Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ   
СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра АОИ

**Метод нечеткой кластеризации Густафсона-Кесселя. Генерация наименований нечетких кластеров.**

**ОТЧЕТ**

ПО РЕЗУЛЬТАТАМ

производственной практики: научно-исследовательская работа

|  |  |
| --- | --- |
|  | Обучающийся гр. 428-2  \_\_\_\_\_\_\_\_\_\_\_\_\_Даниил Александрович Челпанов  (подпись) (И.О. Фамилия)  3.09.2020  (дата) |
| **\_\_\_\_\_\_\_\_\_**  (оценка)  М.П. | Руководитель практики от профильной организации:  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (должность, ученая степень, звание)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) (И.О. Фамилия)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (дата) |
| **\_\_\_\_\_\_\_\_\_**  (оценка) | Руководитель практики от университета:  доцент, к.т.н., доцент  (должность, ученая степень, звание)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ М.П. Силич  (подпись) (И.О. Фамилия)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (дата) |

Томск 2020

# Реферат

Научно-исследовательская работа, 30 стр., 8 рис., 5 источников, 3 прил.

[НЕЧЕТКАЯ КЛАСТЕРИЗАЦИЯ](https://elibrary.ru/keyword_items.asp?id=2621252), [НАИМЕНОВАНИЯ КЛАСТЕРОВ](https://elibrary.ru/keyword_items.asp?id=16643487), [ГЕНЕРИРОВАНИЕ ТЕРМОВ](https://elibrary.ru/keyword_items.asp?id=9760828), [ЗОНИРОВАНИЕ ТЕРРИТОРИЙ](https://elibrary.ru/keyword_items.asp?id=2654132), ГУСТАФСОН-КЕССЕЛ

Объектом исследования являются метод нечеткого кластерного анализа Густафсона-Киселя, кластеризации субъектов Российской Федерации и метод генерации наименований нечетких кластеров.

Цель работы – программная реализация метода генерации наименований нечетких кластеров и применение программного обеспеченья для генерации наименований кластеров, образованных из данных по субъектам Российской Федерации.

В ходе тестирования разработанного программного обеспеченья на реальных данных были сформированы две группы из двух кластеров, с разными выборочными данными, элементами входящими в них и четырех сгенерированных разными способами наименований. Группы и исследования по ним отражены на рисунках и диаграммах.

Метод генерации наименований экономически эффективен для быстрой и точной обработки больших объёмов информации и группировки их по признакам.

# Задание

***Статистические данные:***

Устаревание 2012 (Степень износа основных фондов (%) (по субъектам Российской Федерации за 2012 год)) – x*-*признак (фактор).

Инвестиции 2012 (Инвестиции в основной капитал на душу населения (руб.) (по субъектам Российской Федерации за 2012 год)) – y*-*признак (результат).

Прибыль 2016 (Рентабельность продаж товаров и услуг (%) (по субъектам Российской Федерации за 2016 год)) – x*-*признак (фактор).

Энергия 2016 (Потребление электроэнергии на душу населения (млн. КВт\*час/чел.) (по субъектам Российской Федерации за 2016 год)) – y*-*признак (результат).

***Задание:***

1. Изучить метод Густафсона-Кесселя.
2. Изучить теорию генерации наименований нечетких кластеров в пространстве разнородных признаков.
3. Разработать модуль для генерации наименований нечетких кластеров в пространстве разнородных признаков для зонирования территориальных объектов по внешним условиям.
4. Применить разработанный модуль для генерации наименований кластеров, сформированных на реальных данных.

# Оглавление

[1 Введение 4](#_Toc71012943)

[2 Автоматическая генерация наименований нечетких кластеров 6](#_Toc71012944)

[2.1 Нечеткая кластеризация методом Густафсона-Кесселя 6](#_Toc71012945)

[2.2 Метод генерации наименований кластеров 8](#_Toc71012946)

[3 Программная реализация метода генерации наименований нечетких кластеров 11](#_Toc71012947)

[3.1. Разработка программного обеспечения для генерации наименований нечетких кластеров 11](#_Toc71012948)

[3.2. Применение программного обеспечения для генерации наименований кластеров на реальных данных 12](#_Toc71012949)

[3 Заключение 17](#_Toc71012950)

[4 Терминология 18](#_Toc71012951)

[5 Список использованных источников 19](#_Toc71012952)

[Приложение А Программная реализация алгоритма Густафсона-Кесселя 20](#_Toc71012953)

[Приложение Б Программная реализация метода для генерации наименований нечетких кластеров в пространстве разнородных признаков 22](#_Toc71012955)

[Приложение В Исходные данные для кластеризации субъектов РФ 27](#_Toc71012957)

# 1 Введение

**Цель работы**: Программная реализация метода генерации наименований нечетких кластеров и применение разработанного модуля для генерации наименований нечетких кластеров, образованных из имеющихся данных по методу Густафсона-Кесселя.

**Область исследования**: Наука и инновации, искусственный интеллект, добыча данных (Data Mining).

Нечеткая кластеризация, позволяющая группировать объекты в классы с размытыми границами, может применяться для решения различных задач. Рассмотрим ее применение для задачи оценки территориальных объектов по уровню развития с учетом неоднородности внешних условий.

Сравнительный анализ уровня развития некоторой сферы общественной жизни территориальных объектов (субъектов РФ, муниципальных образований, поселений, отдельных зданий, сооружений) позволяет классифицировать территории по уровням либо расставить их по рейтингу. Как правило, используется индикативный подход, при котором оценки (в виде рангов, баллов или лингвистических значений) выводятся на основе значений индикаторов для сравниваемых территориальных объектов (ТО). Полученные оценки являются основой для принятия решений по исправлению ситуации в проблемных регионах.

Однако неоднородность природно-климатических, социально-экономических, инфраструктурных и других внешних факторов снижает адекватность оценок, поскольку регионы изначально находятся в различных условиях. В этой ситуации помогает зонирование территорий по внешним факторам с помощью нечеткой кластеризации.

Пользователю предъявляются результаты в виде степеней принадлежности объектов кластеризации нечетким кластерам. В качестве идентификаторов кластеров, как правило, используются порядковые номера. Однако желательно, чтобы наименования кластеров несли смысловую нагрузку и были выражены в терминах естественного языка. Это облегчает пользователю интерпретацию результатов зонирования территорий на семантическом уровне. Основной сложностью является то, что заранее не известно каково будет расположение кластеров. Их количество также может варьироваться в ходе итерационного поиска наилучшего разбиения, и сколько в итоге будет получено кластеров – не известно. Даже после получения результатов кластеризации подбор наименований может вызывать затруднения, особенно в случае множества индикаторов, поскольку визуализировать кластеры в многомерном пространстве не всегда возможно.

Задача подбора наименований кластеров также может рассматриваться как задача формирования термов лингвистической переменной. Кластеры при этом представляют собой дискретно заданные многомерные функции принадлежности.

.

**Задачи работы**: исследовать метод нечеткой кластеризации Густафсона-Кесселя и метод генерации наименований нечетких кластеров. Программная реализация метода генерации наименований нечетких кластеров и его применение на реальных данных.

# 2 Автоматическая генерация наименований нечетких кластеров

## 2.1 Нечеткая кластеризация методом Густафсона-Кесселя

**Кластеризация** (англ. *cluster analysis*) — задача группировки множества объектов на подмножества (**кластеры**) таким образом, чтобы объекты из одного кластера были более похожи друг на друга, чем на объекты из других кластеров по какому-либо критерию. [1]

Существует множество методов кластеризации, которые можно классифицировать на четкие и нечеткие. Четкие методы кластеризации разбивают исходное множество объектов X на несколько непересекающихся подмножеств. При этом любой объект из X принадлежит только одному кластеру. Нечеткие методы кластеризации позволяют одному и тому же объекту принадлежать одновременно нескольким (или даже всем) кластерам, но с различной степенью. Нечеткая кластеризация во многих ситуациях более "естественна", чем четкая, например, для объектов, расположенных на границе кластеров. Самым распространённым методом является метод k-средних (с-means). [2]

В классическом алгоритме k-средних элементы выбираются с помощью обычного евклидового расстояния между вектором x и центром кластера c:

где  - операция транспонирования.

При таком задании расстояния между двумя векторами множество точек, равноудаленных от центра, принимает вид сферы с одинаковым масштабом по всем осям. Но если данные создают группы, форма которых отличается от сферической или если шкалы отдельных координат вектора сильно отличаются, в этом случае метрика становится неадекватной. В этом случае качество кластеризации можно значительно повысить за счет улучшенной версии алгоритма самоорганизации, который называется алгоритмом Густафсона-Кесселя. [3]

Основные изменения относительно базового алгоритма k-средних в ведении в формулу расчета метрики масштабирующей матрицы A. При таком масштабировании расстояние между центром c и векторами x определяется формулой:

где – адаптивная норма.

В качестве масштабирующей обычно используется положительно определённая матрица, то есть матрица, у которой все собственные числа действительные и положительные.

Аналогично базовому алгоритму k-средних цель обучения с использованием алгоритма Густафсона-Кесселя в таком размещении центров, при котором минимизируется критерий:

Где - экспоненциальный вес, определяющий нечеткость, размазанность кластеров. Обычно выбирают  =2.

Описание алгоритма Густафсона-Кесселя:

Провести начальное размещение центров в пространстве данных. Создать элементарную форму масштабирующей матрицы А.

Сформировать матрицу коэффициентов принадлежностей всех векторов к центрам по формуле:

Рассчитать новое размещение центров c в соответствии с формулой:

Сгенерировать для каждого вектора матрицу ковариаций:

Если последние изменения положений центров и матрицы ковариации достаточно малые по отношению к предыдущим значениям и не превышают заданной величины, то – завершить итерационный процесс. [4]

Данные методы обладают следующим недостатком – число кластеров k должно быть известным. Для устранения первого недостатка и решения задачи кластерного анализа при незаданном числе кластеров разработан подход, основанный на вычислении величины: ΔE(k)=E(k–1)–E(k), где E(k–1) и E(k) – оптимальное значение критерия для числа кластеров (k–1) и (k). [2]

Как только величина , где = (0.2/0.3) – величина заданного порога, процесс увеличения числа кластеров прекращается.

## 2.2 Метод генерации наименований кластеров

Рассмотрим постановку задачи нечеткой кластеризации территориальных объектов по признакам, характеризующим различные внешние факторы (природно-климатические, социальные, технологические и т.д.).

Имеется множество территориальных объектов *O* = {*ok*}, которые необходимо распределить по группам, содержащим объекты с приблизительно одинаковым состоянием некоторого внешнего фактора. Состояние фактора оценивается по множеству индикаторов *pj* – измеримых показателей, выступающих в роли признаков кластеризации. Например, для фактора, характеризующего погодные условия в некотором заданном году, индикаторами могут быть такие показатели как: «Среднегодовая температура воздуха», «Продолжительность отопительного периода», «Среднегодовая влажность воздуха», «Среднегодовая скорость ветра» и др. Каждый объект *ok* характеризуется своим набором значений индикаторов. Это могут быть данные статистики или синтетические показатели, вычисленные на основе данных статистики. [5]

Наименования кластеров должны формироваться в соответствии с шаблоном, который задает пользователь. Шаблон включает в себя произвольный текст с предполагаемыми наименованиями (термами) для каждого индикатора. Термы в шаблоне должны перечисляться в порядке увеличения показателя индикатора: первым указывается терм для низких значений индикатора, а последним для высоких значений индикатора.

Чтобы сгенерировать наименование кластера , необходимо определить множество термов по каждому из индикаторов *pj* и подставить их в шаблон. Терм по отдельному индикатору может либо совпадать с одним из базовых термов, либо быть составным, сгенерированным на основе базовых термов с использованием различных связок.

Обозначим множество базовых термов для индикатора *pj* через . Термы ранжированы в порядке возрастания свойства, измеряемого индикатором: . Общее количество базовых термов *M* не должно быть большим. В большинстве случаев целесообразно задать *M* = 3. [5]

Для каждого базового терма сопоставляется интервал значений индикатора . Весь диапазон значений индикатора разбивается на M интервалов. Причем границы интервалов могут перекрываться. Для двух соседних термов и , имеющих соответственно ранги *m* и *m*+1, границы зон должны удовлетворять следующим условиям:

Где , – нижние границы зон для термов и , , – верхние границы зон для соответствующих индикаторов. [5]

На рисунке 2.1 представлено пространство значений двух индикаторов разбитое на зоны, соответствующие двум базовым термам по индикатору и трем базовым термам по индикатору .

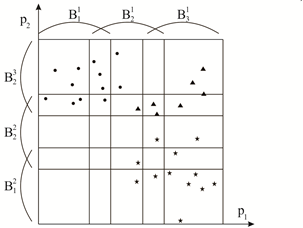


Рисунок 2.1 – Выделение зон, соответствующих базовым термам

Для каждого базового терма определим вес его зоны по формуле:

В зависимости от соотношения величин () определяется терм – наименование *l*-го кластера по *j*-му индикатору. В названии фигурируют те базовые термы, веса зон которых существенно больше весов остальных термов. Для любого названия зоны, вес его зоны либо является максимальным, либо отличается от максимального на некоторую пороговую величину (например, 0.1): , где – максимальный вес зоны ().

Если веса зон всех базовых термов, кроме одного, невелики (), то качестве наименования кластера используется базовый терм , вес зоны которого максимален: .

В ситуации, когда имеется несколько базовых термов, веса зон которых близки к максимальному весу, наименование терма формируется с помощью связки «и».

# 3 Программная реализация метода генерации наименований нечетких кластеров

## 3.1. Разработка программного обеспечения для генерации наименований нечетких кластеров

Для решения данной задачи было решено реализовать отдельный класс принимающий в себя алгоритм кластерного анализа и погрешность. Далее необходимо реализовать метод, принимающий в себя данные и шаблоны термов для индикаторов. Так как в теории, генерация наименований может проводится как по нормализованным данным, так и по исходным, в метод было решено добавить логическую переменную для определения необходимости нормализации данных или работы с исходными данными. Стоит отметить, что при работе с нормализованными данными количество базовых термов должно быть равно трем, так как в таком случае границы зон заданы заранее: [0, 0.4], [0.3, 0.7], [0.6, 1]. В случае же с исходными данными, количество термов не ограниченно, интервалы имеют четкие границы и максимальное значение данных принимается за верх интервала. На рисунке 3.1 представлена диаграмма деятельности метода.

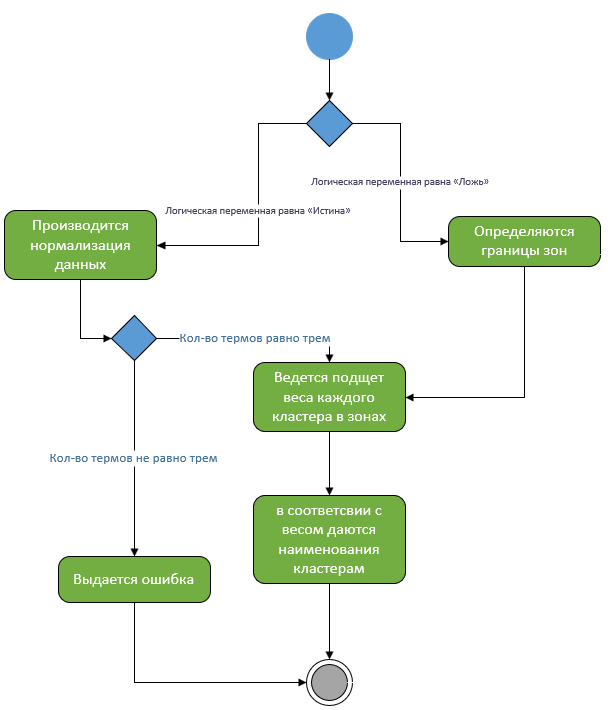


Рисунок 3.1 – Диаграмма деятельности

В приложении Б представлена программная реализация метода.

## 3.2. Применение программного обеспечения для генерации наименований кластеров на реальных данных

Для анализа даны данные энергетической отросли по регионам РФ представлены в приложении В.

В работе будет проведено два кластерных анализа по факторам Прибыль2016 – Энергия2016 и Устаривание2012 – Инвистиции2012. В последствии кластерам будут даны наименования в соответствии с методом, описанным выше.

Для начала, проведем кластерный анализ данных по фактору Прибыль2016 и результату Энергия2016 с помощью метода кластеризации Густафсона-Кесселя (Программная реализация представлена в приложении А).

На рисунке 3.2 можно увидеть результат.

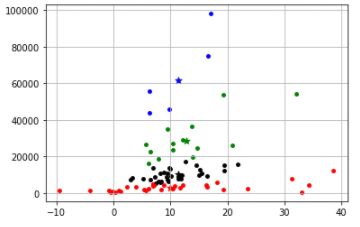


Рисунок 3.2 – Точечный график кластерного анализа по фактору Прибыль2016 и результату Энергия2016

Далее с помощью разработанного метода производим наименование полученных кластеров.

На рисунке 3.3 представлен результат выполнения наименования кластеров. На рисунке *u\_x* – чередуется вес зоны по *x* и вес зоны по *y.* Ниже же выводится массив наименований по индексу кластера. Как видно:

* Первый кластер (красный) получил наименования «Низкая и средняя рентабельность товаров и услуг и низкое потребление электроэнергии на душу населения».
* Второй кластер (зеленый) получил наименования «Средняя рентабельность товаров и услуг и низкое потребление электроэнергии на душу населения».
* Третий кластер (синий) получил наименования «Средняя рентабельность товаров и услуг и среднее потребление электроэнергии на душу населения».
* Четвертый кластер (черный) получил наименования «Средняя рентабельность товаров и услуг и низкое потребление электроэнергии на душу населения».

Если сравнить эти наименования с данными на рисунке 3.2 можно увидеть, большее скопление точек находится в середине графика по оси *x* и в нижней части по оси *y.*

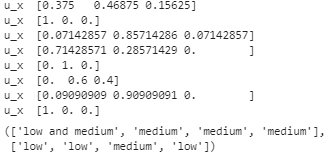


Рисунок 3.3 – Результат выполнения программы по фактору Прибыль2016 и результату Энергия2016 (исходные данные)

Теперь проведем такой же анализ, но уже с нормализацией данных внутри метода наименования кластеров. Для этого была обозначена логическая переменная обозначающая работу с нормализованными данными как «Истина». Нормализация данных происходит методом линейной нормализации. На рисунке 3.4 приведен результат наименования кластеров. Как видно, что с исходными данными, что с нормализованными, наименования одинаковые. Отличаются только некоторые веса зон.











Рисунок 3.4 – Результат выполнения программы по фактору Прибыль2016 и результату Энергия2016 (нормализованные данные)

Теперь сделаем все то же самое, но для кластеров, сформированных по фактору Устаривание2012 и результату Инвистиции2012. На рисунках 3.5-3.7 представлен конечный результат.

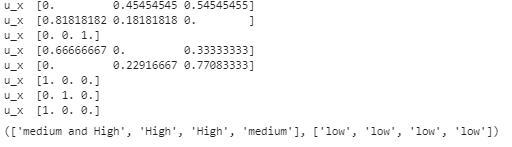


Рисунок 3.5 – Результат выполнения программы по фактору Устаривание2012 и результату Инвистиции2012 (исходные данные)

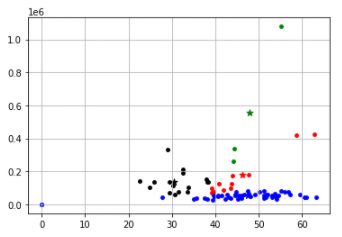


Рисунок 3.6 – Точечный график кластерного анализа по фактору Устаривание2012 и результату Инвистиции2012











Рисунок 3.7 – Результат выполнения программы по фактору Устаривание2012 и результату Инвистиции2012 (нормализованные данные)

Как видно, в этом случае результаты исходных данных и нормализованных отличаются. С не нормализованными данными:

* Первый кластер (красный) получил наименования «Средняя и высокая степень износа фондов и низкие инвестиции в основной капитал на душу населения».
* Второй кластер (зеленый) получил наименования «Высокая степень износа фондов и низкие инвестиции в основной капитал на душу населения».
* Третий кластер (синий) получил наименования «Высокая степень износа фондов и низкие инвестиции в основной капитал на душу населения».
* Четвертый кластер (черный) получил наименования «Средняя степень износа фондов и низкие инвестиции в основной капитал на душу населения».

С нормализованными данными:

* Первый кластер (красный) получил наименования «Высокая степень износа фондов и низкие инвестиции в основной капитал на душу населения».
* Второй кластер (зеленый) получил наименования «Высокая степень износа фондов и низкие инвестиции в основной капитал на душу населения».
* Третий кластер (синий) получил наименования «Высокая степень износа фондов и низкие инвестиции в основной капитал на душу населения».
* Четвертый кластер (черный) получил наименования «Средняя степень износа фондов и низкие инвестиции в основной капитал на душу населения».

# 3 Заключение

Сравнительный анализ территорий, имеющих существенные различия во внешних условиях, требует дифференцированного подхода, заключающегося в выполнении сравнения в рамках групп территориальных объектов с приблизительно похожими условиями. Использование методов нечеткой кластеризации позволяет выделять группы объектов на основании значений множества разнообразных признаков без использования субъективных суждений экспертов.

Для решения возникшей при этом проблемы подбора наименований кластеров, был реализован данный метод. С его помощью наименования формируются автоматическим путчем комбинирования базовых термов. Наименования подбираются в зависимости от расположения кластеров в пространстве значений индикаторов, причем, либо с учетом "размытости" границ, либо с четкими границами кластеров. Метод применим для случаев, когда в качестве признаков кластеризации выступают разнородные индикаторы, для семантической интерпретации значений которых используются разные наборы базовых термов.

В работе были испробованы два способа для наименований нечетких кластеров. Первый способ – способ с четкими границами. Этот способ хорош для применения, когда необходимо минимизировать не четкость и двойственные названия кластеров, а также когда наименований кластеров предполагается больше, чем три. С другой стороны, способ с нормализацией данных дает больше двойных имен, что дает больше пространства для анализа данных.

Если рассматривать данные, проанализированные в данной работе, можно сказать, что в большинстве регионов РФ средняя рентабельность товаров и услуг, предоставляемая тепло-энерго станциями и преимущественно низкое энергопотребление на 2016 год. С другой стороны, в 2012 году, были Высокие показатели устаревания фондов и низкие инвестиции в основной капитал.

# 4 Терминология

**Кла́стер** (англ. *Cluster* — скопление, кисть, рой) — объединение нескольких однородных элементов, которое может рассматриваться как самостоятельная единица, обладающая определёнными свойствами.

**Кластерный анализ** (англ. *cluster analysis*) — многомерная статистическая процедура, выполняющая сбор данных, содержащих информацию о выборке объектов, и затем упорядочивающая объекты в сравнительно однородные группы.

**Терм** — выражение [формального языка](https://ru.wikipedia.org/wiki/%D0%A4%D0%BE%D1%80%D0%BC%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) ([системы](https://ru.wikipedia.org/wiki/%D0%A4%D0%BE%D1%80%D0%BC%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0)) специального вида. По аналогии с [естественным языком](https://ru.wikipedia.org/wiki/%D0%95%D1%81%D1%82%D0%B5%D1%81%D1%82%D0%B2%D0%B5%D0%BD%D0%BD%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA), где [именная группа](https://ru.wikipedia.org/wiki/%D0%98%D0%BC%D0%B5%D0%BD%D0%BD%D0%B0%D1%8F_%D0%B3%D1%80%D1%83%D0%BF%D0%BF%D0%B0) ссылается на объект, а целое [предложение](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%B5%D0%B4%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5_(%D0%BB%D0%B8%D0%BD%D0%B3%D0%B2%D0%B8%D1%81%D1%82%D0%B8%D0%BA%D0%B0)) ссылается на [факт](https://ru.wikipedia.org/wiki/%D0%A4%D0%B0%D0%BA%D1%82)

# Список использованных источников

1. Университет ИТМО [Электронный ресурс] / Кластеризация, – Режим доступа: <http://neerc.ifmo.ru/wiki/> .
2. НГТУ НЭТИ Факультет прикладной математики и информатики [Электронный ресурс] / Лекция – Кластеризация данных при помощи нечетких отношений в Data Mining, – Режим доступа: <https://ami.nstu.ru> .
3. Fuzzy cluster analysis: method for classification, data analysis, and image recognition/ F.Hoeppner, F.Klawonn, R.Kruse, T.Runkler. – NY: John Wiley & Sons, Inc., 1999. – 300 p.
4. E. E. Gustafson and W. C. Kessel, “Fuzzy clustering with a fuzzy covariance matrix”, Proc. IEEE CDC, San-Diego, CA, – 1979 – pp. 761-766.
5. Силич М.П., Аксенов С.В. Генерация наименований нечетких кластеров в пространстве разнородных признаков для зонирования территориальных объектов по внешним условиям// Автоматизация и современные технологии. – 2018. – С. 81-86

# Приложение А

# Программная реализация алгоритма Густафсона-Кесселя

import numpy as np

from scipy.linalg import norm

class GK:

    def \_\_init\_\_(self, n\_clusters=4, max\_iter=100, m=2, error=1e-6):

        super().\_\_init\_\_()

        self.u, self.centers, self.f = None, None, None

        self.clusters\_count = n\_clusters

        self.max\_iter = max\_iter

        self.m = m

        self.error = error

    def fit(self, z):

        N = z.shape[0]

        C = self.clusters\_count

        centers = []

        u = np.random.dirichlet(np.ones(N), size=C)

        iteration = 0

        while iteration < self.max\_iter:

            u2 = u.copy()

            centers = self.next\_centers(z, u)

            f = self.\_covariance(z, centers, u)

            dist = self.\_distance(z, centers, f)

            u = self.next\_u(dist)

            iteration += 1

            # Stopping rule

            if norm(u - u2) < self.error:

                break

        self.f = f

        self.u = u

        self.centers = centers

        return centers

    def next\_centers(self, z, u):

        um = u \*\* self.m

        return ((um @ z).T / um.sum(axis=1)).T

    def \_covariance(self, z, v, u):

        um = u \*\* self.m

        denominator = um.sum(axis=1).reshape(-1, 1, 1)

        temp = np.expand\_dims(z.reshape(z.shape[0], 1, -1) - v.reshape(1, v.shape[0], -1), axis=3)

        temp = np.matmul(temp, temp.transpose((0, 1, 3, 2)))

        numerator = um.transpose().reshape(um.shape[1], um.shape[0], 1, 1) \* temp

        numerator = numerator.sum(0)

        return numerator / denominator

    def \_distance(self, z, v, f):

        dif = np.expand\_dims(z.reshape(z.shape[0], 1, -1) - v.reshape(1, v.shape[0], -1), axis=3)

        determ = np.power(np.linalg.det(f), 1 / self.m)

        det\_time\_inv = determ.reshape(-1, 1, 1) \* np.linalg.pinv(f)

        temp = np.matmul(dif.transpose((0, 1, 3, 2)), det\_time\_inv)

        output = np.matmul(temp, dif).squeeze().T

        return np.fmax(output, 1e-8)

    def next\_u(self, d):

        power = float(1 / (self.m - 1))

        d = d.transpose()

        denominator\_ = d.reshape((d.shape[0], 1, -1)).repeat(d.shape[-1], axis=1)

        denominator\_ = np.power(d[:, None, :] / denominator\_.transpose((0, 2, 1)), power)

        denominator\_ = 1 / denominator\_.sum(1)

        denominator\_ = denominator\_.transpose()

        return denominator\_

    def predict(self, z):

        if len(z.shape) == 1:

            z = np.expand\_dims(z, axis=0)

        dist = self.\_distance(z, self.centers, self.f)

        if len(dist.shape) == 1:

            dist = np.expand\_dims(dist, axis=0)

        u = self.next\_u(dist)

        return np.argmax(u, axis=0)

# Приложение Б

# Программная реализация метода для генерации наименований нечетких кластеров в пространстве разнородных признаков

import numpy as np

import pandas as pd

class MakeClusterName:

    normData = []

    def \_\_init\_\_(self, clusterAlghoritm, eps = 0.1, shablon1 ='',shablon2=' ',shablon3 = ''):

        self.a = clusterAlghoritm

        self.eps = eps

        self.namesX = []

        self.namesY = []

        self.shablonStart = shablon1

        self.shablonBetween = shablon2

        self.shablonEnd = shablon3

    def makeNames(self,data,X\_names, Y\_names , Normalize = False):

        a = self.a

        n\_clusters = a.clusters\_count

        maxX = max(data[:,0])

        minX = min(data[:,0])

        maxY = max(data[:,1])

        minY = min(data[:,1])

        m = np.zeros(2)

        m[0]= int(len(X\_names))

        m[1]= int(len(Y\_names))

        granici\_x , granici\_y = self.\_zone(maxX, minX, maxY, minY, m)

        if Normalize == False:

            self.namesX,self.namesY = self.\_getname(a,data, granici\_x,granici\_y,n\_clusters, X\_names, Y\_names)

        if Normalize == True:

            if len(X\_names) != 3 and len(Y\_names) != 3:

                print("Number of objects more or less then three. This can cause an error. Check your names list")

            self.namesX, self.namesY = self.\_getNormName(a,data, n\_clusters, X\_names, Y\_names)

        return self.namesX,self.namesY

    def \_zone(self,max\_x,min\_x,max\_y,min\_y, m):

        shag\_x = round((max\_x-min\_x)/m[0], 1)

        shag\_y = round((max\_y-min\_y)/m[1], 1)

        granici\_x = [[0]\*2 for i in range(int(m[0]))]

        granici\_y = [[0]\*2 for i in range(int(m[1]))]

        zonex = min\_x

        zoney = min\_y

        for j in range(2):

            if j == 0:

                for x in granici\_x:

                    x[0] = zonex

                    x[1] = zonex+shag\_x

                    zonex = zonex+shag\_x

            if j == 1:

                for y in granici\_y:

                    y[0] = zoney

                    y[1] = zoney+shag\_y

                    zoney = zoney+shag\_y

        return granici\_x , granici\_y

    def \_getname(self,algorithm,data, granici\_x, granici\_y, n\_clusters, name\_x, name\_y):

        numofVectors = data.shape[0]

        names\_x = [0]\*n\_clusters

        names\_y = [0]\*n\_clusters

        for j in range(int(n\_clusters)):

            counter\_x = np.zeros(len(granici\_x))

            counter\_y = np.zeros(len(granici\_y))

            u\_x =np.zeros(len(granici\_x))

            u\_y =np.zeros(len(granici\_y))

            for i in range(numofVectors):

                if algorithm.u[j][i]>0.5:

                    l = 0

                    while l < len(granici\_x):

                        if data[i][0]>granici\_x[l][0] and data[i][0]<=granici\_x[l][1]:

                            counter\_x[l]+=1

                        l+=1

                    l = 0

                    while l < len(granici\_y):

                        if data[i][1]>granici\_y[l][0] and data[i][1]<=granici\_y[l][1]:

                            counter\_y[l]+=1

                        l+=1

            for i in range(int(counter\_x.shape[0])):

                u\_x[i] = counter\_x[i]/sum(counter\_x)

            for i in range(int(counter\_y.shape[0])):

                u\_y[i] = counter\_y[i]/sum(counter\_y)

            names\_x[j] = self.\_nameOnerow(u\_x,name\_x)

            names\_y[j] = self.\_nameOnerow(u\_y,name\_y)

        return names\_x, names\_y

    def \_nameOnerow(self,u\_x , name\_x):

        names = ''

        print("u\_x ", u\_x)

        for i in range(u\_x.shape[0]):

            if u\_x[i]<self.eps:

                continue

            else:

                k = i

                maxim = i

                while k < u\_x.shape[0]:

                    if u\_x[maxim]<u\_x[k]:

                        maxim = k

                    k+=1

                if maxim != u\_x.shape[0]-1 and maxim != 0:

                    if abs(u\_x[maxim] - u\_x[maxim+1])< self.eps and u\_x[maxim+1]> u\_x[maxim-1]:

                        names = name\_x[maxim]+' and '+name\_x[maxim+1]

                        break

                    elif abs(u\_x[maxim] - u\_x[maxim-1])< self.eps and u\_x[maxim+1]< u\_x[maxim-1]:

                        names = name\_x[maxim-1]+' and '+name\_x[maxim]

                        break

                    else:

                        names = name\_x[maxim]

                        break

                elif maxim == 0:

                    if abs(u\_x[maxim] - u\_x[maxim+1])< self.eps:

                        names = name\_x[maxim]+' and '+name\_x[maxim+1]

                        break

                    else:

                        names = name\_x[maxim]

                        break

                elif maxim == u\_x.shape[0]-1:

                    if abs(u\_x[maxim] - u\_x[maxim-1])< self.eps:

                        names = name\_x[maxim-1]+' and '+name\_x[maxim]

                        break

                    else:

                        names = name\_x[maxim]

                        break

        return names

    def showCluster(self,data,nameX ='0',nameY ='0'):

        a = self.a

        n\_clusters = self.a.clusters\_count

        names\_x = self.namesX

        names\_y = self.namesY

        if nameX !='0' and nameY !='0':

            for i in range(n\_clusters):

                if nameX == names\_x[i] and nameY == names\_y[i]:

                    self.\_printCluster(a,data,i)

        elif nameX !='0' and nameY =='0':

            for i in range(n\_clusters):

                if nameX == names\_x[i]:

                    self.\_printCluster(a,data,i)

        elif nameX =='0' and nameY !='0':

            for i in range(n\_clusters):

                if nameY == names\_y[i]:

                    self.\_printCluster(a,data,i)

        else:

            print ('Enter "nameX" or "nameY" ')

    def \_normalizeByVika(self, data):

        normData = pd.DataFrame(data)

        test\_df\_norm = (normData[0] - normData[0].min()) / (normData[0].max() - normData[0].min())

        test\_df\_norm = pd.DataFrame(test\_df\_norm)

        test\_df\_norm[1] = (normData[1] - normData[1].min()) / (normData[1].max() - normData[1].min())

        self.normData = test\_df\_norm.values

        return np.array(test\_df\_norm.values)

    def \_getNormName(self, a,data, n\_clusters, name\_x, name\_y):

        dataNorm = self.\_normalizeByVika(data)

        numofVectors = data.shape[0]

        sumCounter =[0]\*n\_clusters

        names\_x = [0]\*n\_clusters

        names\_y = [0]\*n\_clusters

        granici\_x = [[0, 0.4],[0.3, 0.7],[0.6, 1]]

        granici\_y = [[0, 0.4],[0.3, 0.7],[0.6, 1]]

        for j in range(int(n\_clusters)):

            for i in range(numofVectors):

                if a.u[j][i]>0.5:

                    sumCounter[j] += 1

        for j in range(int(n\_clusters)):

            counter\_x = np.zeros(len(granici\_x))

            counter\_y = np.zeros(len(granici\_y))

            u\_x =np.zeros(len(granici\_x))

            u\_y =np.zeros(len(granici\_y))

            for i in range(numofVectors):

                if a.u[j][i]>0.5:

                    l = 0

                    while l < len(granici\_x):

                        if dataNorm[i][0]>=granici\_x[l][0]:

                            if dataNorm[i][0]<=granici\_x[l][1]:

                                counter\_x[l]+=1

                        l+=1

                    l = 0

                    while l < len(granici\_y):

                        if dataNorm[i][1]>=granici\_y[l][0]:

                            if dataNorm[i][1]<=granici\_y[l][1]:

                                counter\_y[l]+=1

                        l+=1

            for i in range(int(counter\_x.shape[0])):

                u\_x[i] = counter\_x[i]/sumCounter[j]

            for i in range(int(counter\_y.shape[0])):

                u\_y[i] = counter\_y[i]/sumCounter[j]

            names\_x[j] = self.\_nameOnerow(u\_x,name\_x)

            names\_y[j] = self.\_nameOnerow(u\_y,name\_y)

        return names\_x, names\_y

    def \_printCluster(self,a,data,clusterNummber):

        print('cluster № ',clusterNummber, ':')

        print('cluster name - ',self.shablonStart, self.namesX[clusterNummber],self.shablonBetween, self.namesY[clusterNummber], self.shablonEnd, sep='')

        for i in range(data.shape[0]):

            if a.u[clusterNummber][i]>0.5:

                print(data[i])

# Приложение В

# Исходные данные для кластеризации субъектов РФ.

