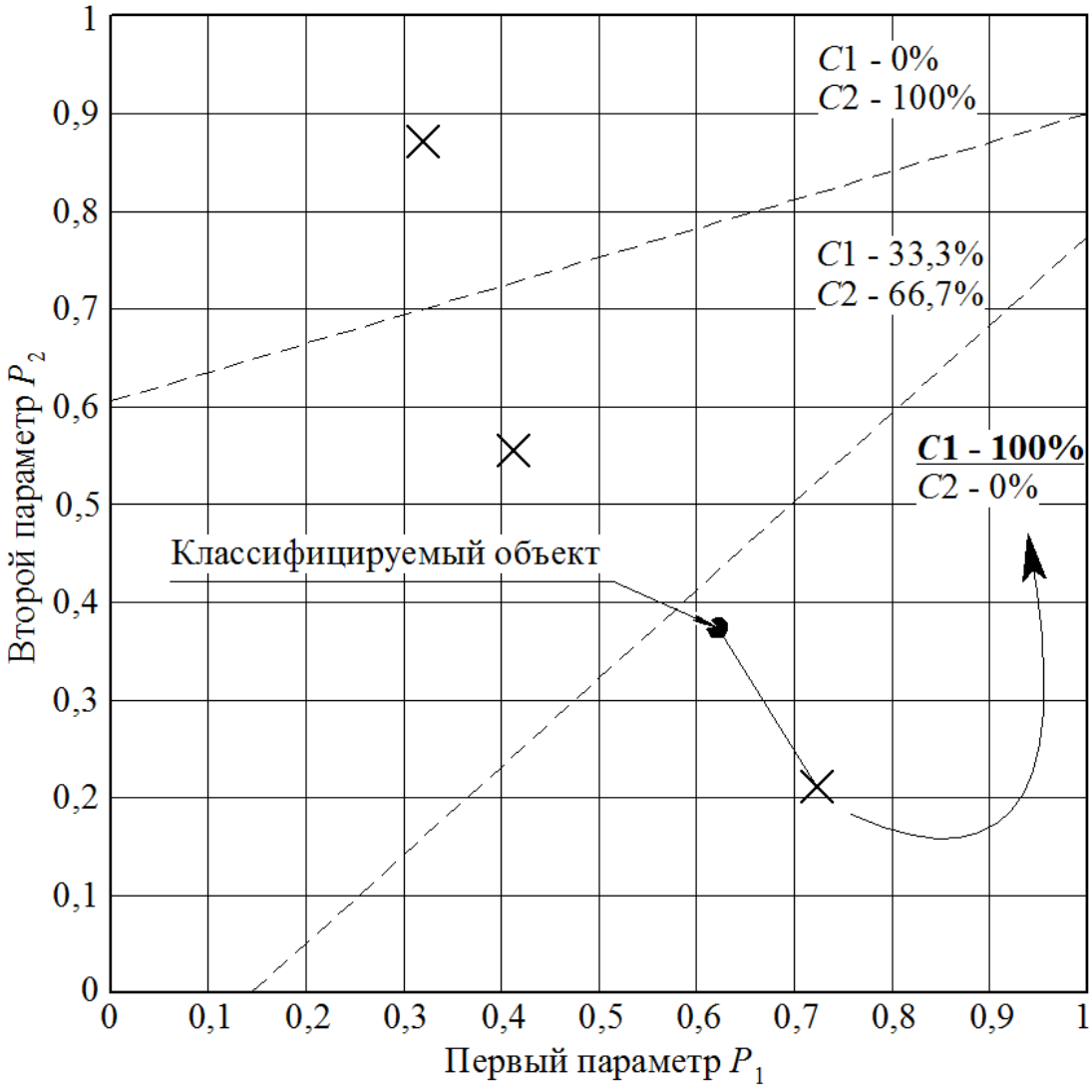


Рисунок 2.7 – Классификация объекта путем определения ближайшего центра кластера

В общем виде основные этапы алгоритма построения классификатора данных и алгоритма классификации данных можно представить так как это показано на рисунке 2.8.

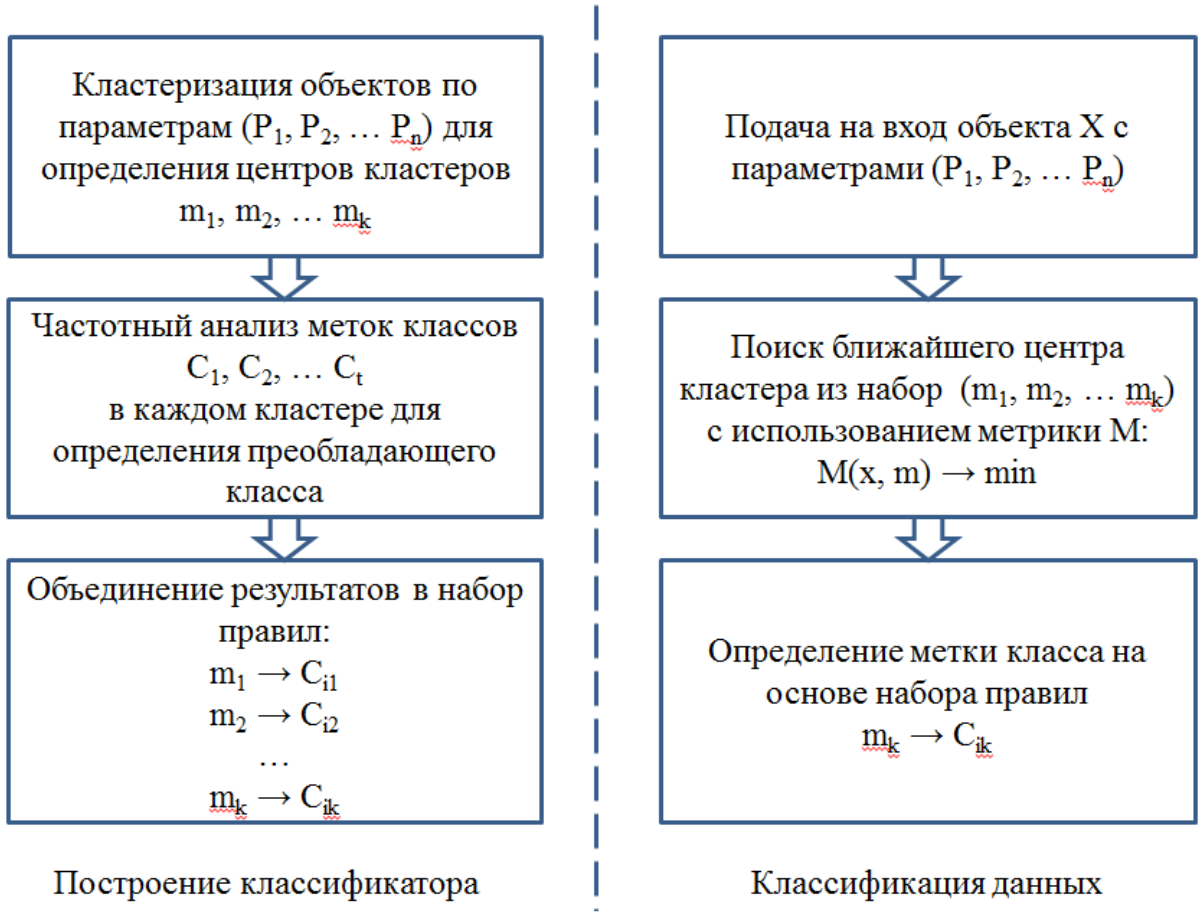


Рисунок 2.8 – Основные этапы алгоритма построения классификатора данных и алгоритма классификации данных

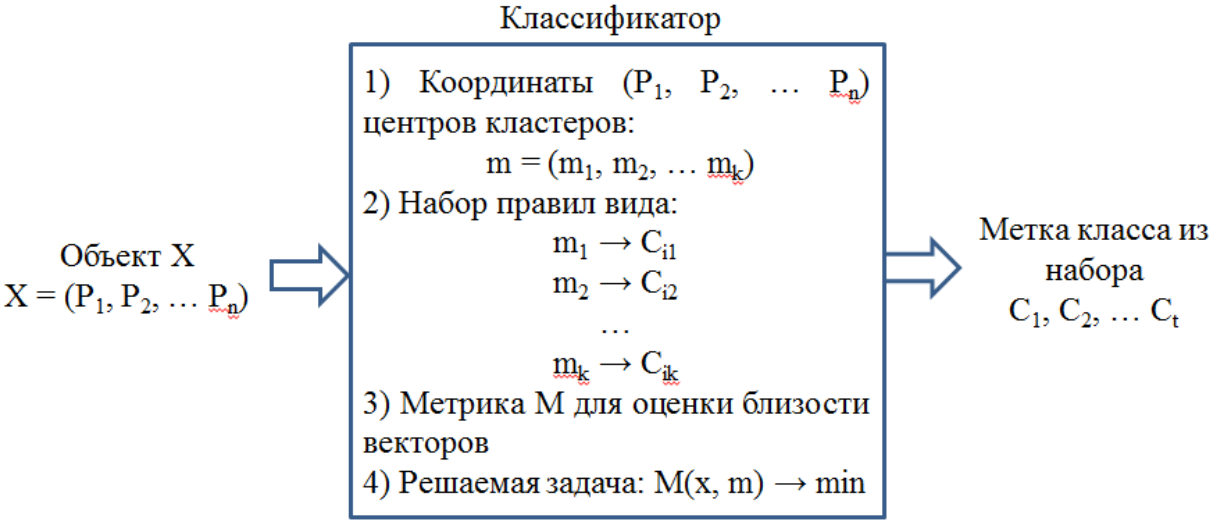


Рисунок 2.9 – Классификатор данных, основанный на алгоритме кластеризации k-means

Таким образом, получаемый, на основе кластерного анализа исходных данных, классификатор будет включать в себя (рисунок 2.9):

- параметры (P_1, P_2, …, P_n) центров кластеров (m_1, m_2, …, m_n);
- набор правил, описывающих доминирующий в каждом кластере класс;
- метрику оценки близости объектов по векторам значений (P_1, P_2, …, P_n).

Выводы по разделу

На основании представленных выше исследований можно отметить следующее. Предложена технология синтеза классификаторов данных на основе результатов кластерного анализа. В соответствии с этой технологией проводится кластеризация данных (без учета значений меток классов) с использование алгоритма k-means. В результате кластеризации исследуемые объекты распределяются по группам (кластерам) и рассчитываются центры кластеров. Затем проводиться статистический анализ каждого кластера для определения преобладающего в нем класса. Классификатор включает в себя параметры центров кластеров, а также для каждого кластера – метку преобладающего класса. При классификации исследуемого объекта определяется его принадлежность к одному из кластеров путем расчёта расстояния от объекта до центра кластеров. Считается, что исследуемый объект относится к тому кластеру, расстояние, до центра которого наименьшее. Исследуемому объекту присваивается метка класса, преобладающего в данном кластере.

3 Проведение тестирования предложенных подходов

3.1 Программной реализация алгоритма построения классификатора и классификации данных на основе алгоритма k-means

Для тестирования предложенных подходов на языке Python было разработано программное обеспечение, реализующее построение классификатора и классификацию входных данных для загруженной произвольной выборки данных.

Последовательность работы с программным обеспечением при построении классификатора следующая.

1. Загрузка данные для классификации (атрибуты, 1 столбец - метка класса).
2. Кодирование с помощью метода one-hot-encoder категориальных признаков.
3. Нормировка всех числовых значений с к диапазону от 0 до 1 с помощью метода MinMaxScaler.
4. Задание значения коэффициента kc для категориальных признаков и нормировка значений к диапазону от 0 до kc (kc=0..1).
5. Запуск алгоритма k-means на всех исходных данных за исключением столбца с метками класса.
6. Формирование на основе результатов кластеризации классификатора. Наш классификатор – это набор центров кластеров. Центр кластера возвращает метку класса, которая чаще всего встречает в его группе записей (из обучающей выборки).

Последовательность работы с программным обеспечением при классификации данных следующая.

1. Получаем набор данных, который необходимо классифицировать

2. С помощью функции попарного сравнения производим рассчет расстояния между исследуемым объектом и центрами кластеров

3. Возвращается та метка класса, которая связана с ближайшим центром кластера.

При разработке программного обеспечения учитывались следующие требования.

1. Программа должна работать на различных выборках с различными типами данных (числовые и категориальные атрибуты записей).

2. Данные приводятся к определенному типизированному виду путем их преобразования.

3. Данные следует разделить на тестовую и обучающую выборку, чтобы оценить точно

4. На основе тестовый выборки сформировать набор кластеров на основе алгоритма k-means

5. На основе тестовой выборки провести классификацию данных

6. Сравнить полученные результаты классификации с исходными, «истинными» результатами, определить точность работы классификации

7. Предусмотреть возможность изменения параметров классификатора используя интерфейс приложения

Весь программный код разделен на 4 основных блока.

1. Загрузка данных для классификации

2. Преобразование данных

3. Запуск алгоритма K-means и получение центров кластеров.

4. Проверка работы классификатора для определения точности его работы на тестовой выборке данных.

Данное разделение отражено в интерфейсе и приведено на рисунке 3.1

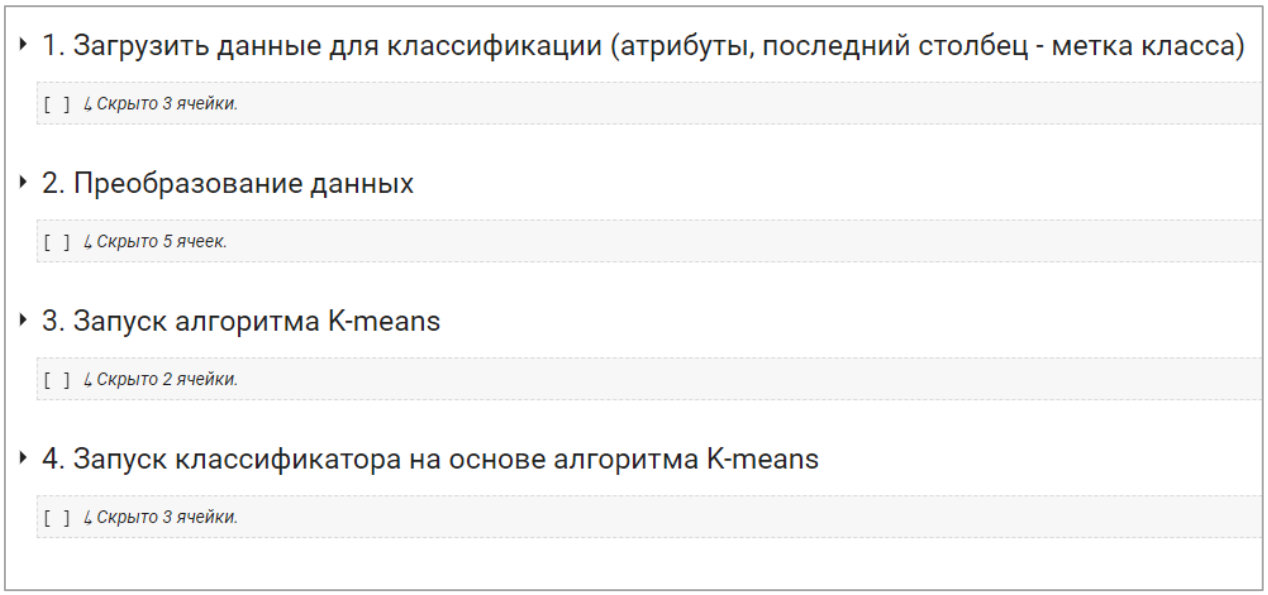


Рисунок 3.1 – Начальный интерфейс приложения

Рассмотрим в отдельности каждый блок программы. Во-первых, подключим необходимые библиотеки. Таковыми являются библиотеки pandas и numpy, которые служат для удобной работы с данными и их отображением. Так же нам понадобится библиотека sklearn из которой нам понадобятся функции преобразования данных и функции реализующие алгоритм k-means. Также импортируем функцию distance из библиотеки scipy для реализации расчета расстояния по метрике Евклида.

Далее нам необходимо будет загрузить исходную выборку данных. Так как Google Colab основан на облачных технологиях, будет необходимо подключить google диск, на котором будет хранить данная выборка и указать путь до нее. Для этого реализуем форму ввода пути к файлу. Данная форма приведена на рисунке 3.2. Весь интерфейс реализован посредством виджетов google forms, что позволяет при желании скрывать исходный код программы, подписывать и давать комментарии к каждому блок, а также пользоваться средствами управления, такими как способы введения данных, слайдеры и т.д.

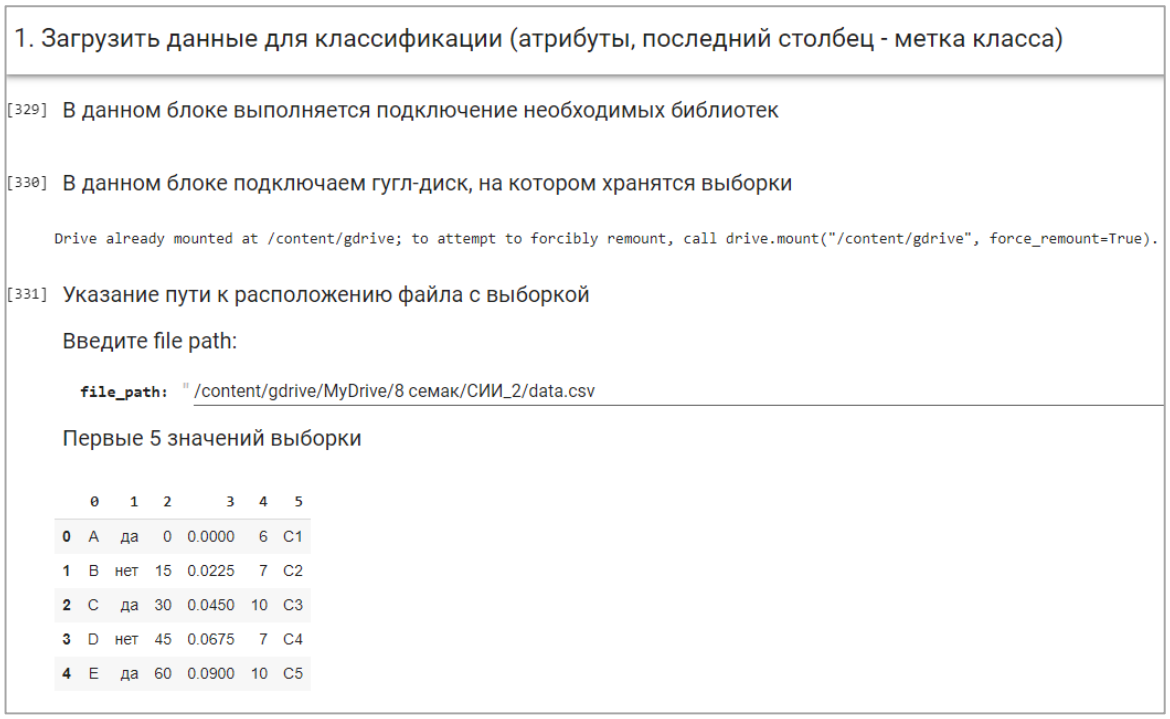


Рисунок 3.2 – Формы подключения Google Drive и ввода путь к файлу

Главное требование к исходной выборке – это чтобы метка класса находилась в последнем столбце. Будет ли она содержать категориальные или численные атрибуты неважно, так как программа умеет обрабатывать оба вида данных.

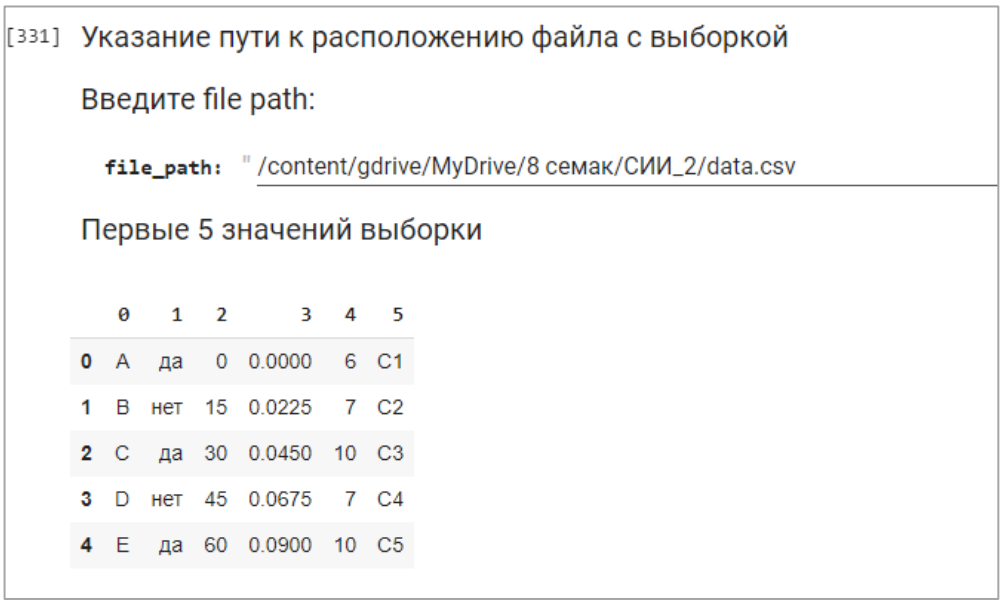


Рисунок 3.3 – Вывод первых несколько значений загруженной выборки данных

Далее, на рисунке 3.3 приведен программный код считывания файла и формирования из него датафрейма, а также вывод первых пяти значений выборки в интерфейсе. Для примера использована выборка, содержащая как категориальные признаки, так и числовые.

Далее данные необходимо преобразовать. Атрибуты делятся на два вида данных: числовые и категориальные. Чтобы все атрибуты имели схожий вес при кластеризации, числовые значения приводится к единому диапазону от 0 до 1. С категориальными атрибутами все не так просто. Чтобы представить его в числовом виде, все возможные варианты атрибутов добавляются к датафрейму в виде столбов, в значениях которых указывается 1 или 0 в зависимости от того, присутствует ли данные значение атрибута в данной записи.

Помимо этого, если у атрибутов много возможных вариантов значений, их вес может измениться, что повлияет на точность работы классификатора. Потому получившиеся значения нормируем к значениям от 0 до kc, где kc – коэффициент веса значения (от 0.1 до 1). Так как данное значение может повлиять на точность работы классификатора следует учесть возможность ввода данного значения с терминала. Таким образом реализация преобразования данных будет выглядеть следующим образом

Сначала отделим метки класса сохранив их отдельно от значений атрибутов (рисунки 3.4, 3.5)

```
#@title В данном блоке отделяются метки класса
L=file.shape[1]-1
LC=file.shape[1]-1
x = file[file.columns[0:L]]
y = file[file.columns[LC]]
#y
```

Рисунок 3.4 – Фрагмент кода отделения меток класса

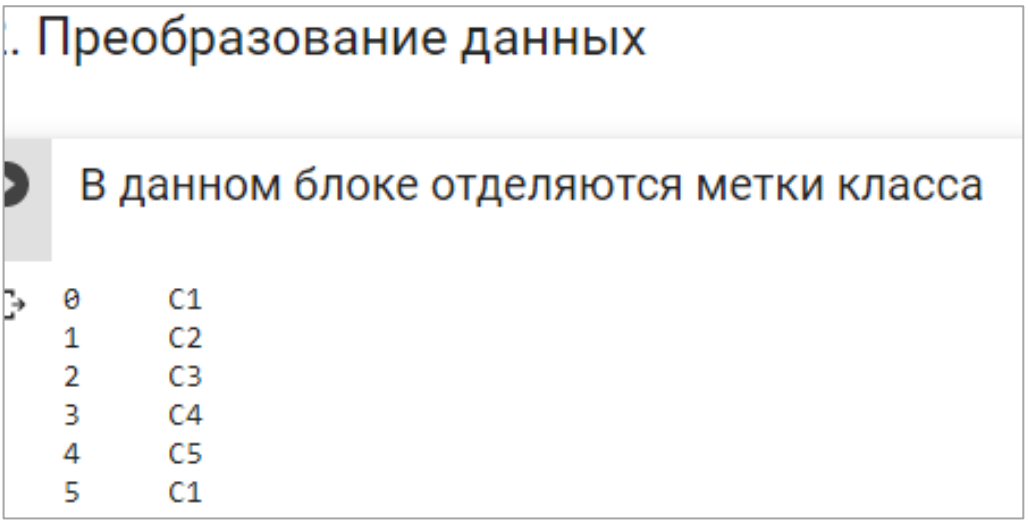
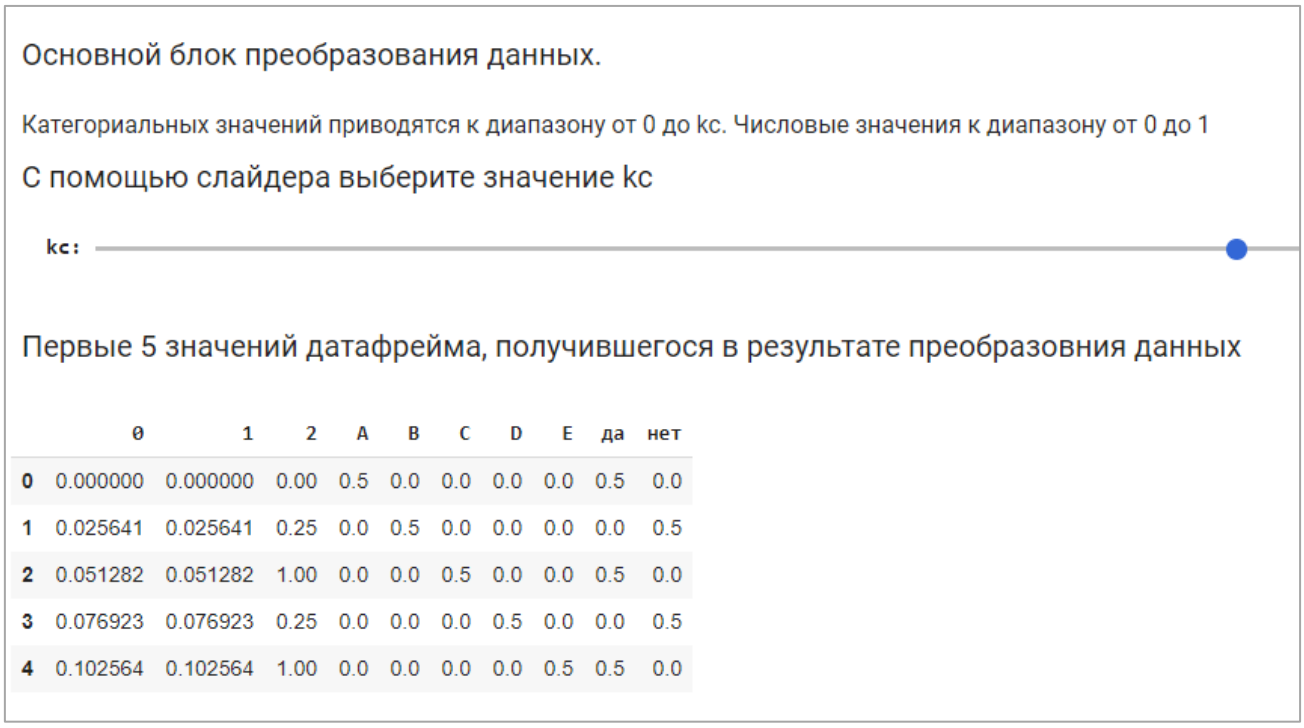


Рисунок 3.5 – Интерфейс блока отделения меток класса от остальных атрибутов

Далее преобразуем данные. Интерфейс преобразования и получившиеся результаты приведены на рисунке 3.6



	0	1	2	A	B	C	D	E	да	нет
0	0.000000	0.000000	0.00	0.5	0.0	0.0	0.0	0.0	0.5	0.0
1	0.025641	0.025641	0.25	0.0	0.5	0.0	0.0	0.0	0.0	0.5
2	0.051282	0.051282	1.00	0.0	0.0	0.5	0.0	0.0	0.5	0.0
3	0.076923	0.076923	0.25	0.0	0.0	0.0	0.5	0.0	0.0	0.5
4	0.102564	0.102564	1.00	0.0	0.0	0.0	0.0	0.5	0.5	0.0

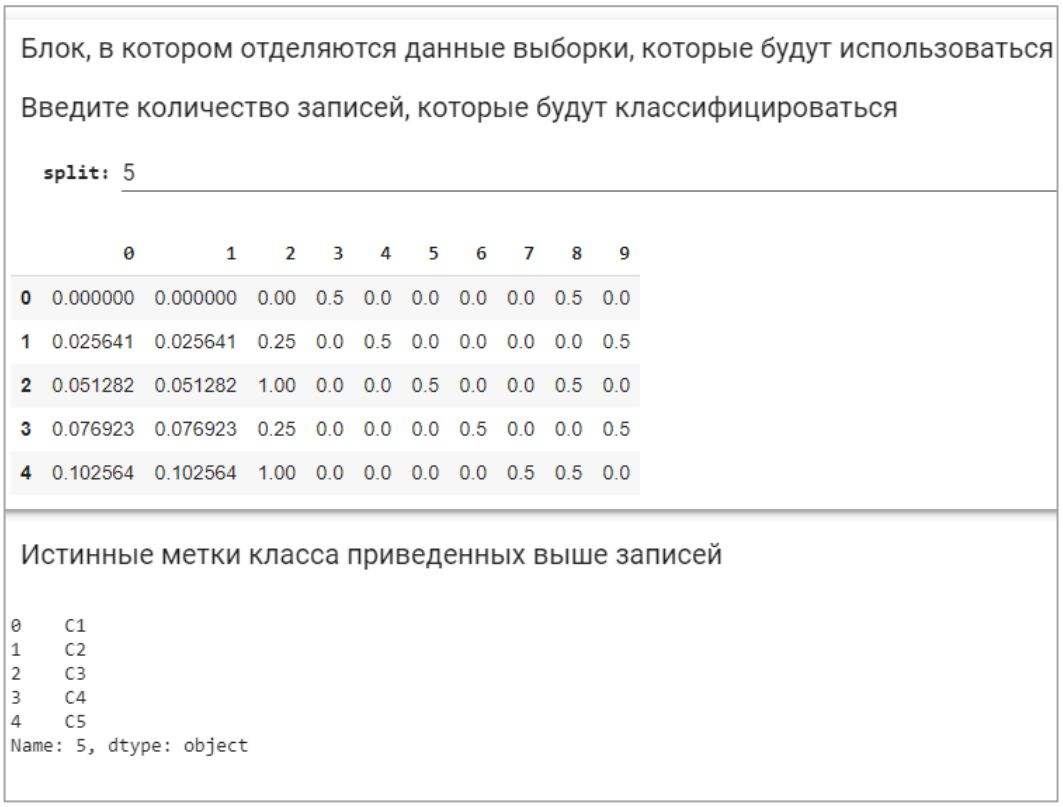
Рисунок 3.6 – Интерфейс преобразования данных и вывод результов

Реализация алгоритма преобразования состоит в следующем. Определяем, какие столбцы содержат числовые атрибуты, а какие категориальные. Разделяем датафрейм на два: только с числовыми и только с категориальными признаками. Для каждого датафрейма проводим преобразование по схеме, описанное выше и снова сливаем полученные после обработки датафреймы в один. В результате получаем единый датафрейм с преобразованными данными. Фрагмент реализации данного процесса в коде представлен на рисунке 3.7

```
] #@title Основной блок преобразования данных.
  #@markdown Категориальных значений приводятся к диапазону от 0 до kc.
  #@markdown Числовые значения к диапазону от 0 до 1
  #@markdown ### С помощью слайдера выберите значение kc
  kc=0.5 #@param {type:"slider", min:0.1, max:1, step:0.1}
  x_num=x
  x_cat=x
  for column in x:
    if x.dtypes[column]==np.int64 or x.dtypes[column]==np.float64:
      x_cat=x_cat.drop(column, axis=1)
    else:
      x_num=x_num.drop(column, axis=1)
  x = x.iloc[0:0]
  x_cat.columns = range(x_cat.shape[1])
  x_num.columns = range(x_num.shape[1])
  ohe = OneHotEncoder(sparse=False)
  for column1 in x_cat:
      temp = ohe.fit_transform(x_cat[[column1]])
      ohe_column = pd.DataFrame(kc*temp, columns=x_cat[column1].unique())
      x_cat=x_cat.drop(column1, axis=1)
      x_cat = pd.concat([x_cat, ohe_column], axis=1)

  scaler = pre.MinMaxScaler(feature_range=(0, 1))
  x_num = pd.DataFrame(scaler.fit_transform(x_num))

] #@title Первые 5 значений датафрейма, получившегося в результате преобразовния данных
  x=pd.concat([x_num,x_cat], axis=1)
  x.head()
```

Рисунок 3.7 – Фрагмент кода преобразования данных

Далее, проведем разделение данных на тестовую и обучающую выборку, для этого просто отделим несколько значений из выборки и сохраним их отдельно, в дальнейшем они потребуются для проведения классификации. Предусмотрим возможность ввода с терминала количества записей, которые будет использоваться для классификации. Данная форма и код приведены на рисунках 3.8 и 3.9 соответственно.



	0	1	2	3	4	5	6	7	8	9
0	0.000000	0.000000	0.00	0.5	0.0	0.0	0.0	0.0	0.5	0.0
1	0.025641	0.025641	0.25	0.0	0.5	0.0	0.0	0.0	0.0	0.5
2	0.051282	0.051282	1.00	0.0	0.0	0.5	0.0	0.0	0.5	0.0
3	0.076923	0.076923	0.25	0.0	0.0	0.0	0.5	0.0	0.0	0.5
4	0.102564	0.102564	1.00	0.0	0.0	0.0	0.0	0.5	0.5	0.0

Рисунок 3.8 – Интерфейс разделения данных на тестовую и обучающую выборку

На этом шаге подготовка и преобразование данных завершены.

Следующий блоком приложения является запуск алгоритма k-means и получения результирующих кластеров. Для этого воспользуемся функцией KMeans(). На вход KMeans() достаточно подать количество кластеров, на которые будет проводиться разбиение входных данных. Предусмотрим возможность введения количества кластеров через интерфейс (рисунок 3.10).

```
#@title Блок, в котором отделяются данные выборки, которые б
#@markdown ### Введите количество записей, которые будут кла
test_split_x=x.copy()
split=5 #@param {type:"number"}
for i in range(split,x.shape[0]):
  test_split_x=test_split_x.drop(index=i)
for i in range(0,split):
  x=x.drop(index=i)
x.index = np.arange(len(x))
test_split_x.columns = range(test_split_x.shape[1])
test_split_x
```

```
#@title Истинные метки класса приведенных выше записей
test_split_y=y.copy()
for i in range(split,y.shape[0]):
  test_split_y=test_split_y.drop(index=i)
for i in range(0,split):
  y=y.drop(index=i)
y.index = np.arange(len(y))
test_split_y
```

Рисунок 3.9 – Фрагмент кода разделения данных на тестовую и обучающую выборки

В качестве выходных данный после работы алгоритма кластеризации получаем два списка - cluster_centers, содержащий координаты центров кластеров и labels содержащий метки кластеров. Фрагмент кода приведен на рисунке 3.11

На этом реализация алгоритма k-means завершена. Далее следует 4 блок программы – классификатор на основе результатов работы k-means. Для этого необходимо собрать данные в датафрейм, значениями которого будут метки классов объектов, попавших в кластер и номер данного кластера.

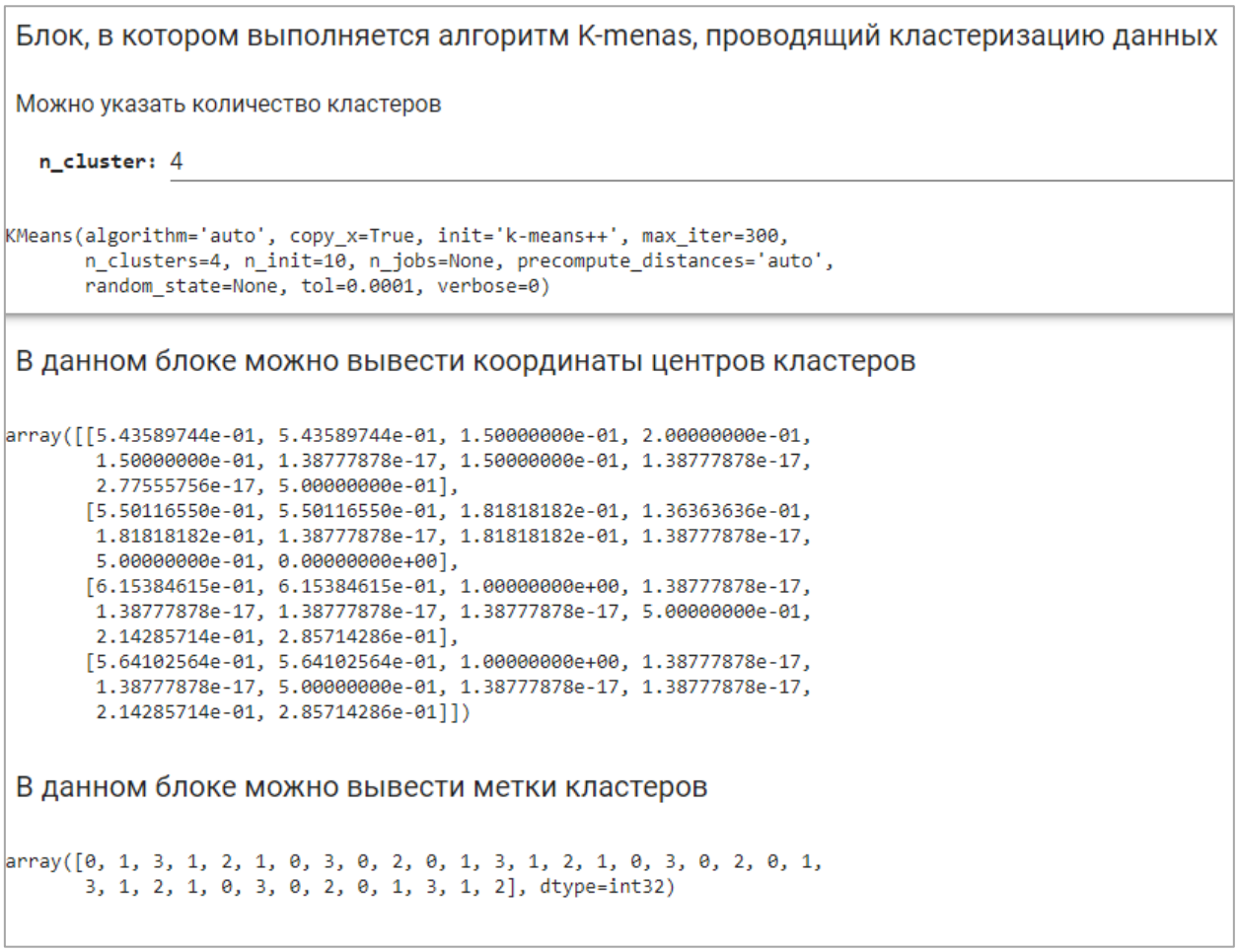


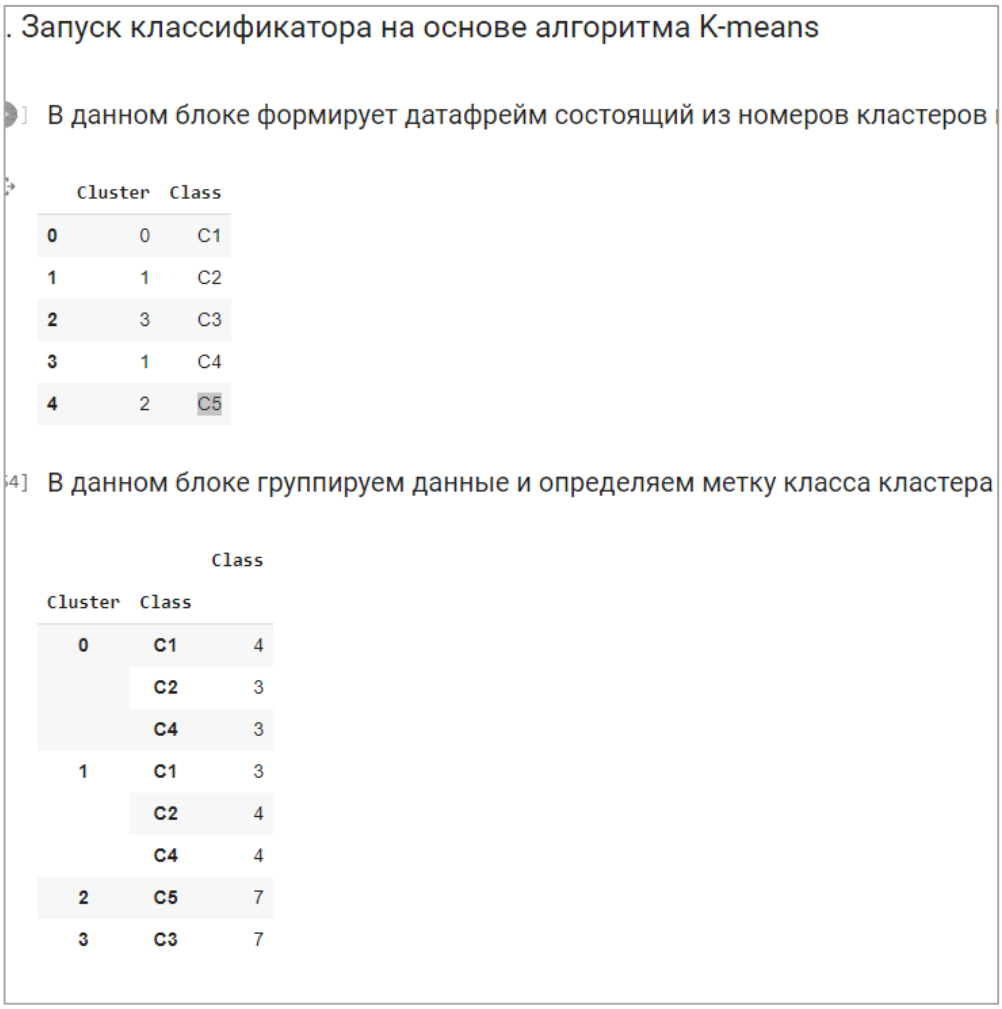
Рисунок 3.10 – Интерфейс и результат работы алгоритма k-means

```
#@title Блок, в котором выполняется алгоритм K-menas, проводящий кластеризацию данных
#@markdown Можно указать количество кластеров
n_cluster=4 #@param {type:"integer"}
km = KMeans(n_clusters=n_cluster)
km.fit(x)


#@title В данном блоке можно вывести координаты центров кластеров
km.cluster_centers_


#@title В данном блоке можно вывести метки кластеров
km.labels_
```

Рисунок 3.11 – Фрагмент кода реализации алгоритма k-means

На основе этого группируем данные, подсчитывая сколько меток классов попало в тот или иной кластер и присваиваем центру данного кластера ту метку, которая чаще всего встречается в кластере. Таким образом мы подготовим наш классификатор для решения задач классификации. Примеры интерфейса и результатов приведены на рисунке 3.12, фрагмент кода реализации данного процесса на рисунке 3.13

Последней задачей является проведение классификации. Для этого используем тестовую выборку, рассчитав расстояние от каждого объекта выборки до каждого центра кластера, найдем ближайшие к каждой записи центроиды и присвоим записям метку классов центроидов.


. Запуск классификатора на основе алгоритма K-means

В данном блоке формирует датафрейм состоящий из номеров кластеров

	Cluster	Class
0	0	C1
1	1	C2
2	3	C3
3	1	C4
4	2	C5

В данном блоке группируем данные и определяем метку класса кластера

		Class
Cluster	Class	
0	C1	4
	C2	3
	C4	3
1	C1	3
	C2	4
	C4	4
2	C5	7
3	C3	7

Рисунок 3.12 - Примеры интерфейса и результатов

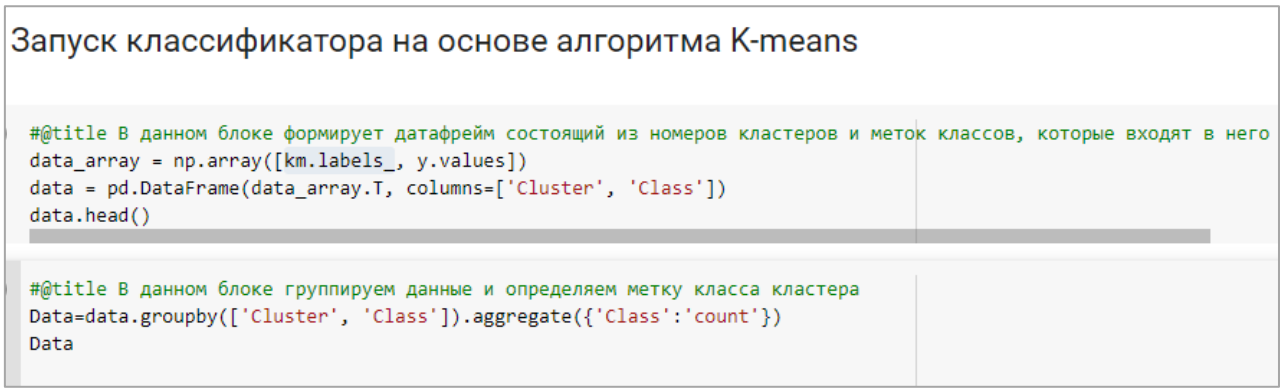
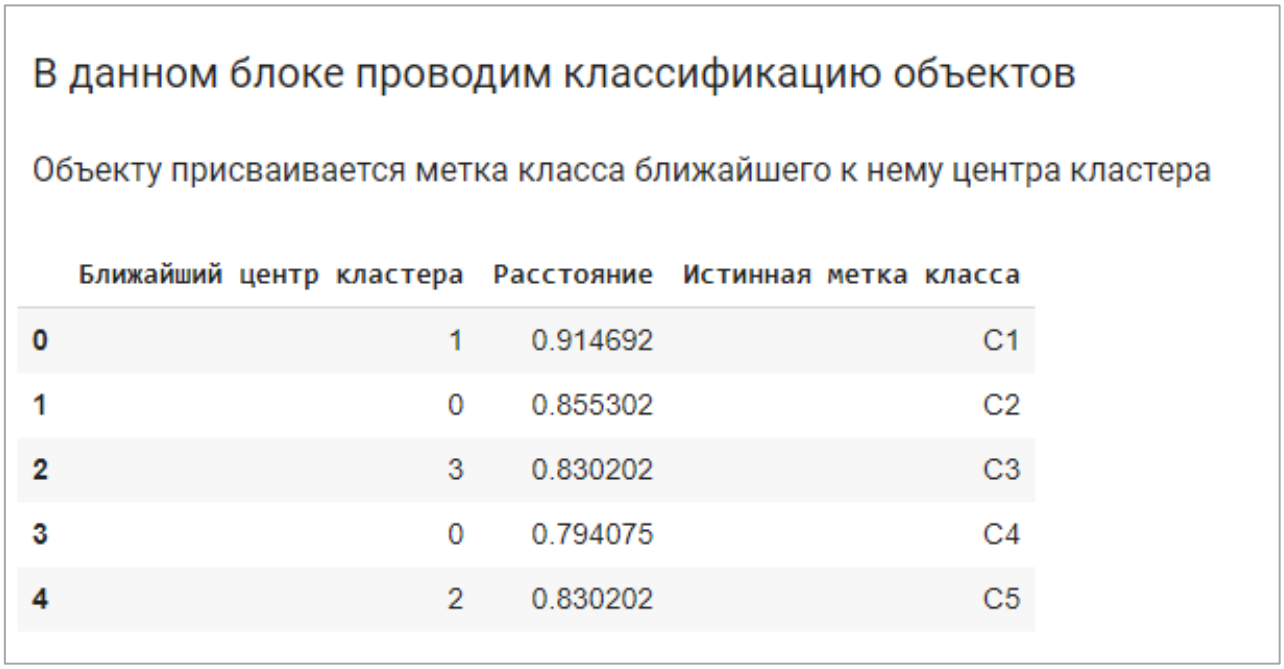


Рисунок 3.13 - Фрагмент кода реализации

Пример интерфейса последнего блока приведен на рисунке 1.14, а фрагмент кода реализации на рисунке 1.15



	Ближайший центр кластера	Расстояние	Истинная метка класса
0	1	0.914692	C1
1	0	0.855302	C2
2	3	0.830202	C3
3	0	0.794075	C4
4	2	0.830202	C5

Рисунок 3.14 – Результата классификация объекта

```
#@title В данном блоке проводим классификацию объектов
#@markdown Объекту присваивается метка класса ближайшего к нему центра кластера
d = []
for i in range(test_split_x.shape[0]):
  dst_min=100
  for j in range(len(km.cluster_centers_)):
    dst = distance.euclidean(km.cluster_centers_[j], test_split_x.iloc[i])
    if dst<dst_min:
      dst_min=dst
      p=j
  d.append({'Ближайший центр кластера':p,'Расстояние':dst_min,'Истинная метка класса':test_split_y[i]})
pd.DataFrame(d)
```

Рисунок 3.15 – Фрагмент кода классификации

На этом задача классификации считается решенной, а программа закончит свою работу. Для проверки работы классификатора и оценки точности проведем некоторые вычислительные эксперименты на различных наборах данных.

3.2 Вычислительный эксперимент на наборе данных «Fisher's Iris»

Проведем вычислительный эксперимент на классическом наборе данных в машинном обучении – Fisher's Iris. Выборка состоит из 5 атрибутов, 4 из которых числовые, а 1 – категориальный. Категориальный атрибут – вид ириса - примем за метку класса. Остальные 4 атрибута: ширина и длина чашелистика, ширина и длина лепестка будут использоваться как атрибуты для ассоциативных правил. Далее на рисунках 3.16-3.22 приведено как меняется выборка данных на всех этапах построения классификатор, а также полученные и модифицированные наборы данных и результат классификации.

В качестве параметров для первого прохождения зададим количество кластеров равно 4, размер тестовой выборки – 5. Значение коэффициента kc не играет роли, так как атрибуты выборки являются числовыми.

Введите file path:

file_path: /content/gdrive/MyDrive/8 семак/СИИ_2/iris.csv

Первые 5 значений выборки

	0	1	2	3	4
0	4.9	3.0	1.4	0.2	Iris-setosa
1	4.7	3.2	1.3	0.2	Iris-setosa
2	4.6	3.1	1.5	0.2	Iris-setosa
3	5.0	3.6	1.4	0.2	Iris-setosa
4	5.4	3.9	1.7	0.4	Iris-setosa

Рисунок 3.16 – Загрузка выборки данных

	0	1	2	3
0	0.166667	0.416667	0.067797	0.041667
1	0.111111	0.500000	0.050847	0.041667
2	0.083333	0.458333	0.084746	0.041667
3	0.194444	0.666667	0.067797	0.041667
4	0.305556	0.791667	0.118644	0.125000

Рисунок 3.17 – Преобразованные данные выборки

Введите количество записей, которые будут классифицироваться

split: 5

	0	1	2	3
0	0.166667	0.416667	0.067797	0.041667
1	0.111111	0.500000	0.050847	0.041667
2	0.083333	0.458333	0.084746	0.041667
3	0.194444	0.666667	0.067797	0.041667
4	0.305556	0.791667	0.118644	0.125000

Рисунок 3.18 – Тестовая выборка

```
0    Iris-setosa
1    Iris-setosa
2    Iris-setosa
3    Iris-setosa
4    Iris-setosa
```

Рисунок 3.19 – Истинные метки класса тестовой выборки

На рисунке 3.16 показан результат загрузки исходной выборки данных из файла iris.csv. На рисунке 3.17 показан результат нормировки входных параметров к диапазону от 0 до 1. На рисунке 3.18 показан результат отбора 5 случайных записей, на которых будет оцениваться точность работы классификаторов, также на рисунке 3.19 показаны метки классов отобранных записей. На рисунке 3.20 показан результат кластеризации исходных данных, в т.ч. выводится распределение записей по кластерам. Далее выводится статистическая информация о распределении классов по кластерам (рисунок 3.21)

Можно указать количество кластеров

```
n_cluster: 4
```

```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
       n_clusters=4, n_init=10, n_jobs=None, precompute_distances='auto',
       random_state=None, tol=0.0001, verbose=0)
```

В данном блоке можно вывести координаты центров кластеров

```
array([[0.54166667, 0.375     , 0.65657789, 0.64186508],
       [0.19823232, 0.59280303, 0.07896764, 0.06060606],
       [0.73850575, 0.47270115, 0.82291058, 0.86350575],
       [0.35632184, 0.23706897, 0.50905903, 0.47126437]])
```

В данном блоке можно вывести метки кластеров

```
array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       0, 0, 0, 3, 0, 3, 0, 3, 0, 3, 3, 0, 3, 0, 3, 0, 0, 3, 3, 3, 0, 3,
```

Рисунок 3.20 –Запуск и результаты работы алгоритма k-means

Cluster	Class	
0	Iris-versicolor	23
	Iris-virginica	19
1	Iris-setosa	44
2	Iris-virginica	29
3	Iris-versicolor	27
	Iris-virginica	2

Рисунок 3.21 – Группировка данных

В данном блоке проводим классификацию объектов

Объекту присваивается метка класса ближайшего к нему центра кластера

	Ближайший центр кластера	Расстояние	Истинная метка класса
0	1	0.180288	Iris-setosa
1	1	0.131727	Iris-setosa
2	1	0.177977	Iris-setosa
3	1	0.077160	Iris-setosa
4	1	0.238298	Iris-setosa

Рисунок 3.22 – Результаты классификации

В результате классификации на наборе Fisher's Iris с четыремя кластерами итоге получаем 100% попадание в истинные классы. Проверим точность работы на других алгоритмах классификации, разработанных ранее в ходе освоения данной дисциплине, а именно классификаторами Knn, деревом принятия решений и лесом деревьем. Отчеты о работе данных классификаторов приведены на рисунках 2.8-2.10 соответственно.

KNeighborsClassifier

	precision	recall	f1-score	support
Iris-setosa	1.00	1.00	1.00	6
Iris-versicolor	1.00	1.00	1.00	6
Iris-virginica	1.00	1.00	1.00	3
accuracy			1.00	15
macro avg	1.00	1.00	1.00	15
weighted avg	1.00	1.00	1.00	15

Рисунок 3.23 – Отчет классификатора knn

DecisionTreeClassifier

	precision	recall	f1-score	support
Iris-setosa	1.00	1.00	1.00	6
Iris-versicolor	1.00	1.00	1.00	6
Iris-virginica	1.00	1.00	1.00	3
accuracy			1.00	15
macro avg	1.00	1.00	1.00	15
weighted avg	1.00	1.00	1.00	15

Рисунок 3.24 – Отчет классификатора дерева принятия решения

RandomForestClassifier

	precision	recall	f1-score	support
Iris-setosa	1.00	1.00	1.00	6
Iris-versicolor	1.00	1.00	1.00	6
Iris-virginica	1.00	1.00	1.00	3
accuracy			1.00	15
macro avg	1.00	1.00	1.00	15
weighted avg	1.00	1.00	1.00	15

Рисунок 3.25 – Отчет классификатора леса деревьев

Как видно из отчетов, все классификаторы показали стопроцентную точности выполнения задачи классификации, что говорит о том, что, во-первых, реализованных в ходе выполнения данной работы классификатор ничуть не хуже других классификаторов, а во-вторых, классификация на основе данных Fisher's Iris - очень проста задача для классификаторов.

Для оценки зависимости точности работы классификатора построим. таблицу с данными о том, как меняется точность работы классификаторы в зависимости от значения количества кластеров.

Таблица 3.1 – Зависимость точность работы классификатора в зависимости от количества k кластеров (набор данных – Fisher's Iris)

k	2	3	4	6	8	10
Точность, %	100	100	100	100	100	100

По полученным данным невозможно сделать выводов о зависимости точности от количества кластеров или коэффициента kc.

Проведем ещё один вычислительный эксперимент.

3.3 Вычислительный эксперимент на наборе данных «Machine»

Рассмотрим набор данных machine. Это набор компьютерных данных, состоящий как из категориальных, так и числовых признаков, которых большинство. Состоит из 39 записей и содержит 10 атрибутов. Приведем смысл значения каждого атрибута.

- NAME_M – наименование модели;
- MYCT – время машинного цикла в наносекундах;
- MMIN – минимальный объем оперативной памяти в килобайтах;
- MMAX – максимальный объем оперативной памяти в килобайтах;
- CACH – кэш-память в килобайтах;
- CHMIN – минимальное количество каналов в единицах;
- CHMAX – максимальное количество каналов в единицах;
- PRP – опубликованная относительная производительность;
- ERP – оценочная относительная эффективность исходной статьи;
- NAME_C – наименование компании – продавца.

В качестве метки класса будет использовать наименование компании-продавца

Далее на рисунках 3.26-3.32 приведено как меняется выборка в процессе прохождения через классификацию, а также полученные и модифицированные наборы данных и результат классификации.

В качестве параметров зададим количество кластеров – 4, размер тестовой выборки – 4. Значение коэффициента kc установим равным 0.5.

	0	1	2	3	4	5	6	7	8	9
0	470v/7	29	8000	32000	32	8	32	269	253	amdahl
1	universe:2203t	320	512	2000	4	1	3	69	21	c.r.d
2	b2900	143	512	5000	0	7	32	28	29	burroughs
3	cyber:170/750	25	1310	2620	131	12	24	274	102	cdc
4	470v/7b	29	8000	32000	32	8	32	172	253	amdahl

Рисунок 3.26 – Загрузка выборки в классификатор

	0	1	2	3	4	5	6	7	470v/7	universe:2203t	b2900
0	0.020202	0.246988	0.483871	0.225352	0.225806	0.475410	0.222222	0.190633	0.5	0.0	0.0
1	1.000000	0.012048	0.000000	0.028169	0.000000	0.000000	0.044444	0.000000	0.0	0.0	0.0
2	0.404040	0.012048	0.048387	0.000000	0.193548	0.475410	0.008000	0.006574	0.0	0.0	0.0
3	0.006734	0.037086	0.010000	0.922535	0.354839	0.344262	0.226667	0.066557	0.0	0.0	0.0
4	0.020202	0.246988	0.483871	0.225352	0.225806	0.475410	0.136000	0.190633	0.0	0.0	0.5

Рисунок 3.27 -Преобразованные данные выборки

	0	1	2	3	4	5	6	7	8
0	0.020202	0.246988	0.483871	0.225352	0.225806	0.475410	0.222222	0.190633	0.5
1	1.000000	0.012048	0.000000	0.028169	0.000000	0.000000	0.044444	0.000000	0.0
2	0.404040	0.012048	0.048387	0.000000	0.193548	0.475410	0.008000	0.006574	0.0
3	0.006734	0.037086	0.010000	0.922535	0.354839	0.344262	0.226667	0.066557	0.0

Рисунок 3.28 – Тестовая выборка

```
0       amdahl
1        c.r.d
2    burroughs
3          cdc
Name: 9, dtype: object
```

Рисунок 3.29 – Истинные метки класса тестовой выборки

На рисунке 3.26 показан результат загрузки исходной выборки данных из файла iris.csv. На рисунке 3.27 показан результат нормировки входных параметров к диапазону от 0 до 1. На рисунке 3.28 показан результат отбора 4 случайных записей, на которых будет оцениваться точность работы классификаторов, также на рисунке 3.29 показаны метки классов отобранных записей. На рисунке 3.30 показан результат кластеризации исходных данных, в т.ч. выводится распределение записей по кластерам. Далее выводится статистическая информация о распределении классов по кластерам (рисунок 3.31)

Можно указать количество кластеров

n_cluster: 4

```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
       n_clusters=4, n_init=10, n_jobs=None, precompute_distances='auto',
       random_state=None, tol=0.0001, verbose=0)
```

В данном блоке можно вывести координаты центров кластеров

```
array([[0.00000000e+00, 7.48995984e-01, 1.00000000e+00, 6.76056338e-01,
        7.41935484e-01, 7.37704918e-01, 7.74222222e-01, 7.99096138e-01,
```

Рисунок 3.30 – Запуск и результаты работы алгоритма K-means

В данном блоке группируем данные и

Cluster	Class	Class
0	amdahl	2
1	amdahl	6
	burroughs	1
	cdc	5
2	burroughs	6
	cdc	3
3	c.r.d	5

Рисунок 3.31 – Группировка данных

Объекту присваивается метка класса ближайшего к нему центра кластера

	Ближайший центр кластера	Расстояние	Истинная метка класса
0	1	0.594247	amdahl
1	3	0.553568	c.r.d
2	2	0.561665	burroughs
3	1	0.827276	cdc

Рисунок 3.32 – Результаты классификации

По результатам классификации видно, что классификатор ошибся один раз при определении метки класса для четвертого объекта (рисунок 3.32).

Для оценки зависимости точности работы классификатора построим. таблицу с данными о том, как меняется точность работы классификаторы в зависимости от значения количества кластеров.

Таблица 3.2 – Зависимость точность работы классификатора в зависимости от количества k кластеров (набор данных – «Machine»)

k	2	4	6	8	10
Точность, %	20%	60%	60%	80%	100%

В данной выборке количество уникальных классов равно 4. Из приведенных в таблице 3.2 результатов можно сделать выводы, что результаты работы классификатора на данной выборке сильно зависят от количества кластеров. Вполне ожидаемо, что, если количество кластеров будет меньше количества уникальных классов, точность работы классификатора становится никуда не годной. Из 5 записей было верно распознано только 2, которые имели метки классов в классификаторах. При росте количества кластеров результаты сильно улучшаются. Как минимум можно сделать выводы, что количество кластеров должно быть не меньше количество классов выборки, а желательно и больше, примерно раза в 2.

Из чего следует, что при работе с данным классификатором значение количества кластеров следует устанавливать экспериментальным путем, постепенно увеличивая, до тех пор, пока не будет достигнута заданная точность.

3.4 Вычислительный эксперимент на наборе данных «DryBean»

Рассмотрим набор данных DryBean. Это набор компьютерных данных, из 16 числовых атрибутов и одного категориального, который будем принимать за метку класса. Состоит из 150 записей и содержит 17 атрибутов. Приведем смысл значения каждого атрибута и описание датасета:

Описание: В этом исследовании было использовано несколько различных типов сухих бобов с учетом таких характеристик, как форма, размеры, тип и структура, в зависимости от рыночной ситуации. Система

компьютерного зрения была разработана для различения нескольких зарегистрированных сортов сухих бобов со схожими характеристиками с целью получения единой классификации семян. Для модели классификации изображения 150 зерен 4 различных зарегистрированных сухих бобов были сделаны с помощью камеры высокого разрешения. Изображения фасоли, полученные с помощью системы компьютерного зрения, были подвергнуты этапам сегментации и выделения признаков, и в общей сложности было обнаружено 16 признаков.

Из зерен были получены 12 размеров и 4 комплексных параметра, описывающих форму.

1. Area (A): Площадь зоны bean-компонента и количество пикселей в ее границах.

2. Периметр (P): Окружность фасоли определяется как длина ее границы.

3. Длина главной оси (L): расстояние между концами самой длинной линии, которую можно провести от фасоли.

4. Длина вспомогательной оси (l): самая длинная линия, которую можно провести от фасоли, стоя перпендикулярно главной оси.

5. Соотношение сторон (K): определяет соотношение между L и l.

6. Эксцентриситет (Ec): Эксцентриситет эллипса, имеющий те же моменты, что и область.

7. Выпуклая область (C): количество пикселей в наименьшем выпуклом многоугольнике, которое может содержать область семени компонента.

8. Эквивалентный диаметр (Ed): диаметр круга, имеющий такую же площадь, что и площадь семян фасоли.

9. Экстент (Ex): отношение пикселей в ограничивающей рамке к области bean-компонента.

10. Плотность (S): также известна как выпуклость. Отношение пикселей выпуклой оболочки к пикселям фасоли.

11. Округлость (R): рассчитывается по следующей формуле: (4piA) / (P ^ 2).

12. Компактность (CO): измерение округлости объекта: Ed / L.

13. ShapeFactor1 (SF1) – первый параметр геометрической формы боба.

14. ShapeFactor2 (SF2) – второй параметр формы боба.

15. ShapeFactor3 (SF3) – третий параметр формы боба.

16. ShapeFactor4 (SF4) – четвертый параметр формы боба.

17. Класс.

Далее на рисунках 3.33-3.39 приведено как меняется выборка в процессе прохождения через классификацию, а также полученные и модифицированные наборы данных и результат классификации.

Исходя из предыдущих результатов, для достижения наилучшего результата установим количество кластеров равное – 4, размер тестовой выборки – 5. Значение коэффициента kc установим равным 0.5.

Первые 5 значений выборки

	0	1	2	3	4	5	6	7	8	9	10
0	47224	805.648	300.922462	200.781355	1.498757	0.744861	47747	245.209022	0.805760	0.989046	0.914286
1	59563	961.018	390.245047	195.973442	1.991316	0.864763	60387	275.386940	0.785792	0.986355	0.810445
2	88489	1188.149	467.295713	244.070996	1.914589	0.852759	90923	335.660087	0.735056	0.973230	0.787694
3	41623	766.942	295.073232	179.917999	1.640043	0.792601	42054	230.208709	0.782387	0.989751	0.889238
4	41626	760.147	283.751181	187.117650	1.516432	0.751755	42103	230.217005	0.774149	0.988671	0.905273

Рисунок 3.33 – Загрузка исходных данных

Первые 5 значений датафрейма, получившегося в результате преобразовния данных

	0	1	2	3	4	5	6	7	8	9	10
0	0.115671	0.136428	0.188818	0.263774	0.219136	0.455209	0.115104	0.138392	0.914812	0.884519	0.910634
1	0.370493	0.487207	0.595395	0.213895	0.680793	0.867277	0.369016	0.416813	0.838376	0.811063	0.629571
2	0.967866	1.000000	0.946113	0.712874	0.608880	0.826022	0.982423	0.972892	0.644172	0.452898	0.567992
3	0.000000	0.049042	0.162194	0.047331	0.351558	0.619278	0.000743	0.000000	0.825345	0.903754	0.842839
4	0.000062	0.033701	0.110659	0.122022	0.235702	0.478900	0.001728	0.000077	0.793812	0.874264	0.886238

Рисунок 3.34 – Преобразованные данные выборки

split: 5

	0	1	2	3	4	5	6	7	8	9	10
0	0.115671	0.136428	0.188818	0.263774	0.219136	0.455209	0.115104	0.138392	0.914812	0.884519	0.910634
1	0.370493	0.487207	0.595395	0.213895	0.680793	0.867277	0.369016	0.416813	0.838376	0.811063	0.629571
2	0.967866	1.000000	0.946113	0.712874	0.608880	0.826022	0.982423	0.972892	0.644172	0.452898	0.567992
3	0.000000	0.049042	0.162194	0.047331	0.351558	0.619278	0.000743	0.000000	0.825345	0.903754	0.842839
4	0.000062	0.033701	0.110659	0.122022	0.235702	0.478900	0.001728	0.000077	0.793812	0.874264	0.886238

Рисунок 3.35 – Тестовая выборка

На рисунке 3.33 показан результат загрузки исходной выборки данных. На рисунке 3.34 показан результат нормировки входных параметров к диапазону от 0 до 1. На рисунке 3.35 показан результат отбора 5 случайных записей, на которых будет оцениваться точность работы классификаторов, также на рисунке 3.36 показаны метки классов отобранных записей. На рисунке 3.37 показан результат кластеризации исходных данных, в т.ч. выводится распределение записей по кластерам. Далее выводится статистическая информация о распределении классов по кластерам (рисунок 3.39)

```
0         SIRA
1        HOROZ
2         CALI
3     DERMASON
4     DERMASON
Name: 16, dtype: object
```

Рисунок 3.36 – Истинные метки класса тестовой выборки

```
n_cluster: 10

KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
       n_clusters=10, n_init=10, n_jobs=None, precompute_distances='auto',
       random_state=None, tol=0.0001, verbose=0)
```

В данном блоке можно вывести координаты центров кластеров

```
array([[3.79018490e-01, 5.11711997e-01, 5.51752491e-01, 2.92549259e-01,
        5.69470063e-01, 7.96038152e-01, 3.83230215e-01, 4.25600468e-01,
        4.15700564e-01, 6.84942138e-01, 5.97491183e-01, 3.12066875e-01,
        6.36522479e-01, 1.86372197e-01, 2.80077819e-01, 5.61625066e-01],
```

Рисунок 3.37–Запуск и результаты работы алгоритма K-means

Объекту присваивается метка класса ближайшего к нему центра кластера

	Ближайший центр кластера	Расстояние	Истинная метка класса
0	1	0.168572	SIRA
1	3	0.112202	HOROZ
2	9	0.383069	CALI
3	7	0.193470	DERMASON
4	8	0.187258	DERMASON

Рисунок 3.38 – Результаты классификации

		Class
Cluster	Class	
0	HOROZ	12
1	SIRA	19
2	CALI	30
3	HOROZ	18
4	SIRA	30
5	HOROZ	17
6	HOROZ	2
7	DERMASON	24
8	DERMASON	23
9	CALI	19

Рисунок 3.39 – Группировка данных

По результатам классификации видно (рисунок 3.38), что точность составила 100%. Таким образом, если задавать количество кластеров равное в 1.5-2.5 раза больше количества классов в обучающей выборке, то можно добиться максимальной точности работы классификатора на большинстве выборок исходных данных.

Таким образом, точность получаемых классификаторов, основанных на предложенных в исследовании подходах, сопоставима с точностью стандартных классификаторов – k-nearest neighbors, Decision Tree, Random Forest.

Выводы по разделу

По результатом приведенных исследований были сделаны следующий выводы:

- На языке Python разработано программное обеспечение, реализующее: загрузку данных из указанного файла, построение классификатора данных на основе алгоритма k-means и оценку точности работы классификаторов.
- Тестирование предложенной технологии синтеза классификатора на основе результатов кластерного анализа проводилось на трех различных выборках данных из репозитория UCI Machine Learning Repository. Результаты тестирования показали, что точность работы получаемых классификаторов, основанных на предложенных в исследовании подходах, сопоставима с точностью стандартных классификаторов – k-nearest neighbors, Decision Tree, Random Forest.
- Установлено, что если задавать количество кластеров равное в 1.5-2.5 раза больше количества классов в обучающей выборке, то можно добиться максимальной точности работы классификатора на большинстве выборок исходных данных

Заключение

В заключении приведем основные выводы по проделанной работе:

1. Анализ литературных источников по теме исследования показал, что алгоритмы машинного обучения разрабатывались для решения универсальных типов задач (классификации, регрессии, кластеризации, аффинитивного анализа, оптимизации, поиска аномалий).

2. Анализ научных статей показал, что в настоящее время сформировались две тенденции развития алгоритмов машинного обучения: увеличение доли участия алгоритмов в различных этапах анализе данных (рисунок 1.12) и разработка способов по расширению перечня типов задач, решаемых существующими алгоритмами машинного обучения (рисунок 1.14)

3. Показана связь, между типом решаемых задач и алгоритмами машинного обучения (рисунок 1.13). Также для каждого типа задачи приведены практические примеры использования алгоритмов.

4. Предложена технология синтеза классификаторов данных на основе результатов кластерного анализа (рисунок 2.8). В соответствии с этой технологией проводится кластеризация данных (без учета значений меток классов) с использование алгоритма k-means. В результате кластеризации исследуемые объекты распределяются по группам (кластерам) и рассчитываются центры кластеров. Затем проводиться статистический анализ каждого кластера для определения преобладающего в нем класса. Классификатор включает в себя параметры центров кластеров, а также для каждого кластера – метку преобладающего класса (рисунок 2.9). При классификации исследуемого объекта определяется его принадлежность к одному из кластеров путем расчёта расстояния от объекта до центра кластеров. Считается, что исследуемый объект относится к тому кластеру, расстояние, до центра которого наименьшее. Исследуемому объекту присваивается метка класса, преобладающего в данном кластере.

5. На языке Python разработано программное обеспечение, реализующее: загрузку данных из указанного файла, построение классификатора данных на основе алгоритма k-means и оценку точности работы классификаторов.

6. Тестирование предложенной технологии синтеза классификатора на основе результатов кластерного анализа проводилось на трех различных выборках данных из репозитория UCI Machine Learning Repository. Результаты тестирования показали, что точность работы получаемых классификаторов, основанных на предложенных в исследовании подходах, сопоставима с точностью стандартных классификаторов – k-nearest neighbors, Decision Tree, Random Forest.

7. Установлено, что если задавать количество кластеров равное в 1.5-2.5 раза больше количества классов в обучающей выборке, то можно добиться максимальной точности работы классификатора на большинстве выборок исходных данных.

На основании всего вышеизложенного можно сделать вывод, что цель исследования достигнута.

Список используемой литературы

1. Алифбекова, Н.Р. Сравнительный анализ алгоритмов распознавания человеческого лица / Н.Р. Алифбекова, А.В. Рытов // Сборник статей Всероссийской студенческой научно-практической междисциплинарной конференции «Молодежь. Наука. Общество». 2020. – Тольяттинский государственный университет, 2020. – с. 38-41. – Текст : непосредственный.

2. Власов, А.В. Машинное обучение применительно к задаче классификации семян зерновых культур в видеопотоке / А.В. Власов, А.С. Федеев // Молодежь и современные информационные технологии – сборник трудов XIV Международной научно-практической конференции студентов, аспирантов и молодых учёных, 07–11 ноября 2016. – Национальный исследовательский Томский политехнический университет (Томск), 2016. – с. 133-135. – Текст : непосредственный.

3. Клячин В.Н. Использование агрегированных классификаторов при технической диагностике на базе машинного обучения / В.Н. Клячин, Ю.Е. Кувайскова, Д.А. Жуков // Информационные технологии и нанотехнологии (ИТНТ-2017) – сборник трудов III международной конференции и молодежной школы. Самарский национальный исследовательский университет имени академика С.П. Королева. 2017. – Предприятие "Новая техника" (Самара), 2017. – с. 1770-1773. – Текст : непосредственный.

4. Кононова, Н.В. Исследование подсистемы контентной фильтрации с использованием методов машинного обучения / Н.В. Кононова, Ю.А. Андрусенко, Т.А. Самокаева // Студенческая наука для развития информационного общества – сборник материалов VI Всероссийской научно-технической конференции. 22–26 мая 2017. – Северо-Кавказский федеральный университет (Ставрополь), 2017. – с. 268-270. – Текст : непосредственный.

5. Мелдебай, М.А. Анализ мнений покупателей на основе машинного обучения / М.А. Мелдебай, А.К. Сарбасова // Прикладная математика и информатика: современные исследования в области естественных и технических наук – материалы III научно-практической всероссийской конференции (школы-семинара) молодых ученых. 24–25 апреля 2017 года. – Издатель Качалин Александр Васильевич, 2017. – с. 360-363. – Текст : непосредственный.

6. Наумов, Д.П. Регулятор САР на основе машинного обучения / Д.П. Наумов, Д.П. Стариков // Информационные технологии в управлении, автоматизации и мехатронике – сборник научных трудов Международной научно-технической конференции. 06–07 апреля 2017 года. – ЗАО "Университетская книга" (Курск), 2017. – с. 106-114. – Текст : непосредственный.

7. Осколков, В.М. Использование метода машинного обучения для повышения продуктивности на предприятии / В.М. Осколков, Н.И. Шаханов, И.А. Варфоломеев, О.В. Юдина, Е.В. Ершов // Автоматизация и энергосбережение машиностроительного и металлургического производств, технология и надежность машин, приборов и оборудования – материалы XII Международной научно-технической конференции, 21 марта 2017. – Вологодский государственный университет (Вологда), 2017. – с. 177-180. – Текст : непосредственный.

8. Осколков, В.М. Применение параллельных вычислений для прогнозирования на основе алгоритма машинного обучения Random Forest / В.М. Осколков, Н.И. Шаханов, И.А. Варфоломеев, О.В. Юдина, Л.Н. Виноградова, Е.В. Ершов // Сборник трудов конференции Оптико-электронные приборы и устройства в системах распознавания образов, обработки изображений и символьной информации. Распознавание, Курск, 16–19 мая 2017 года. – Юго-Западный государственный университет (Курск), 2017. – с. 267-269. – Текст : непосредственный.

9. Соловьев, А.Ю. Применение машинного обучения для прогнозирования неблагоприятных исходов в ургентной хирургии / Соловьев А.Ю., Берегов М.М., Вахеева Ю.М., Баутин А.Н., Гусев А.В. // Медико-биологические, клинические и социальные вопросы здоровья и патологии человека – материалы III Всероссийской образовательно-научной конференции студентов и молодых ученых с международным участием в рамках XIII областного фестиваля "Молодые ученые - развитию Ивановской области". 2017. – Ивановская государственная медицинская академия (Иваново), 2017. – с. 129-130. – Текст : непосредственный.

10. Ткач, Т.Ч. Машинное обучение и обработка больших данных - обучение в основной и средней школе / Т.Ч. Ткач // Актуальные проблемы методики обучения информатике и математике в современной школе – материалы международной научно-практической интернет-конференции. Московский педагогический государственный университет, Москва, 24 апреля 2020 года. – Московский педагогический государственный университет (Москва), 2020. – с. 217-223. – Текст : непосредственный.

11. Федотов, И.А. Применение технологий машинного обучения для прогнозирования ситуации на финансовых рынках / И.А. Федотов // Студенческая наука для развития информационного общества – сборник материалов VI Всероссийской научно-технической конференции. 22–26 мая 2017. – Северо-Кавказский федеральный университет (Ставрополь), 2017. – с. 361-363. – Текст : непосредственный.

12. Якимчук, А.А. Глубокое обучение как эффективный метод машинного обучения / А.А. Якимчук // Научное сообщество студентов XXI столетия. Технические науки – сборник статей по материалам XCII студенческой международной научно-практической конференции. 2020. – ООО “Сибирская академическая книга” (Новосибирск), 2020. – с. 40-43. – Текст : непосредственный.

13. Filipczuk P. Automatic Breast Cancer Diagnosis Based on K-Means Clustering and Adaptive Thresholding Hybrid Segmentation [Text] / Paweł

Filipczuk, Marek Kowal, Andrzej Obuchowicz // Image Processing and Communications Challenges 3 – Advances in Intelligent and Soft Computing – Springer-Verlag Berlin Heidelberg 2011. – pp. 295-302. – Текст : непосредственный.

14. Fu L. A Robust Text Segmentation Approach in Complex Background Based on Multiple Constraints [Text] / Libo Fu, Weiqiang Wang, Yaowen Zhan // Pacific-Rim Conference on Multimedia – 6th Pacific Rim Conference on Multimedia, Jeju Island, Korea, November 13-16, 2005, Proceedings, Part I: Advances in Multimedia Information Processing - PCM 2005. – Springer-Verlag Berlin Heidelberg 2005. – pp. 594-605. – Текст : непосредственный.

15. Ghosh S. Aggregation Pheromone Density Based Image Segmentation [Text] / Susmita Ghosh, Megha Kothari, Ashish Ghosh // 5th Indian Conference, ICVGIP 2006, Madurai, India, December 13-16, 2006. Proceedings – Computer Vision, Graphics and Image Processing. – Springer-Verlag Berlin Heidelberg 2006. – pp. 118-127. – Текст : непосредственный.

16. Gllavata J. Adaptive Fuzzy Text Segmentation in Images with Complex Backgrounds Using Color and Texture [Text] / Julinda Gllavata, Bernd Freisleben // International Conference on Computer Analysis of Images and Patterns – 11th International Conference, CAIP 2005, Versailles, France, September 5-8, 2005. –Proceedings: Computer Analysis of Images and Patterns. – Springer-Verlag Berlin Heidelberg 2005. – pp. 756-765. – Текст : непосредственный.

17. Gou Sh. Spectral Clustering Based on Dictionary Learning Sampling for Image Segmentation [Text] / Shuiping Gou, Jingyu Yang, Tiantian Yu // International Conference on Intelligent Science and Intelligent Data Engineering. Second Sino-foreign-interchange Workshop, IScIDE 2011, Xi'an, China, October 23-25, 2011, Revised Selected Papers – IScIDE 2011: Intelligent Science and Intelligent Data Engineering. – Springer-Verlag Berlin Heidelberg 2012. – pp. 334-340. – Текст : непосредственный.

18. Huang Z. Chinese Historic Image Threshold Using Adaptive K-means Cluster and Bradley's [Text] / Zhi-Kai Huang, Yong-Li Ma, Li Lu, Fan-Xing Rao, Ling-Ying Hou // 12th International Conference, ICIC 2016, Lanzhou, China, August 2-5, 2016, Proceedings, Part III – ICIC 2016: Intelligent Computing Methodologies. – Springer International Publishing Switzerland 2016. – pp. 171-179. – Текст : непосредственный.

19. Jiang Y. Fast and Effective Image Segmentation via Superpixels and Adaptive Thresholding [Text] // Yunsheng Jiang, Jinwen Ma // International Symposium on Neural Networks – 11th International Symposium on Neural Networks, ISNN 2014, Hong Kong and Macao, China, November 28- December 1, 2014. Proceedings: Advances in Neural Networks – ISNN 2014. – Springer International Publishing Switzerland 2014. – pp. 568-575. – Текст : непосредственный.

20. Klepaczko A. Automated Segmentation of Endoscopic Images Based on Local Shape-Adaptive Filtering and Color Descriptors [Text] / Artur Klepaczko, Piotr Szczypiński // International Conference on Advanced Concepts for Intelligent Vision Systems, ACIVS 2010, Sydney, Australia, December 13-16, 2010, Proceedings, Part I – ACIVS 2010: Advanced Concepts for Intelligent Vision Systems. – Springer-Verlag Berlin Heidelberg 2010. – pp. 245-254. – Текст : непосредственный.

21. Leydier Y. Serialized k-Means for Adaptative Color Image Segmentation [Text] / Yann Leydier, Frank Le Bourgeois, Hubert Emptoz // 6th International Workshop, DAS 2004, Florence, Italy, September 8 - 10, 2004. Proceedings – DAS 2004: Document Analysis Systems VI. – Springer-Verlag Berlin Heidelberg 2004. – pp. 252-263. – Текст : непосредственный.

22. Li T. K-Means Optimization Algorithm Based on Tightness Mutation [Text] / Tie Fei Li, Jian Fei Ma, Rui Xin Yang, Di Wu, Yan Guang Shen // International Conference on Geo-informatics in Sustainable Ecosystem and Society – 6th International Conference, GSES 2018, Handan, China, September 25–26, 2018, Revised Selected Papers: Geo-informatics in Sustainable Ecosystem

and Society. – Springer Nature Singapore Pte Ltd. 2019. – pp. 146-156. – Текст : непосредственный.

23. Ouadfel S. Unsupervised Image Segmentation Using a Colony of Cooperating Ants [Text] / Salima Ouadfel, Mohamed Batouche // International Workshop on Biologically Motivated Computer Vision – Second International Workshop, BMCV 2002 Tübingen, Germany, November 22–24, 2002 Proceedings: Biologically Motivated Computer Vision. – Springer-Verlag Berlin Heidelberg 2002. – pp. 109-116. – Текст : непосредственный.

24. Pachai Ch. Unsupervised and Adaptive Segmentation of Multispectral 3D Magnetic Resonance Images of Human Brain: A Generic Approach [Text] / Chahin Pachai, Yue Min Zhu, Charles R. G. Guttmann, Ron Kikinis, Ferenc A. Jolesz, Gérard Gimenez, Jean-Claude Froment, Christian Confavreux, Simon K. Warfield // International Conference on Medical Image Computing and Computer-Assisted Intervention – 4th International Conference Utrecht, The Netherlands, October 14–17, 2001. Proceedings: Medical Image Computing and Computer-Assisted Intervention – MICCAI 2001. – Springer-Verlag Berlin Heidelberg 2001. – pp. 1067-1074. – Текст : непосредственный.

25. Shi R. An Adaptive Image Content Representation and Segmentation Approach to Automatic Image Annotation [Text] / Rui Shi, Huamin Feng, Tat-Seng Chua, Chin-Hui Lee // International Conference on Image and Video Retrieval – Third International Conference, CIVR 2004, Dublin, Ireland, July 21-23, 2004. Proceedings: Image and Video Retrieval. – Springer-Verlag Berlin Heidelberg 2004. – pp. 545-554. – Текст : непосредственный.

26. Tian Y. An Adaptive Threshold Algorithm for Moving Object Segmentation [Text] / Yumin Tian, Dan Wang, Risan Lin, Qichao Chen // CCF Chinese Conference, CCCV 2015, Xi'an, China, September 18-20, 2015, Proceedings, Part I – CCCV 2015: Computer Vision. – Springer-Verlag Berlin Heidelberg 2015. – pp. 230-239. – Текст : непосредственный.

27. Vemuri B. Multiresolution adaptive K-means algorithm for segmentation of brain MRI [Text] / B. C. Vemuri, S. Rahman, J. Li // Third

International Computer Science Conference – ICSC '95 Hong Kong, December 11–13, 1995 Proceedings. ICSC 1995: Image Analysis Applications and Computer Graphics. – Springer, Berlin, Heidelberg. – pp. 347-354. – Текст : непосредственный.

28. Wu Y. Multi-scale Medical Image Segmentation Based on Salient Region Detection [Text] / Yingxue Wu, Xi Zhao, Guiyang Xie, Yangkexin Liang, Wei Wang, Yue Li // Chinese Conference on Biometric Recognition 10th Chinese Conference, CCBR 2015, Tianjin, China, November 13-15, 2015, Proceedings – CCBR 2015: Biometric Recognition. – Springer International Publishing Switzerland 2015. – pp. 624-632. – Текст : непосредственный.

29. Xin D. SOR Based Fuzzy K-Means Clustering Algorithm for Classification of Remotely Sensed Images [Text] / Dong-jun Xin, Yen-Wei Chen // 10th International Symposium on Neural Networks, Dalian, China, July 4-6, 2013, Proceedings, Part I – ISNN 2013: Advances in Neural Networks – ISNN 2013. – Springer-Verlag Berlin Heidelberg 2013. – pp. 375-382. – Текст : непосредственный.

30. Xue X. Detection of Overlapped Apples in Orchard Scene Using Improved K-means and Distance Least Square [Text] / Xia Xue, Zhou Guomin, Qiu Yun, Li Zhuang, Wang Jian, Hu Lin, Fan Jingchao, Guo Xiuming // International Conference on Computer and Computing Technologies in Agriculture – 11th IFIP WG 5.14 International Conference, CCTA 2017, Jilin, China, August 12-15, 2017, Proceedings, Part I: Computer and Computing Technologies in Agriculture XI. – IFIP International Federation for Information Processing 2019. – pp. 269-284. – Текст : непосредственный.

31. Zhang M. A Customer Segmentation Model Based on Affinity Propagation Algorithm and Improved Genetic K-Means Algorithm [Text] / Meiyang Zhang, Zili Zhang, Shi Qiu // 10th IFIP TC 12 International Conference, IIP 2018, Nanning, China, October 19-22, 2018, Proceedings – IIP 2018: Intelligent Information Processing IX. – IFIP International Federation for Information Processing 2018. – pp. 321-327. – Текст : непосредственный.