**Министерство образования и науки РФ**

Федеральное государственное автономное

образовательное учреждение высшего образования

«Казанский (Приволжский) федеральный

университет»

ИНСТИТУТ ВЫЧИСЛИТЕЛЬНОЙ МАТЕМАТИКИ И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

КАФЕДРА АНАЛИЗА ДАННЫХ И ИССЛЕДОВАНИЯ ОПЕРАЦИЙ

Специальность (направление): 01.04.02 – Прикладная математика и информатика

Специализация: Анализ данных и его приложения

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

**Классификация говоров татарского языка**

**на основе морфологических атрибутов**

**Работа завершена:**

"\_\_\_"\_\_\_\_\_\_\_\_\_ 20\_\_ г. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.С.Бадыков

**Работа допущена к защите:**

Научный руководитель

к.ф.-м.н., доцент

"\_\_\_"\_\_\_\_\_\_\_\_\_ 20\_\_ г. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Ф.И.Салимов

Заведующий кафедрой

д.ф.-м.н., доцент

"\_\_\_"\_\_\_\_\_\_\_\_\_ 20\_\_ г. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ М.Д.Миссаров

**Казань – 2017**

Table of Contents

[Введение 4](#_Toc484961621)

[Глава 1. Постановка задачи 10](#_Toc484961622)

[Глава 2. Теоретическая часть 11](#_Toc484961623)

[*2.1 Обзор алгоритмов кластеризации* 11](#_Toc484961624)

[**Алгоритмы иерархической кластеризации** 13](#_Toc484961625)

[**Алгоритмы неиерархической кластеризации** 15](#_Toc484961626)

[*2.2 Анализ системы признаков, заложенной в атласе* 17](#_Toc484961627)

[*2.3 Выбор алгоритмов кластеризации и метрики в пространстве признаков* 22](#_Toc484961628)

[*2.4 Метод многомерного шкалирования* 28](#_Toc484961629)

[Глава 3. Программная реализация и исследование 30](#_Toc484961630)

[3.1 Анализ особенности исходных данных 30](#_Toc484961631)

[3.2 Алгоритм кластеризации транзакционных данных CLOPE 32](#_Toc484961632)

[Описание алгоритма 32](#_Toc484961633)

[Реализация алгоритма 36](#_Toc484961634)

[Результаты 39](#_Toc484961635)

[3.4 Алгоритм кластеризации транзакционных данных ROCK 43](#_Toc484961636)

[Описание алгоритма 43](#_Toc484961637)

[Реализация алгоритма 48](#_Toc484961638)

[Результаты. 50](#_Toc484961639)

[3.5 Алгоритм k-медоидов с метрикой на основе расстояния Ливенштейна 53](#_Toc484961640)

[3.6 Сравнение результатов получившихся результатов 53](#_Toc484961641)

[Глава 4. Результаты работы 54](#_Toc484961642)

[Заключение 54](#_Toc484961643)

[Приложение 1. Алгоритм CLOPE: Полученные кластеры 55](#_Toc484961644)

[Приложение 2. Алгоритм ROCK: Полученные кластеры 62](#_Toc484961645)

# Введение

 Филологи и, в целом, ученые рассматривают лингвистические различия народностей в трех часто пересекающихся плоскостях: социальной, исторической и лингвистической. Социальная плоскость определяет эти различия как признак социальной идентичности. Историческая плоскость предполагает, что на протяжении некоторого времени сохранение лингвистических атрибутов происходило по-разному. Изучая лингвистические различия, особенно шаблоны необычных лингвистических особенностей, можно лучше понять старые формы языка, возможно, даже реконструировать шаблоны общего развития языка.

Эти плоскости могут не соглашаться между собой в том плане, что они могут по-разному объяснять лингвистические различия. Однако, чтобы характеристика отражалась в лингвистической истории языка, она должна передаться от одного поколения к другому. Этот переход не лингвистический, а культурный, поэтому лингвистическая теория, даже имея огромное количество описательных механизмов, должна дополнительно искать описание, которое наиболее подходит данным.

В общем, индивидуальные лингвистические характеристики – слова, конструкции, произношение – слабо связаны с географическим расположением. Для каждой многообещающей характеристики, которая могла бы определять диалект, всегда найдется исключительный контр пример внутри географической области. Далектология - раздел лингвистики, предметом изучения которого является диалект как некоторое целое – была полна вопросов относительно связи далекта с расположением населенного пункта. Но в 1973 Джин Сегай (Jean S´eguy) отметил, что даже если индивидуальные атрибуты ведут к неточным характеризациям диалектов, их обобщение безоговорочно говорит о географической зависимости. Это открытие является рождением диалектометрии.

Диалектометрия занимается синтетически-количественным изучением и интерпретацией лингвистических данных. Методы, используемые в диалектометрии, используются также в естественных, социальных и гуманитарных науках. В наши дни все эти подходы и методы объеденены в такой науке как глубинный анализ данных (data mining). Диалектометрия пытается найти лингвистические шаблоны, которые были скрыты от глаз исследователя.

Очень часто перед исследователями встают задачи, имеющие большую размерность и их структуру, определяющуюся целым рядом признаков, необходимо определить. В частности задача распознавания языков. Очень много языков и подразбиений этих языков. Каждый язык многообразен и периодически претерпевает изменения из-за различных факторов, в результате появляются новые вариации в произношении, в грамматике и в других секторах языка.

В настоящей работе исследовалась вышеописанная задача для татарского языка. Татарский язык широко распространен по всему миру. Территория наибольшего распространения этого языка – Республики Татарстан и Башкортостан. Татарский язык распространен и используется в некоторых районах следующих субъектов России: Марий Эл, Удмуртия, Чувашия, Мордовия, Челябинская, Оренбургская, Свердловская, Тюменская, Ульяновская, Самарская, Астраханская, Саратовская, Нижегородская, Пензенская, Рязанская, Тамбовская, Курганская, Томская область, Пермский край. А также широко распространен в отдельных районах Узбекистана, Казахстана и Киргизии..

Исследование концентрируется в основном на региональном распределении диалектных черт, таких как ядро диалекта и перекрывающиеся зоны, которые могут быть помечены в соответствии с небольшим различием диалектов между граничащими местами. Тем не менее, анализ диалектных отношений не всегда может быть четко изображен, поскольку часто бывают диалект континуум случаи, а также примеры с элементами конвергенции[10].

Языковые атласы служат эмпирической базой данных, которая детально документирует большое количество диалектных контуров. Различные широко-известные численные методологии классификации используются для извлечения и визуализации основной модели из огромного количества данных, найденных в языковых атласах.

Очень часто используются различные методы, включающие алгоритмы классификации такие, как кластеризация, чтобы определить группы внутри диалектов.

Кластеризация представляет собой задачу разбиения заданного множества объектов на различные подмножества, называемые кластерами, таким образом, чтобы кластеры состояли из схожих по свойствам объектов, при этом объекты разных классов отличались. Эта задача является одной из фундаментальных в области интеллектуального анализа данных.

Цель кластеризации – получить разбиение на содержательные классы, при этом полученные кластеры должны отображать естественную структуру исходных данных. Предполагается получить классификацию объектов по группам таким образом, чтобы степень сходства двух представителей одной группы была максимальной и минимальной в противном случае.

Решение задачи кластеризации принципиально допускает различные интерпретации, поскольку не существует однозначно наилучшего критерия качества кластеризации. Число кластеров, как правило, неизвестно заранее и устанавливается в соответствии с некоторым субъективным критерием. Результат кластеризации во многих алгоритмах существенно зависит от метрики, выбор которой обычно определяется экспертом и также чаще всего субъективен. В метрических пространствах “схожесть” векторов, как правило, определяется через норму расстояния. Может рассматриваться как непосредственно взаимное расстояние между векторами, так и расстояние между векторами и некоторым формирующим кластером. То есть при использовании любого существующего алгоритма важно понимать его достоинства и недостатки, учитывать природу данных, с которыми он лучше работает.

Задача нахождения количества групп в множестве данных является одной из основополагающих и трудоёмких в кластерном анализе данных. Кластеризация применяется в очень широком спектре научных областей: статистика, финансовая математика, экономика, биология, оптимизация, в информатике для "интеллектуального" анализа данных, сегментации изображений, распознавания образов, сжатия данных и др.

В зависимости от особенностей конкретной задачи, кластеризация может иметь различные цели:

– определение структуры множества данных, путем разбиения его на группы схожих объектов;

– выделение объектов, не подходящих ни к одному из кластеров;

– упрощение работы с данными, когда рассматриваются не целые классы данных, а лишь типичные представители классов.

Цель данной работы состоит в автоматической классификации говоров татарского языка на базе электронного атласа татарских народных говоров и разработка системы для анализа алгоритмов кластеризации для выбранной системы признаков, а также анализ полученных результатов и их сравнение с экспертным вариантом, предложенным лингвистами ИЯЛИ АН РТ.

Составление атласа говоров того или иного языка, используя компьютерные технологии – очень актуальная и не глубоко изученная в нашей стране проблема. Этому есть ряд причин:

1. Недостаточно данных по отдельным диалектам для исследования этого вопроса;
2. Не изучен вопрос, какой сфере языка надо отдать предпочтение при составлении данных, чтобы получить хорошие результаты: фонетике, морфологии, лексике или синтаксису языка;
3. Сбор и анализ данных – достаточно трудоемкая и требующая тщательного отбора процедура, на которую может уйти много времени, так как информацию нужно получить из всех населенных пунктов, где данный язык распространен.

В некоторых странах подобный анализ уже проводился. В ходе исследований изучался опыт зарубежных ученых по автоматической классификации говоров болгарского языка[2; 3]. Ее авторы – Питер Аутзагерс, Джон Нербонне и Елена Прокик – анализировали болгарские диалекты и попытались найти оптимальное разбиение на кластеры. Также известны работы по кластеризации финских, немецких языков.

В разделе 1 рассматриваются теоретические аспекты задачи кластеризации, дается постановка задачи. В главе 2 описаны используемые методы анализа алгоритмов кластеризации. В главе 3 описаны особенности программной реализации и архитектуры алгоритмов кластеризации, а также приведен сравнительный анализ полученных результатов. В разделе 4 резюмированы результаты дипломной работы. Приведен список использованной литературы, интернет-ресурсов и приложение, содержащее таблицы с данными о разбиении нашего множества объектов на кластеры в отдельных случаях, карты говоров и различные информационные таблицы, полученные и использованные в процессе исследования.

# Глава 1. Постановка задачи

В 2012 году коллективом ученых АН РТ и КФУ был создан электронный атлас татарских народных говоров. Атлас базировался на материалах печатной версии, изданной в 1989 году ИЯЛИ АН РТ и неопубликованных материалах по сибирскому диалекту.

В атласе была предложена экспертная классификация татарских говоров, созданная лингвистами ИЯЛИ АН РТ. Цель настоящей дипломной работы - автоматическая классификация говоров с использованием алгоритмов кластеризации с дальнейшим сравнением результатов с экспертным вариантом.

# Глава 2. Теоретическая часть

# *2.1 Обзор алгоритмов кластеризации*

Термин кластеризации следует понимать, как разбиение исходного множества объектов на некоторые непересекающиеся подмножества (кластеры), с изначально неизвестными параметрами. В зависимости от решаемой задачи, количество кластеров или выбирают на основе каких-либо статистических исследований, или фиксируют. Для всех кластеров характерны внутренняя однородность (все объекты одного класса схожи между собой по определенным признакам) и внешняя изолированность (объекты разных классов существенно отличаются).

Пусть имеется набор данных *(n>0),* где – вектор из m-мерного пространства признаков. А еще имеем функцию, определяющую степень сходства объектов **ρ(,).** Требуется разбить последовательность на непересекающиеся подмножества (кластеры) таким образом, что для каждого кластера, объекты, принадлежащие этому кластеру, были близки друг к другу по метрике ρ, а объекты не из одного кластера имели отличия.

Алгоритм кластеризации можно представить как функцию *A: X → Y,* каждому объекту *xX* ставящую в соответствие метку кластера*Y*. Чаще всего множество Y заранее не известно и дополнительной задачей является определение оптимального числа кластеров с точки зрения тех или иных показателей качества кластеризации.

Основной целью процесса кластеризации является получение интересующей информации о структуре исследуемых данных, что, как правило, является первым этапом на пути их более детального анализа.

Применяя различные методы кластеризации, в результате могут получиться неодинаковые результаты, это связано с характерными особенностями работы тех или иных алгоритмов, последние необходимо учитывать, выбирая метод кластеризации для конкретной задачи.

Следует знать и помнить, что кластерный анализ сопряжен с рядом сложностей:

– выбор метода кластеризации, эффективного для решения конкретной задачи, требует тщательного отбора алгоритмов и условий их применения;

– выбор характеристик, на основании которых проводится кластеризация: метрики, изначальных значений центров кластеров, условий остановки алгоритма. Изначально некорректный выбор приводит к неадекватному разбиению множества на классы;

– выбор изначального числа кластеров. Если нет никаких сведений относительно возможного числа кластеров, необходимо осуществить ряд экспериментов и проанализировать полученные результаты.

– интерпретация результатов кластеризации. Конкретные методы ориентированы на создание кластеров определенных форм и свойств, при этом в исследуемом наборе подобных данных их может и не быть.

Алгоритмы кластеризации могут быть Исключительными, Перекрывающимися, Иерархическими или Вероятностными. В первом случае, данные группируются исключительным способом так, чтобы определенный элемент мог быть помещен только в одну группу (исключительно). С другой стороны, перекрывающийся метод кластеризации использует нечеткие множества для кластеризации данных так, что каждая точка может принадлежать двум или нескольким группам с различными степенями принадлежности. Иерархический алгоритм кластеризации организовывает данные в иерархии и основан на объединении между двумя ближайшими кластерами.

**Алгоритмы иерархической кластеризации**

Алгоритмы иерархической кластеризации, или, по-другому говоря, алгоритмы таксономии, не нацелены на построение одного разбиения выборки на непересекающиеся подмножества, а на системы вложенных разбиений. Результат подобных алгоритмов обычно представляется в виде таксономического дерева — дендрограммы. Классическим примером такого дерева является иерархическая классификация животных и растений.

Алгоритмы иерархической кластеризации разделяют на два основных типа: восходящие и нисходящие алгоритмы.

На первом этапе работы алгоритма все множество объектов представляется отдельными одноэлементными кластерами (в восходящем случае). На последующих этапах наиболее схожие объекты объединяются в кластеры до тех пор, пока все объекты не объединятся в один, либо изначально все объекты принадлежат одному кластеру (в нисходящем случае), который на последующих шагах разделяется на меньшие кластеры, в результате чего образуется последовательность разделяющихся подмножеств.

Рассмотрим сначала восходящую иерархическую кластеризацию. Вначале каждый элемент множества представляется как кластер. Для каждого одноэлементного кластера естественным образом определяется функция схожести[1]

Затем запускается процесс слияний. На каждой итерации вместо пары самых близких кластеров U и V образуется новый кластер . Расстояние от нового кластера W до любого другого кластера S вычисляется по расстояниям и которые к этому моменту уже должны быть известны[1]:

R

где , , β, γ — числовые параметры. Эта универсальная формула обобщает практически все разумные способы определить расстояние между кластерами. Она была предложена Лансом и Уильямсом в 1967 году[1].

На практике используются следующие способы вычисления расстояний R(W,S) между кластерами W и S. Для каждого из них доказано соответствие формуле Ланса-Вильямса при определённых сочетаниях параметров[9]:

*Расстояние ближнего соседа:*

β=0, γ=.

*Расстояние дальнего соседа:*

β=0, γ=.

*Среднее расстояние:*

β= γ =0.

*Расстояние между центрами*:

β= , γ =0.

*Расстояние Уорда:*

β= , γ =0.

Главное достоинство методов этой группы состоит в том, что они наиболее наглядны для исследователя и дают возможность получить более детальную информацию о структуре данных. Недостатки: негибкая классификация, получаемая на выходе, ограничение в объеме данных, подлежащих анализу. Большая трудоемкость описанных методов делает их неприменимыми при большом объеме множества объектов, структуру которых следует изучить.

**Алгоритмы неиерархической кластеризации**

Неиерархические алгоритмы основываются на построении разбиения множества объектов на определенное, фиксированное количество кластеров и, во многих случаях, оптимизации целевой функции, которая определяет, насколько оптимально (оптимальность обусловливается особенностями алгоритма) составленное разбиение множества на кластеры. Итеративный процесс ограничивается условием остановки, зависящим от параметра алгоритма.

Достоинства этого типа методов определяются более высокой устойчивостью по отношению к шумам, выбранной метрике, добавлению дополнительных незначимых объектов в исходные данные, участвующих в кластеризации. Необходимо заранее установить параметры для кластеризации, а также и количество кластеров, и количество итераций или правило остановки и некоторые другие параметры.

Одним из наиболее популярных и широко распространенных аналогичных методов является алгоритм k–means (k–средних). Данный алгоритм направлен на сокращение среднеквадратичного отклонения внутри кластеров:

где – набор векторов принадлежащих–му кластеру, а – среднее значение соответствующих векторов

В основе данного алгоритма лежит следующая идея: на каждой итерации для каждого кластера вычисляется новый центр масс, после этого исходные векторы снова разбиваются на новые кластеры, в зависимости от того, который из вновь вычисленных центров ближе по метрике.

Данный алгоритм является одним из самых простых и самых быстрых алгоритмов кластеризации. Не смотря на это, данный алгоритм имеет свой существенный недостаток. Стартовые кластеры выбираются случайно и из-за этого результаты  алгоритма поиска *k*-средних очень часто меняются в последующих циклах.

Сегодня насчитывается много различных модификаций данного алгоритма, которые нацелены на устранение вышеописанных недостатков данного метода.

# *2.2 Анализ системы признаков, заложенной в атласе*

Атлас татарских народных говоров[8] включает в себя все основные районы расселения татар и отражает сведения по фонетике, морфологии, лексике и синтаксису татарского языка, собранные на основе специально разработанной программы по сбору материалов для диалектологического атласа (1959 г.). Материалы для атласа собирались в течение 1958 – 1980 гг. (первые два тома) и 1997 - 2008 гг. для 3-го тома во время многочисленных экспедиций, организованных ИЯЛИ АН РТ. Первые два тома охватывают татарские народные говоры, распространенные на территории областей и республик Среднего Поволжья и Приуралья; третий том - говоры татарского населения Западной Сибири. При сборе информации происходила тщательная выверка собираемого материала, его интерпретация, паспортизация. Расстояния между охватываемыми для исследования населенными пунктами не превышало 10-15км.

В структурном отношении атлас состоит из 215 карт языковых явлений, исторических карт и сводных карт изоглосс, каждая из которых визуализируется в отдельной экранной форме.



Рисунок 1.

Структура электронного атласа с указанием количества карт в каждом разделе

База данных атласа содержит информацию о распределении языковых явлений по 1031 населенным пунктам. На картах отражены основные признаки диалектного различия, закономерности и характер распространения диалектных явлений.

Электронный атлас[8] состоит из картографической и атрибутивной баз данных. Картографическая база содержит пространственную информацию, разбитую на слои: административное деление РФ по состоянию на 1995 год, основные водные объекты на территории РФ, населенные пункты (НП), в которых проводился сбор информации. Разнесение информации по слоям позволяет проводить ее фильтрацию и показывать карты в виде удобном конечному пользователю. Атрибутивная база включает в себя данные по распределению языковых явлений по населенным пунктам, а также содержит дополнительную информацию по фонетическим, морфологическим и лексическим особенностям диалектов татарского языка.

Наша база данных содержит ранговую информацию по признакам. У каждого признака несколько рангов(>=2), то есть вариантов употребления этого признака. Для наглядности приведем примеры из каждой группы признаков.

Фонетические:

1. Лабиализованный и нелабиализованный варианты гласного *а* (в односложных корневых словах). Всего имеется три вариации употребления: , a,// a.
2. Распространение различных вариантов гласного ә. Варианты: ә гласный переднего ряда, нижнего подъема (литературный тип), ә более переднего образования, ә узкий вариант ә, ә главным образом после узких гласных.
3. Соответствие губным гласным *о-ө* негубных гласных *ы-е* или слабоогубленных *ыо-еө* в первом слоге. Множество вариантов: о-ө: он, сөт; ы-е: ын, сет; *ыо-еө* : *ыо*н, с*еө*т; о-ө//ы-е: он//ын, сөт//сет; ыо-еө//ы-е: ыон//ын, сеөт//сет.
4. и~иэ~е~иә~ә~ˆә в частице бик ‘очень’. Имеется 6 вариантов: бик/би', биэк/биәк, бәк, бик//бәк, бик//биэк/бˆәк, отсутствие слова.
5. Наличие губной гармонии. Отобранные варианты: озон, төлкө; частичное употребление: озыон, төлкәө; озон, но: телке; ызын, телке; озон//ызын, төлкө//телке.
6. Изменение звукосочетаний уй, уйы в группе слов. Варианты: уйна-, уйла-, куйын, муйын; ун'а-, ул'-, кун', мун'/бун'; уйна-/ун'а-, уйла-/ул'а-, куйын//кун', муйын//мун'/бун'; редкие случаи монофтонгизации.
7. д~з в словах: Идел ‘Волга’, ‘река’, кадак  ‘гвоздь’, идән  ‘пол’, бодай  ‘пшеница’, надан  ‘невежда’. Варианты: д, з, з не во всех приведенных словах, д//з, различные единичные случаи д~з.

Морфологические:

1. Указательное местоимение шул ‘этот’, ‘тот’ и его соответствия. Варианты: шушы/шышы, тушы, тушы/шушы, ошто, ошо/ушы, шул/шу л/шол/шол/шыл/шал, шушал, шу, ул и бу в значении «шул».
2. Распространение указательного местоимения шал (~литер. шул ‘этот’). Варианты: отмечено, не отмечено.
3. Распространение слов анау ‘вон тот’, монау ‘вот этот’. Варианты: анау, монау; редкое употребление; не отмечены.
4. Форма сравнительной степени на -дырак/-дерәк (после гласных и согласных з, й, л, р, н, напр., салкындырак ‘холодноватый’). Варианты использования: довольно активное употребление, отдельные случаи употребления, не отмечено.
5. Деепричастие ди-ди ‘говоря’. Варианты: ди-ди активно, редкие случаи употребления ди-ди, отмечена только форма дип/дийп.

Лексические:

1. Распространение слова ызба ‘дом’. Варианты: ызба/ызма, не отмечено.
2. Названия теплого помещения для скота. Варианты: абзар/азбар/абdар/бурабзар/ылабзар..., аран, сарай, абзар...//сарай, аран//абзар..., токон, токон//абзар..., кура, кура//аран, другие названия.
3. Значения слова морҗа /мөрҗә /морз'а /мөрзә /морйа /мира /мирйа /миралык. Варианты: ‘дымоходная труба’; ‘печь’; 1) ‘дымоходная труба’, 2) ‘печь’; это слово не отмечено.

Синтаксические:

1. Распространение формы на -мастан (алмастан ‘не взяв’, ‘чтобы не взял’, ‘так как не взял’, ‘прежде чем взять’). Варианты: наличие формы на –мастан, указанная форма не отмечена.
2. Распространение деепричастия на -сынга/-сыңға (син күрсеңгә ‘чтобы ты видел’). Варианты: активное употребление, редкое употребление, не отмечено.
3. Употребление формы на -ныкы/-ныңкы/нынкысы в определительной функции. Варианты: отмечено, не отмечено.

Мы тут привели лишь малую часть признакового пространства – всего 18 особенностей из 215. Каждый вариант употребления признака имеет свой код, который и идентифицирует этот вариант использования. Видим, что у некоторых признаков варианты употребления трудноразличимы, где-то всего две вариации употребления, а где-то десять и даже больше. Встает вопрос, как же с ними обращаться. Первое, что приходит на ум – это разделить данный признак так, чтобы пересечений уже не было. Но этот вариант решения данной проблемы требует достаточно трудоемкого лингвистического анализа признаков. Было принято решение из всего множества признаков для исследований изучить лишь раздел морфологии, ведь морфологи является наиболее значимой частью языка и наиболее четко отражает языковые особенности, разнообразие использования языка.

Электронный атлас[8] можно рассматривать как объект, заданный в трехмерном пространстве, где каждая карта, являясь плоскостью, несет в себе информацию по одному языковому явлению. Набор карт создает объемный образ, и запросы по выбранному набору позволяют сопоставить распределения соответствующих языковых явлений, что может оказаться полезным при решении задач кластеризации говоров.

# *2.3 Выбор алгоритмов кластеризации и метрики в пространстве признаков*

Следующий возникающий вопрос, на основе чего следует определить «схожесть» объектов и определить понятие расстояния между объектами? Сначала составляется вектор признаков для каждого объекта, обычно это числовые значения, такие как вес-рост человека. Но, тем не менее, есть еще и алгоритмы, которые оперируют с ранговыми характеристиками.

Существует множество метрик, вот лишь основные из них:

1. Евклидово расстояние

Является одной из наиболее часто используемых функций вычисления расстояния между объектами. Представляется как геометрическое расстояние , вычисляемое в многомерном пространстве:

2. Квадрат евклидова расстояния

Эта метрика придает больший вес объектам, расположенным как можно дальше друг от друга. Для вычисления расстояния по этой метрике используется нижеприведенная формула:

3. Расстояние городских кварталов (манхэттенское расстояние)

Под этим расстоянием подразумевается покоординатная сумма разностей. В большинстве случаев эта мера расстояния приводит к результатам аналогичным результатам, при использовании расстояния Евклида. Но следует отметить, что влияние отдельных больших разностей или, по-другому говоря, выбросов снижается и менее ощутимо, так как отсутствует возведение в квадрат. Соответствующая формула для расчета расстояний:

4. Расстояние Хэмминга

Расстояние Хемминга – это число позиций, в которых соответствующие символы сравниваемых слов одинаковой длины различны. В широком смысле, расстояние Хэмминга предназначено для сравнения строк одинаковой длины из какого-либо q-ичного алфавита. Оно является метрикой отражающей различия между объектами с одинаковой размерностью. Данная метрика применима только в случаях, когда данные являются ранговыми. Для расчета расстояния Хэмминга применяется формула:

Нидерландские ученые, исследовавшие вопрос выбора подходящей метрики[2], использовали метрику, называемую «расстоянием Ливенштейна», для вычисления расстояний между произношениями того или иного слова в разных населенных пунктах. Над применимостью этого метода к болгарских диалектам, Нербонне и Прокик[2] работали с 2006 года по 2009. В ходе их исследований ими было отобрало 156 самых общеупотребительных слов и собран материал по произношению этих слов в различных населенных пунктах.

«Расстояние Ливенштейна»[11] между двумя строками – это наименьшее количество операций вставки, удаления и замены одного символа, которые необходимо совершить для получения из одной строки другую.

Самым первым человеком, упомянувшим эту задачу, был советский математик [Владимир Иосифович Левенштейн](https://ru.wikipedia.org/wiki/%D0%9B%D0%B5%D0%B2%D0%B5%D0%BD%D1%88%D1%82%D0%B5%D0%B9%D0%BD,_%D0%92%D0%BB%D0%B0%D0%B4%D0%B8%D0%BC%D0%B8%D1%80_%D0%98%D0%BE%D1%81%D0%B8%D1%84%D0%BE%D0%B2%D0%B8%D1%87) в [1965 году](https://ru.wikipedia.org/wiki/1965_%D0%B3%D0%BE%D0%B4), при изучении последовательностей [11]. В дальнейшем, задачу более общего характера для произвольного алфавита стали связывать с его именем.

У этого метода, наряду с достоинствами, есть и свои недостатки:

1. При перестановке местами слов или частей слов могут получиться относительно большие расстояния;
2. Небольшими могут быть расстояния между совершенно разными короткими словами, а вот расстояния между очень похожими, но длинными словами будут оказываться намного больше, чем следовало бы оценить расстояние между ними.

Опишем алгоритм для вычисления «расстояния Ливенштейна». Пусть и - две строки (длиной M и N соответственно) над некоторым алфавитом, тогда расстояние d(, ) можно подсчитать по следующей рекуррентной формуле

d(, )=D(M, N), где

, где m(a, b) равно нулю, если a=b и единице в противном случае. min(a,b,c) возвращает наименьший из аргументов. Здесь шаг по i символизирует удаление из первой строки, по j – вставку в первую строку, а шаг по обоим индексам символизирует замену символа или отсутствие изменений.

Имеют место следующие утверждения [11]:

* {\rm{d}}(S_1,S_2) \geqslant \bigl| |S_1| - |S_2| \bigr|
* {\rm{d}}(S_1,S_2) \leqslant \max\bigl( |S_1| , |S_2| \bigr)
* \mathop{\rm{d}}(S_1,S_2) = 0 \Leftrightarrow S_1 = S_2

Доказательство этих соотношений проведено в [11].

На первом шаге, сравнив попарно транскрипции всех слов методом Ливенштейна, они для каждого признака составили матрицы расстояний. А потом, на втором шаге, на основе этих матриц составили общую матрицу расстояний, где элемент - это расстояние между объектами i и j, представляющее собой обычную сумму элементов, расположенных на пересечении i-й строки и j-го столбца, по всем матрицам из предыдущего шага. Таким образом, они получили матрицу расстояний, которую можно подавать различным кластерным алгоритмам и получить необходимое разбиение для дальнейших исследований и сравнения с экспертным вариантом. Они использовали нижеописанные алгоритмы, а потом сравнили результаты.

*Правила объединения или связи*

Используя иерархические алгоритмы, необходимо знать ответ на вопросы о том, как определить, какие из всех кластеров следует объединить, как вычислять степень схожести между ними. На первом этапе каждый объект представляется как кластер, состоящий из одного элемента, и расстояния между ними вычисляются согласно выбранной мере. Но проблема возникает уже на втором шаге, когда нужно объединить несколько объектов в один кластер и всплывает вопрос, как правильно определить расстояния между разными кластерами? Иначе говоря, следует придумать правило объединения нескольких кластеров. Существуют различные варианты решения этой проблемы: можно связать два кластера вместе, если любые два элемента, отобранные из двух кластеров, ближе друг к другу, чем соответствующее значение расстояния связи. Этот подход для определения расстояния между кластерами назван правилом ближайшего соседа, его также называют методом одиночной связи. Метод строит "волокнистые" кластеры, иначе говоря, кластеры, "сцепленные вместе" только отельными объектами, оказавшимися случайно ближайшими друг к другу. В качестве альтернативы, возможен вариант использования соседей в кластерах, находящихся дальше друг от друга, чем все остальные пары объектов. Другое название данного метода – метод полной связи. Известно большое количество других методов объединения кластеров, схожих с теми, которые только что были приведены, рассмотрим их:

1. Одиночная связь (расстояния ближайшего соседа, single link)

Этот метод, в качестве расстояния между двумя кластерами, рассматривает расстояние между двумя наиболее близкими объектами (ближайшими соседями), находящимися в разных кластерах. В итоге работы этого метода, кластеры стремятся объединиться в цепочки.

1. Полная связь (расстояние наиболее удаленных соседей, complete link)

Метод находит расстояние между кластерами как наибольшее расстояние между любыми двумя объектами из разных кластеров (то есть наиболее удаленными соседями). Метод лучше всего использовать, если объекты происходят из отдельных групп. Ну а если кластеры имеют цепочечную форму, то этот метод неприменим.

1. Невзвешенное попарное среднее (upgma или, по-другому, average)

Расстояние между двумя кластерами определяется средним расстоянием между всеми парами объектов, входящими в них. Данный способ стоит использовать или в случае, если объекты из различных групп, или при присутствии кластеров, имеющих цепочечный тип, или при наличии кластеров неравных размеров.

1. Взвешенное попарное среднее (wpgma)

Данный метод очень схож с методом невзвешенного попарного среднего, отличие состоит лишь в том, что при вычислениях в качестве весового коэффициента используются размеры кластеров (то есть число объектов, содержащихся в них). По этой причине, данный метод наиболее подходит, когда предполагаются неравные размеры кластеров.

1. Метод Варда(Уорда)

Как расстояние между кластерами рассчитывается прирост суммы квадратов расстояний объектов до центров кластеров, который выделяется при их объединении. Алгоритм отличается от всех других тем, что здесь для оценивания расстояния между кластерами используются методы дисперсионного анализа. На каждом шаге минимизируется сумма квадратов между кластерами. В общем, метод достаточно эффективен, стремится занести близко расположенные кластеры в один кластер, однако, он имеет тенденцию создавать кластеры с маленькой размерностью.

Рассмотрим метод кластеризации, существенно отличающийся от агломеративных методов, это метод k-средних. Представим, что имеются некоторые предположения о количестве кластеров. Теперь можно потребовать у системы образовать k кластеров таким образом, чтобы полученные кластеры были максимально различны. Метод k-средних решает задачи именно такого характера. В начале работы алгоритма случайно выбираются k кластеров, чтобы затем принадлежность к кластерам объектов изменить. Это делается с целью:

1. – минимизировать изменчивость внутри кластеров,
2. – максимизировать изменчивость между кластерами.

Метод вычисления «расстояния Ливенштейна» тоже непригоден для наших данных, потому что в [2; 3] признаками выступают слова и расстояние определяется числом несовпадений в произношении этих слов, а в нашем случае признаки имеют совсем другой характер и необходимо подобрать и использовать метрику, наиболее подходящую для анализа нашего пространства признаков.

Решение этого вопроса рассмотрим немного позднее, а сейчас рассмотрим метод, использующийся для сокращения размерности признакового пространства. В частном случае, при уменьшении размерности пространства до 2-х, метод используется для визуализации многомерных данных на плоскости.

# *2.4 Метод многомерного шкалирования*

Сформулируем задачу, которую решает многомерное шкалирование. Дана квадратная матрица ***R***размерностью***n* x*****n*** и известно, что элемент этой матрицы, находящийся на пересечении ***i***-й строки и ***j***-го столбца, содержит сведения о сходстве между анализируемыми объектами ***i*** и ***j***. На выходе работы алгоритма многомерного шкалирования получаются числовые значения координат. Эти координаты приписываются объектам в новой системе координат (во "вспомогательных шкалах").

Многомерное шкалирование – не только математическое преобразование, но и возможность более удачно расположить объекты, приближенно сохраняя расстояния от одного объекта до другого в новом меньшем по размерам признаковом пространстве.

Многомерное шкалирование создано для анализа многомерных данных и отличается от остальных схожих методов, прежде всего, видом исходных данных. В данном случае это матрица схожести между объектами, а для вычисления этой матрицы существует множество различных метрик. В виде цели многомерного шкалирования выделяются следующие:

1. точное описание матрицы схожести на основе расстояний между объектами;
2. преобразование данных о близости объектов в систему точек в пространстве меньшей размерности. Чаще всего, стараются отобразить данные на двумерной плоскости.

Проще говоря, на вход метода подается матрица схожести, а в результате получаем размещение точек в пространстве меньшей размерности.

Перечислим основные возможности метода многомерного шкалирования:

1. Сохраняя наилучшим образом структуру исходных данных о расстоянии между парами объектов, построение признакового пространства меньшей размерности. Процесс шкалирования заключается в проектировании объектов на оси нового пространства.

2. Возможность визуализации структуры исходных данных в виде множества точек, соответствующим объектам, в двухмерном или трехмерном пространстве.

# Глава 3. Программная реализация и исследование

База данных состоит из 1031-го населенного пункта и 215 характеристик этих населенных пунктов. В процессе кластеризации и анализа результатов необходимо учесть тот факт, что результаты напрямую зависят от того, какие критерии множества исходных данных мы используем. Всякого рода неточности могут возникнуть при сокращении размерности исходного множества данных, в случае обобщения, есть риск потерять некоторую информацию, особенность объектов. Принимая во внимание эти утверждения, были проведены исследования для объектов в пределах Республики Татарстан, так как это зона основного распространения татарского языка. Это решение принято для того, чтобы проанализировать данные на задаче меньшей размерности. Для исследований было выбрано множество морфологических признаков.

## 3.1 Анализ особенности исходных данных

Из приведенных примеров атрибутов можно видеть, что они имеют категорийный характер, и мы не можем в явном виде сравнивать значения атрибутов между собой, потому что мы не можем сказать, какое из них более значимо. Задача анализа категорийных данных встречается не только в лингвистике.

Задачи кластеризации больших массивов категорийных данных весьма актуальна для систем анализа данных. Категорийные данные встречаются в любых областях: производство, коммерция, маркетинг, медицина и т.д. Категорийные данные включают в себя и так называемые транзакционные данные: чеки в супермаркетах, логи посещений веб-ресурсов. Сюда же относится анализ и классификация текстовых документов.

Здесь и далее под категорийными данными понимаются качественные характеристики объектов, измеренные в шкале наименований. При использовании шкалы наименований указывается только, одинаковы или нет объекты относительно измеряемого признака.

Применять для кластеризации объектов с категорийными признаками традиционные алгоритмы неэффективно, а часто – невозможно. Основные трудности связаны с высокой размерностью и гигантским объемом, которыми часто характеризуются такие базы данных. В наших данных всего 49 атрибутов, у каждого атрибута около 5-6 значений. Таким образом, в наших данных всего в районе 250 атрибутов.

Алгоритмы, основанные на парном вычислении расстояний (k-means и аналоги) эффективны в основном на числовых данных. Их производительность на массивах записей с большим количеством нечисловых факторов неудовлетворительная. И дело даже не столько в сложности задания метрики для вычисления расстояния между категорийными атрибутами, сколько в том, что на каждой итерации алгоритма требуется попарно сравнивать объекты между собой, а итераций может быть очень много. Для таблиц с миллионами записей и тысячами полей это неприменимо.

Поэтому в последнее десятилетие ведутся активные исследования в области разработки масштабируемых (scalable) алгоритмов кластеризации категорийных и транзакционных данных. К ним предъявляются особые требования, а именно:

- минимально возможное количество "сканирований" таблицы базы данных;

- работа в ограниченном объеме оперативной памяти компьютера;

- работу алгоритма можно прервать с сохранением промежуточных результатов, чтобы продолжить вычисления позже;

- алгоритм должен работать, когда объекты из базы данных могут извлекаться только в режиме однонаправленного курсора (т.е. в режиме навигации по записям).

## 3.2 Алгоритм кластеризации транзакционных данных CLOPE

### Описание алгоритма

CLOPE предложен в 2002 году группой китайских ученых. При этом он обеспечивает более высокую производительность и лучшее качество кластеризации в сравнении с многими иерархическими алгоритмами.

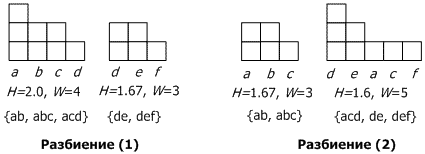
Под термином *транзакция* здесь понимается некоторый произвольный набор объектов, будь это список ключевых слов статьи, товары, купленные в супермаркете, множество симптомов пациента, характерные фрагменты изображения и так далее. В нашем случае транзакцией является набор морфологических значений атрибутов одного населенного пункта. Задача кластеризации транзакционных данных состоит в получении такого разбиения всего множества транзакций, чтобы похожие транзакции оказались в одном кластере, а отличающиеся друг от друга – в разных кластерах.

В основе алгоритма кластеризации CLOPE лежит идея максимизации глобальной функции стоимости, которая повышает близость транзакций в кластерах при помощи увеличения параметра *кластерной гистограммы*. Рассмотрим простой пример из 5 транзакций: {(a,b), (a,b,c), (a,c,d), (d,e), (d,e,f)}. Представим себе, что мы хотим сравнить между собой следующие два разбиения на кластеры:

(1) {{ab, abc, acd}, {de, def}}

(2) {{ab, abc}, {acd, de, def}}.

Для первого и второго вариантов разбиения в каждом кластере рассчитаем количество вхождений в него каждого элемента транзакции, а затем вычислим высоту (H) и ширину (W) кластера. Например, кластер {ab, abc, acd} имеет вхождения a:3, b:2, c:2 с H=2 и W=4. Для облегчения понимания на рис. 1 эти результаты показаны геометрически в виде гистограмм.



*Рисунок 1. Гистограммы двух разбиений*

Качество двух разбиений оценим, проанализировав их высоту H и ширину W. Кластеры {de, def} и {ab, abc} имеют одинаковые гистограммы, следовательно, равноценны. Гистограмма для кластера {ab, abc, acd} содержит 4 различных элемента и имеет площадь 8 блоков (H=2.0, H/W=0.5), а кластер {acd, de, def} – 5 различных элементов с такой же площадью (H=1.6, H/W=0.32). Очевидно, что разбиение (1) лучше, поскольку обеспечивает большее наложение транзакций друг на друга (соответственно, параметр H там выше).

На основе такой очевидной и простой идеи геометрических гистограмм и работает алгоритм CLOPE (англ.: Clustering with sLOPE). Рассмотрим его подробнее в более формальном описании.

Алгоритм CLOPE

Пусть имеется база транзакций D, состоящая из множества транзакций {t1,t2,…,tn}. Каждая транзакция есть набор объектов {i1,…,im}. Множество кластеров {C1,…,Ck} есть разбиение множества {t1,…,tn}, такое, что C1 … Ck={t1,…,tn} и Ci≠∅∧Ci⋂Cj=∅Ci≠∅∧Ci⋂Cj=∅, для 1<=i, j<=k. Каждый элемент Ci называется *кластером*, n, m, k – количество транзакций, количество объектов в базе транзакций и число кластеров соответственно.

Каждый кластер C имеет следующие характеристики:

D(C) – множество уникальных объектов;

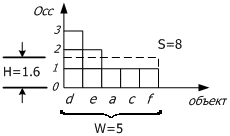
Occ(i,C) – количество вхождений (частота) объекта i в кластер C;

S(C)=∑i∈ D(C) Occ (i,C)=∑ti∈C∣ti∣S(C)=∑i∈ D(C) Occ (i,C)=∑ti∈C∣ti∣

W(C) = |D(C)|;

H(C) = S(C)/W(C).

*Гистограммой* кластера C называется графическое изображение его расчетных характеристик: по оси OX откладываются объекты кластера в порядке убывания величины Occ(i,C), а сама величина Occ(i,C) – по оси OY (рис. 2).

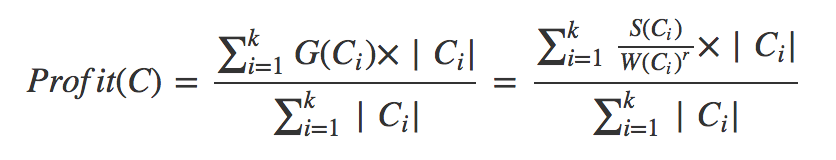


*Рисунок 2. Иллюстрация гистограммы кластера*

На рис. 2 S(C), равное 8, соответствует площади прямоугольника, ограниченного осями координат и пунктирной линией. Очевидно, что чем больше значение H, тем более "похожи" две транзакции. Поэтому алгоритм должен выбирать такие разбиения, которые максимизируют H.

Однако учитывать одно только значение высоты H недостаточно. Возьмем базу, состоящую из 2-х транзакций: {abc, def}. Они не содержат общих объектов, но разбиение {{abc, def}} и разбиение {{abc}, {def}} характеризуются одинаковой высотой H=1. Получается, оба варианта разбиения равноценны. Но если для оценки вместо H(C) использовать градиент G(C)=H(C)/W(C)=S(C)/W(C)2, то разбиение {{abc},{def}} будет лучше (градиент каждого кластера равен 1/3 против 1/6 у разбиения {{abc, def}}).

Обобщив вышесказанное, запишем формулу для вычисления глобального критерия – функции стоимости Profit(C):



|Ci| – количество транзакций в i-том кластере, k – количество кластеров, r – положительное вещественное число большее 1.

С помощью параметра r, названного авторами CLOPE *коэффициентом отталкивания* (repulsion), регулируется уровень сходства транзакций внутри кластера, и, как следствие, финальное количество кластеров. Этот коэффициент подбирается пользователем. Чем больше r, тем ниже уровень сходства и тем больше кластеров будет сгенерировано.

Формальная постановка задачи кластеризации алгоритмом CLOPE выглядит следующим образом: для заданных D и r найти разбиение C: Profit(C,r) -> max.

### Реализация алгоритма

Предположим, что транзакции хранятся в таблице базы данных. Лучшее решение ищется в течение последовательного итеративного перебора записей базы данных. Поскольку критерий оптимизации имеет глобальный характер, основанный только на расчете H и W, производительность и скорость алгоритма будет значительно выше, чем при попарном сравнении транзакций.

Реализация алгоритма требует первого прохода по таблице транзакций для построения начального разбиения, определяемого функцией Profit(C,r). После этого требуется незначительное (1-3) количество дополнительных сканирований таблицы для повышения качества кластеризации и оптимизации функции стоимости. Если в текущем проходе по таблице изменений не произошло, то алгоритм прекращает свою работу. Псевдокод алгоритма имеет следующий вид.

1. // Фаза 1 – инициализация
2. Пока не конец
3. прочитать из таблицы следующую транзакцию [t, -];
4. положить t в существующий либо в новый кластер Ci, который дает максимум Profit(C,r);
5. записать [t,i] в таблицу (номер кластера);
6. // Фаза 2 – Итерация
7. Повторять
8. перейти в начало таблицы;
9. moved := false;
10. пока не конец таблицы
11. читать [t,i];
12. положить t в существующий либо в новый кластер Cj, который максимизирует Profit(C,r);
13. если Ci<>Cj тогда
14. записать [t,i];
15. moved := true;
16. пока (not moved).
17. удалить все пустые кластеры;

Как видно, алгоритм CLOPE является масштабируемым, поскольку способен работать в ограниченном объеме оперативной памяти компьютера. Во время работы в RAM хранится только текущая транзакция и небольшое количество информации по каждому кластеру, которая состоит из: количества транзакций N, числа уникальных объектов (или ширины кластера) W, простой хэш-таблицы для расчета Occ(i,C) и значения S площади кластера. Они называются *кластерными характеристиками* (CF – cluster features). Для простоты обозначим их как свойства кластера C, например, C.Occ[i] означает число вхождений объекта i в кластер C и т.д. Можно посчитать, что для хранения частоты вхождений 10 тыс. объектов в 1 тыс. кластерах необходимо около 40 Мб оперативной памяти.

Для завершения реализации алгоритма нам нужны еще две функции, рассчитывающие прирост Profit(C,r) при добавлении и удалении транзакции из кластера. Это легко сделать, зная величины S,W и N каждого кластера:

1. function DeltaAdd(C,t,r): double;
2. begin
3. S\_new := C.S + t.ItemCount;
4. W\_new := C.W;
5. for i:=0 to t.ItemCount–1 do
6. if (C.Occ[t.items[i]]=0) then W\_new := W\_new + 1;
7. result := S\_new\*(C.N+1)/(W\_new)r–C.S\*C.N/(C.W)r
8. end;

Здесь t.Items[i] – значение i-го объекта транзакции t. Заметим, что DeltaAdd(C,t,r) при добавлении t в новый кластер равна S/Wr, где S и W – площадь и ширина кластера, состоящего из добавляемой транзакции t.

Реализация функции прироста Profit(C,r) при удалении транзакции похожа на DeltaAdd(C,t,r), поэтому опустим ее подробный код.

Следующая теорема гарантирует корректность использования функции DeltaAdd.

**Теорема.** Если DeltaAdd(Ci,t) есть максимум, то перемещение t в кластер Ciмаксимизирует Profit(C,r).

Теперь можно оценить вычислительную сложность алгоритма CLOPE. Пусть средняя длина транзакции равна A, общее число транзакций N, максимально возможное число кластеров K. Временная сложность одной итерации равна O(N\*K\*A), показывающая, что скорость работы алгоритма растет линейно с ростом кластеров и размера таблицы. Это делает алгоритм быстрым и эффективным на больших объемах.

Была произведена имплементация данного алгоритма на функциональном языке программирования Elixir по оригинальной статье авторов алгоритма Yiling Yang, Xudong Guan и Jinyuan You (<http://www.inf.ufrgs.br/~alvares/CMP259DCBD/clope.pdf>.

Были написаны тесты для валидации корректноcти выполнения (в том числе интеграционные). В качестве сервиса для непрерывной интеграции (CI) был иcпользован<https://semaphoreci.com/>.

На вход принимается массив кортежей вида {название транзакции, [объекты транзакции] } и вспомогательный параметр repulsion, необходимый для функции качества и определяющий близость кластеров.

На выход получаем массив кластеров (классов) из тех же кортежей.

Пример вызова:

**input = [**

**{"transaction1", ["object1", "object2", "object3"]},**

**{"transaction2", ["object1", "object5"]},**

**{"transaction3", ["object2", "object3"]},**

**{"transaction4", ["object1", "object5"]}**

**]**

**iex> result = input |> Clope.clusterize(2)**

**[**

**[**

**{"transaction1", ["object1", "object2", "object3"]},**

**{"transaction3", ["object2", "object3"]}**

**],**

**[**

**{"transaction2", ["object1", "object5"]},**

**{"transaction4", ["object1", "object5"]}**

**]**

**]**

Исходный код, реализованного алгоритма, находится в открытом доступе и доступен по адресу<https://github.com/ayrat555/clope>. Библиотека включена в список awesome elixir (https://github.com/h4cc/awesome-elixir) – список лучших библиотек на языке Elixir - в раздел Алгоритмы и Структуры Данных На момент написания текста работы она была загружена больше 30 раз.

### Результаты

Определение диалектов.

Экспертные данные

В Татарстане всего 3 диалекта:

1. Средный – 205 нас. пунктов

2. Западный – 66 нас. пунктов

3. Переходный – 2 нас. пункта

Полученные результаты

Параметр repulsion = 1

1. 1-ый кластер – 220 нас. пунктов

2. 2-ой кластер - 51 нас. пунктов

Отличия населенных пунктов Среднего диалекта от 1-ого кластера 17 населенных пунктов, причем все населенные пункты из Среднего диалекта присутствуют в 1-ом кластере, также 2 населенных пункта из Переходного говора попали в 1-ый кластер.

Нужно отметить, что отличие западного диалекта от полученного первого кластера только в населенных пунктах Чистопольского говора. В основном совпадение полное.

Во втором полученном кластере присутствуют населенные пункты из Западного диалекта и нет населенных пунктов из других классов экспертного разбиения, т.е они полностью пересекаются.

Таким образом, совпадение полученных диалектов составляет 93%.

Определение говоров.

Экспертные данные

В Татарстане всего 18 говоров.

1. Диалект: Средний. Говор: Балтачевский. 21 нас. пункт.

2. Диалект: Средний. Говор: Заказанско-кряшенский. 7 нас. пункт.

3. Диалект: Средний. Говор: Нижнекамско-кряшенский. 21 нас. пункт.

4. Диалект: Средний. Говор: Мамадышский. 30 нас. пункт.

5. Диалект: Средний. Говор: Мензелинский. 71 нас. пункт.

6. Диалект: Средний. Говор: Дубъязский. 17 нас. пункт.

7. Диалект: Средний. Говор: Лаишевский. 12 нас. пункт.

8. Диалект: Средний. Говор: Нурлатский. 12 нас. пункт.

9. Диалект: Средний. Говор: Камско-устьинский. 6 нас. пункт.

10. Диалект: Средний. Говор: Камско-устьинский. 8 нас. пункт.

11. Диалект: Западный. Говор: Подберезинский. 2 нас. пункт.

12. Диалект: Западный. Говор: Чистопольский. 47 нас. пункт.

13. Диалект: Западный. Говор: Мордва-каратаевский. 1 нас. пункт.

14. Диалект: Западный. Говор: Чистопольско-кряшенский. 2 нас. пункт.

15. Диалект: Западный. Говор: Стерлитамакский. 2 нас. пункт.

16. Диалект: Западный. Говор: Дрожжановский. 12 нас. пункт.

17. Диалект: Группа переходных говоров. Говор: Тарханский-камско-устьинский. 1 нас. пункт.

18. Диалект: Группа переходных говоров. Говор: Мамадышский-лаишский. 1 нас. пункт.

Здесь и далее приводятся количество населенных пунктов, которые попали в каждый кластер. Сами населенные пункты, которые попали в каждый кластер, можно найти в приложении.

Получение различных разбиений на кластеры осуществлялось с помощью подбора параметра repulsion функции качества.

Полученные результаты (repulsion = 2)

|  |  |
| --- | --- |
| Номер кластера | Количество населенных пунктов |
| 1 | 2 |
| 2 | 2 |
| 3 | 5 |
| 4 | 11 |
| 5 | 5 |
| 6 | 173 |
| 7 | 75 |

1-ый кластер полностью совпадает со Стерлитамакским говором. 2-ой кластер полностью совпадает с Подберезинским говором. 4-ый полностью совпадает с Нурлатским говором. 6-ой кластер состоит из говоров Среднего диалекта. 7-ой кластер состоит из говоров Западного диалекта.

Полученные результаты (repulsion = 3)

|  |  |
| --- | --- |
| Номер кластера | Количество населенных пунктов |
| 1 | 9 |
| 2 | 2 |
| 3 | 8 |
| 4 | 65 |
| 5 | 2 |
| 6 | 7 |
| 7 | 2 |
| 8 | 1 |
| 9 | 1 |
| 10 | 16 |
| 11 | 25 |
| 12 | 32 |
| 13 | 56 |
| 14 | 47 |

1-ый кластер совпадает с Нурлатским говором. 2-ой кластер совпадает со Стерлитамакским говором. 3-ий кластер состоит из населенных пунктов Мензилинского говора. 4-ый кластер включает населенные пункты Нижнекамско-кряшенского говора.

Вычисление разбиений с большим значением числа repulsion не проводилось, потому что даже при его значении 3 начали получаться вырожденные кластеры из одного населенного пункта.

Нужно отметить, в обоих разбиениях (при параметре repulsion 2 и 3) с экспертными данными совпали Нурлатский и Стерлиматский говоры. Это можно отнести к тому, что эти населенные пункты находятся близко к друг от друга и поэтому среди них существуют определенные лингвистические шаблоны.

## 3.4 Алгоритм кластеризации транзакционных данных ROCK

### Описание алгоритма

Алгоритм устойчивой кластеризации с иcпользованием связей ROCK (RObust Clustering using linKs) был совместно разработан Sudipto Guha, Rajeev Rastogi и Kyuseok Shim при работе в Bell Laboratories. Впервые алгоритм был опубликован в 1999 году в статье под названием "ROCK: A Robust Clustering Algorithm for Categorical Attributes".

Данный алгоритм принадлежит классу алгоритмов кластеризации, целью которых является разбиение данных на некоторое заранее заданное число групп. Основная особенность алгоритма ROCK заключается в использовании связей между точками (количество общих соседей), в отличие от методов, базирующихся на различных метриках, таких как расстояние между точками (Евклидово и прочие). Такой подход улучшает определение глобальных зависимостей, а также наиболее эффективен при рассмотрении данных, свойства которых принимают достаточно малое конечное количество значений.

Одним из основных понятий для алгоритма ROCK является соседство двух точек. Пусть нам дана функция схожести sim(p_i,p_j) , принимающая значения от 0 до 1, которая выражает схожесть или близость объектов(точек) _i и _j. Предполагается, что 1 соответствует абсолютной близости, и 0 - наоборот. Тогда при некоторой границе theta между 0 и 1, если sim(p_i,p_j) \geq \theta , то _i и _j будут соседними точками. Выбор Функции sim(p_i,p_j)  и граничного значения theta зависит входных данных и особенности реализации.

Вторым ключевым понятием являются связи. Функция связи link(p_i,p_j)  определяется как количество общих соседей у _i и _j. Из такого определения сразу видно, что чем больше значение связи, тем больше вероятность, что эти точки принадлежат одному и тому же кластеру. Такой подход является более глобальным, по сравнению с использованием только близости двух точек, что позволяет снизить число ошибок, особенно в тех случаях, когда кластеры имеют несколько близких точек.

Алгоритм состоит из двух основных этапов. Изначально имеется https://algowiki-project.org/w/ru/images/math/7/b/8/7b8b965ad4bca0e41ab51de7b31363a1.png точек и https://algowiki-project.org/w/ru/images/math/8/c/e/8ce4b16b22b58894aa86c421e8759df3.png - желаемое число кластеров. На первом этапе вычисляются значения связей link(p_i,p_j)  между всеми парами точек, каждая точка объявляется отдельным кластером. Для каждого кластера https://algowiki-project.org/w/ru/images/math/8/6/5/865c0c0b4ab0e063e5caa3387c1a8741.png создается локальная куча [i], которая содержит все такие кластеры https://algowiki-project.org/w/ru/images/math/3/6/3/363b122c528f54df4a0446b6bab05515.png, что связь между ними не нулевая. Кроме этого, создается глобальная куча https://algowiki-project.org/w/ru/images/math/f/0/9/f09564c9ca56850d4cd6b3319e541aee.png, содержащая все кластеры. После этого алгоритм переходит ко второму этапу. Вторая часть представляет из себя цикл, на каждом шаге которого объединяются два кластера с максимальным значением функции полезности (i,j), после чего вносятся соответствующие изменения в кучи. Алгоритм завершает работу в двух случаях: когда осталось https://algowiki-project.org/w/ru/images/math/8/c/e/8ce4b16b22b58894aa86c421e8759df3.png кластеров, или когда все связи между оставшимися кластерами равны нулю.

Дано множество S  из n  элементов и число k . Для каждых p_i,p_j \in S  задана функция схожести 0 \leq sim(p_i,p_j) \leq 1. Также дано число 0 \leq \theta \leq 1. Дана функция f( \theta ) .

Требуется разбить S  на k  не пересекающихся подмножеств(кластеров) _1, \dots, C_k так, чтобы значение целевой функции _lбыло как можно большим.

Определим следующие функции, полагая, что p_i,p_j \in S. 

Функция, определяющая соседство двух точек:

\begin{ali

Функция, определяющая количество связей между точками:

link(p_i, 

Определим теперь целевую функцию, считая _i количеством элементов в _i:

E_l = \sum

Определим функцию связи для двух подмножеств C_i, C_j :

link[C_i, 

Введем функцию полезности, выражающую то, насколько выгодно объединить подмножества C_i, C_j :

g(C_i, C_j

В данном алгоритме сначала все элементы разбиваются на n  подмножеств, затем делается n - k  шагов, на каждом из которых объединяются два подмножества, для которых значение функции полезности g(i,j)  наибольшее.

Вычислительное ядро алгоритма состоит из двух основных частей.

1. Вычисление всех связей между точками. Для каждой точки создается список соседей. Проходим по каждому такому списку и увеличиваем количество связей между всеми точками из списка, поскольку они имеют общую соседнюю точку (ту, которой принадлежит список).

2. Цикл по объединению кластеров, содержит n-k  шагов, на каждом из которых мы находим наилучшие кластеры для объединения и перестраиваем соответствующие кучи.

Вычисление связей может рассматриваться как перемножение двух матриц размера n , что можно реализовать за O(n^{2.37}). Затраты памяти на хранение связей не превышают n(n+1)/2 , когда любая пара точек - соседние. Однако в большинстве случаев среднее число соседей m_a  и максимальное число соседей m_m  значительно меньше n , в связи с этим оценки сложности зависят от данных параметров. Сложность построения списка соседей оценивается O(n^2). Для каждой точки, после вычисления списка своих соседей, алгоритм рассматривает все пары его соседей. Для каждой пары точка вносит одну связь. Если процесс повторяется для каждой точки и счетчик ссылок увеличивается на единицу для каждой пары соседей, то в конце концов, ссылка рассчитывает на будут получены все пары точек. Если _i - число соседей точки https://algowiki-project.org/w/ru/images/math/8/6/5/865c0c0b4ab0e063e5caa3387c1a8741.png, тогда для нее мы должны увеличить количество связей на единицу _i^2 раз. Таким образом, сложность алгоритма является sum m_i^2 и может быть оценено O(n m_m m_a). Поскольку каждая точка https://algowiki-project.org/w/ru/images/math/8/6/5/865c0c0b4ab0e063e5caa3387c1a8741.png может иметь не более min \{  m_m m_i, n \} ) связей, то общие затраты памяти не превышают O(\min \{  n m_m m_a, n^2 \} ).

Построение каждой кучи потребует не более O(n) времени (поскольку размер каждой кучи не может превышать n ) Сортировка каждой кучи потребует O( n \log{} n). Рассмотрим теперь основной while-цикл. Он содержит O(n) итераций, где основная сложность каждого шага приходится на for-цикл. В худшем случае потребуется вставить новый кластер в O(n) локальных куч размера n, что потребует O( n \log{} n) времени. Таким образом, сложность всего внешнего цикла составляет O(n^2 \log{} n) в худшем случае. Затраты памяти зависят от начального размера локальных куч, поскольку при слиянии кластеров их старые кучи удаляются, а размер новой не превышает суммы старых. Поскольку каждая куча содержит лишь кластеры с ненулевыми связями, то сложность совпадает с первой частью и равна O(\min \{  n m_m m_a, n^2 \} ).

Итоговые оценки для всего алгоритма:

**Сложность по времени**:

O(n^2 + n m_m m_a + n^2 \log{} n

**Сложность по памяти**:

O(\min \{ n^2, n m_m m_a \} )

**Список обозначений**:

m_a - среднее число соседей

m_m - максимальное число соседей

_i- число соседей точки https://algowiki-project.org/w/ru/images/math/8/6/5/865c0c0b4ab0e063e5caa3387c1a8741.png

https://algowiki-project.org/w/ru/images/math/7/b/8/7b8b965ad4bca0e41ab51de7b31363a1.png- размер входных данных

### Реализация алгоритма

Данный алгоритм был реализован на функциональном языке программирования Elixir по оригинальной статье авторов алгоритма. Были написаны тесты на основе примеров, описываемых в статье.

Алгоритм имеет следующий интерфейс.

Входные параметры.

* Points – массив кортежей. Каждый кортеж состоит из названия транзакции и списка ее атрибутов.
* Number\_of\_clusters – количество кластеров, которое необходимо получить на выходе
* Theta – граница, по которой определяется соседство двух транзакций. По умолчанию, значение границы – 0.5.
* Similarity\_function – функция расстояния, необходимая для определения являются 2 транзакции соседями. По умолчанию, используется Коэффициент Жаккара.

Результат работы алгоритмы:

* Clusters – список, каждый элемент которого представляет кластер.

Пример вызова алгоритма:

points =

[

{"point1", ["1", "2", "3"]},

{"point2", ["1", "2", "4"]},

{"point3", ["1", "2", "5"]},

{"point4", ["1", "3", "4"]},

{"point5", ["1", "3", "5"]},

{"point6", ["1", "4", "5"]},

{"point7", ["2", "3", "4"]},

{"point8", ["2", "3", "5"]},

{"point9", ["2", "4", "5"]},

{"point10", ["3", "4", "5"]},

{"point11", ["1", "2", "6"]},

{"point12", ["1", "2", "7"]},

{"point13", ["1", "6", "7"]},

{"point14", ["2", "6", "7"]}

]

Rock.clusterize(points, 5, 0.4)

[

[

{"point4", ["1", "3", "4"]},

{"point5", ["1", "3", "5"]},

{"point6", ["1", "4", "5"]},

{"point10", ["3", "4", "5"]},

{"point7", ["2", "3", "4"]},

{"point8", ["2", "3", "5"]}

],

[

{"point11", ["1", "2", "6"]},

{"point12", ["1", "2", "7"]},

{"point1", ["1", "2", "3"]},

{"point2", ["1", "2", "4"]},

{"point3", ["1", "2", "5"]}

],

[

{"point9", ["2", "4", "5"]}

],

[

{"point13", ["1", "6", "7"]}

],

[

{"point14", ["2", "6", "7"]}

]

]

Реализованный алгоритм можно найти по адресу <https://github.com/ayrat555/rock>. Также библиотека с алгоритмом опубликована в системе управления пакетами экосистемы Erlang - <https://hex.pm/packages/rock>. В качестве сервиса непрерывной интеграции использовался <https://semaphoreci.com>.

### Результаты.

При кластеризации исходных данных с помощью алгоритма ROCK для вычисления соседей использовался коэффициент Жаккара. Вычисление кластеров проводилось с различными параметрами Theta и Number\_of\_clusters: Number\_of\_clusters менялось от 2 до 20 с шагом 1, Theta от 0.1 до 0.5 с шагом 0.05. Все полученные кластеры можно найти по адресу <https://github.com/ayrat555/masters_work/tree/master/data/output/rock_clusters>. Полное содержание описываемых ниже разбиений можно найти в приложении.

Определение диалектов.

В качестве параметров кластеризации выбраны: количество кластеров = 2, параметр theta = 0.2. При большем значении theta один из кластеров получался вырожденным.

Полученные кластеры:

1. 127 населенных пункта
2. 145 населенных пункта

В первый кластер попали все населенные пункты западного диалекта, за исключением 9. Однако значительное число населенных пунктов среднего диалекта также попало в первый кластер. Второй кластер состоит из населенных пунктов среднего диалекта. Процент населённых пунктов, которые совпадают с диалектами экспертного разбиения составляет около 75 %.

Определение говоров.

Аналогично определению диалектов оптимальным значением theta оказалось 0.2. Далее описываются разбиения с параметром Number\_of\_clusters 7 и 14.

Полученные результаты (Number\_of\_clusters = 7)

|  |  |
| --- | --- |
| Номер кластера | Номер кластера |
| 1 | 64 |
| 2 | 49 |
| 3 | 32 |
| 4 | 32 |
| 5 | 32 |
| 6 | 32 |
| 7 | 32 |

Первый кластер состоит из населенных пунктов Чистопольского и Мамадышского говоров. Второй кластер состоит из населенных пунктов Тархановского и Дрожжановского говоров. Третий кластер состоит из населенных пунктов Нурлатского и Мензелинского говоров. Четвертый кластер состоит из насленных рунктов Балтачевского говора. Пятый кластер состоит из населенных Нижнекаско-Кряшенского говора.

Полученные результаты (Number\_of\_clusters = 14)

|  |  |
| --- | --- |
| Номер кластера | Номер кластера |
| 1 | 32 |
| 2 | 32 |
| 3 | 32 |
| 4 | 17 |
| 5 | 16 |
| 6 | 16 |
| 7 | 16 |
| 8 | 16 |
| 9 | 16 |
| 10 | 16 |
| 11 | 16 |
| 12 | 16 |
| 13 | 16 |
| 14 | 16 |

Первый кластер и второй кластер состоят из населенных пунктов Мензелинского говора. Третий кластер состоит из населенных пунктов Нижнекаско-Кряшенского говора. Четвертый кластер состоит из населенных пунктов Мамадышского говора. Пятый кластер состоит из населенных пунктов Нурлатского говора. Шестой кластер состоит из населенных пунктов Лаишевского говора, седьмой – балтачевского говора, восьмой и девятый – чистопольского говора, двенадцатый – тархановского. Десятый, тринадцатый и четырнадцатый кластер не удалось поставить в соответствие экспертному разбиению.

Из количества кластеров в обоих разбиениях видно, что на каждом шаге алгоритма кластеры объединяются между собой, причем это происходит последовательно. Такая ситуация возникает из-за того, что у каждого населенного пункта большое число атрибутов и поэтому практически все населенные пункты являются соседями в терминах алгоритма ROCK, однако ROCK объединяет на каждом шаге кластеры с максимальным числом ссылок, поэтому населенные пункты в кластерах более похожи между собой,  чем с населенными пунктами из других кластеров.

## 3.5 Алгоритм k-медоидов с метрикой на основе расстояния Ливенштейна

В качестве еще одного метода кластеризации населенных пунктов использовался алгоритм k-медоидов c метрикой на основе расстояния Левенштейна. Подобный подход использовали Джон Нербон (John Nerbonne) и Елена Прокич (Jelena Prokić) в своей работе по кластеризации насленных пунктов Болгарии на говоры.

Однако они обладали полными фонетичексими транскрипциями слов, используемых в этих регионах. В наших же данных нет слов, а есть только морфологические признаки - ряд грамматических категорий, которые присущи словам данной части речи. Поэтому использование расстояния Левенштейна для наших данных не совсем корректно, такое заключение было сделано на позднем этапе и все-таки кластеры диалектов были получены. Результаты оказались удовлетворительными, поэтому было решено включить их в работу. Кластеризация на говоры c использованием данного подхода не проводилась.

### Описание алгоритма

Алгоритм k-медоидов является модификацией алгоритма k-средних. Далее приведено описание алгоритма k-средних.

В отличие от иерархических методов, которые не требуют предварительных предположений относительно числа кластеров, для возможности использования этого метода необходимо иметь гипотезу о наиболее вероятном количестве кластеров.

Алгоритм k-средних строит k кластеров, расположенных на возможно больших расстояниях друг от друга. Основной тип задач, которые решает алгоритм k-средних, - наличие предположений (гипотез) относительно числа кластеров, при этом они должны быть различны настолько, насколько это возможно. Выбор числа k может базироваться на результатах предшествующих исследований, теоретических соображениях или интуиции.

Общая идея алгоритма: заданное фиксированное число k кластеров наблюдения сопоставляются кластерам так, что средние в кластере (для всех переменных) максимально возможно отличаются друг от друга.

#### Шаги алгоритма

1. Первоначальное распределение объектов по кластерам.

Выбирается число k, и на первом шаге эти точки считаются "центрами" кластеров. Каждому кластеру соответствует один центр.

Выбор начальных центроидов может осуществляться следующим образом:

* + выбор k-наблюдений для максимизации начального расстояния;
  + случайный выбор k-наблюдений;
  + выбор первых k-наблюдений.

В результате каждый объект назначен определенному кластеру.

1. Итеративный процесс.

Вычисляются центры кластеров, которыми затем и далее считаются покоординатные средние кластеров. Объекты опять перераспределяются.

Процесс вычисления центров и перераспределения объектов продолжается до тех пор, пока не выполнено одно из условий:

* + кластерные центры стабилизировались, т.е. все наблюдения принадлежат кластеру, которому принадлежали до текущей итерации;
  + число итераций равно максимальному числу итераций.

После получений результатов кластерного анализа методом k-средних следует проверить правильность кластеризации (т.е. оценить, насколько кластеры отличаются друг от друга). Для этого рассчитываются средние значения для каждого кластера. При хорошей кластеризации должны быть получены сильно отличающиеся средние для всех измерений или хотя бы большей их части.

Достоинства алгоритма k-средних:

* простота использования;
* быстрота использования;
* понятность и прозрачность алгоритма.

Недостатки алгоритма k-средних:

* алгоритм слишком чувствителен к выбросам, которые могут искажать среднее. Возможным решением этой проблемы является использование модификации алгоритма - алгоритм k-медианы;
* алгоритм может медленно работать на больших базах данных. Возможным решением данной проблемы является использование выборки данных.

В отличие от k-средних, в k-медоидах в качестве центроидов может выступать не любая точка, а только какие-то из имеющихся наблюдений. Алгоритм менее чувствителен к шумам и выбросам данных, чем алгоритм k-средних, поскольку медиана меньше подвержена влияниям выбросов.

### Реализация

Было написано приложение на функциональном языке программирования Elixir, которое обрабатывает входные данных с морфологическими атрибутами населенных пунктов, составляет матрицу расстояний между населенными пунктами, считая расстояние как нормализованное среднее между значениями расстояний Левенштейна соотсветствующих атрибутов.

Также был написан скрипт на языке R, который проводит саму кластеризация, используя входную матрицу расстояний и необходиоме количество кластеров.

Приложение и скрипт доступны по адресу https://github.com/ayrat555/masters\_work.

### Результаты

Определение диалектов

С помощью данного алгоритма были полученые кластеры:

1. 217 населенных пунктов
2. 56 населенных пунктов

Во второй кластер входят только населенные пункты западного диалекта, однако в нем не хватает десяти населенных пуктов для полного совпадения, они попали в первый кластер. Первый кластер состоит из населенных пунктов среднего диалекта. Переходный диалект определить не удалось, населенные пункты из этого диалекта попали в первый кластер.

В целом, процент совпавших насленных пунктов с экспертными данными составляет 95%.

## 3.6 Сравнение результатов результатов

В целом

# Глава 4. Результаты работы

# Заключение

# 

# Приложение 1. Алгоритм CLOPE: Полученные кластеры

Параметр repulsion = 1

|  |  |
| --- | --- |
| Кластер 1 | Кугарчино, Янга-Сала, Татарское Ходяшево, Кибячи, Большие Ачасыры, Бурундуки, Молвино, Сатламышево, Кугеево, Мамадыш-Акилово, Большие Кушманы, Акзигитово, Старые Чечкабы, Старые Бугады, Тюгеевка, Урмышла, Ильтень-Бута, Чубуклы, Сарапала, Маметьево, Савалеево, Светлое Озеро, Калейкино, Васильевка, Сармашбаш, Абдрахманово, Верхние Шипки, Старое Утяганово, Девятерня, Исенбаево, Ямурзино, Крынды, Янга-Аул, Биктово, Табарле, Тойгузино, Камаево, Татарская Тумбарла, Новое Алимово, Кзыл-Яр, Сарабикулово, Федотовка, Акбаш, Дым-Тамак, Нижние Чершилы, Иж-Бобья, Янчиково, Старый Юмралы, Кряш-Шуран, Старое Шугурово, Сугушла, Татар. Шуган, Старый Каразерик, Салауз-Мухан, Учалле, Нижний Табын, Баюково, Старые Карамалы, Байряки-Тамак, Верхнее Стярле, Сарлы, Каракашлы, Карамалы, Асеево, Большой Сухояш, Тойгильдино, Балтачево, Масягутово, Кавзияково, Мальбагуш, Иляксаз, Сулеево, Бишмунча, Новый Имян, Старое Байсарово, Старое Сафарово, Татарские Суксы, Старое Курмашево, Атясево, Исансупово, Семяково, Старый Иркеняш, Аю, Ст.Мелькен, Верхние Юшады, Сарсаз Такермень, Ляки, Средний Багряж, Кабан-Бастрык, Кадырово, Бурды, Средние Пинячи, Мелекес, Верхний Такермен, Калмия, Мунайка, Рысово, Верхнее Абдулово, Нижние Гордалы, Штырь, Псеево, Бизяки, Старое Гришкино, Абалачи, Морты, Большой Шурняк, Мокрые Курнали, Альвидино, Большой Салтан, Большой Ошняк, Татарский Янтык, Большая Елга, Ямашурма, Шали, Пелево, Именьково, Кирби, Курманаково, Татарские Саралы, Татарский Кабан, Нижний Искубаш, Кугунур, Арбор, Нуринер, Верхний Субаш, Янгулово, Мамашир, Бурбаш, Норма, Карадуван, Туембаш, Байлянгар, Сиза, Верхняя Корса, Измя, Евлаштау, Псяк, Урясьбаш, Кильдебяк, Старый Мичан, Большие Верези, Сардебаш, Апазово, Средний Пшалым, Мендюш, Качелино, Байкал, Ашитбаш, Утар-Аты, Большой Менгер, Ключи-Сап, Кубян, Коморгузя, Ташлы-Ковали, Ислейтар, Ст. Баран//Иске Рязап, Новый Баран, Березняк, Тавель, Татарское Бурнаево, Ямбухтино, Измери, Ташкирмень, Средний Юрткуль, Большие Кармалы, Азимово-Курлебаш, Большие Янгасала, Балтачево, Большие Болгояры, Большие Атряси, Большая Турма, Старое Барышево, Чуру-Барышево, Татарская Беденьга, Большая Карланга, Большой Сардек, Шамбулыхчы, Чишма, Деушево, Черки-Кильдуразы, Сюндюково, Бакрче, Бик-Утеево, Верхние Лащи, Нижний Шандер, Куюк Ерыкса, Крещеный Пакшин, Верхняя Ошма, Нижний Таканыш, Средние Кирмени, Уразбахты// Татарский Омар, Тяберлины Челны, Янцевары, Большие Кибя-Кози, Кукча, Ишкеево, Верхняя Сунь, Нижняя Сунь, Ятмас Дусай, Татарская Икшурма, Албай, Тюбяк, Шеморбаш, Козяково-Челны, Ядыгерь, Сатышево, Баландыш, Субаш, Новый Арыш |
| Кластер 2 | Салихово, Новые Чути, Янга-Аул, Ерыклинский, Татарские Шатрашаны, Старые Чукалы, Большая Цильна, Старая Задоровка, Мочалей, Старое Шаймурзино, Новые Ишли, Татарские Тюки, Энтуганы, Чутеево, Хозесаново, Старое Тябердино, Малый Сулабаш, Альдермыш, Большой Битаман, Большие Ковали, Большие Кургузи, Бикулово, Нижний Нурлат, Савиново, Кривое Озеро, Курманаево, Нижняя Каменка, Лашманка, Татар. Волчье, Нов. Кадеево, Шахмайкино, Старое Ибрайкино, Старые Киязлы, Исляйкино, Средние Тиганы, Старое Алпарово, Старое Ямкино, Сухие Курнали, Новые Челны, Нижнее Алькеево, Татарское Ахметьево, Каргополь, Тяжбердино, Старые Салманы, Татарское Тюгульбаево, Татарская Тахтала, Мордовский Каратай, Старый Студенец, Старые Тинчали, Новое Узеево, Тубулга, Татарский Елтан, Старая Киреметь, Новое Демкино, Сунчелеево, Муслюмкино, Акбулатово, Ст.Татар. Адам, Верхняя Татарская Майна, Новая Амзя, Степная Шентала, Кутлушкино, Кульбаево-Мараса, Кичкальня |

Параметр repulsion = 2

|  |  |
| --- | --- |
| Кластер 1 | Салихово, Новые Чути |
| Кластер 2 | Хозесаново, Старое Тябердино |
| Кластер 3 | Старый Юмралы, Татарские Саралы, Ташкирмень, Мордовский Каратай, Азимово-Курлебаш |
| Кластер 4 | Большие Ачасыры, Бурундуки, Молвино, Сатламышево, Кугеево, Мамадыш-Акилово, Большие Кушманы, Акзигитово, Старые Чечкабы, Шамбулыхчы, Бакрче |
| Кластер 5 | Большой Битаман, Тавель, Нижний Шандер, Шеморбаш, Козяково-Челны |
| Кластер 6 | Кугарчино, Янга-Сала, Татарское Ходяшево, Кибячи, Старые Бугады, Тюгеевка, Урмышла, Ильтень-Бута, Чубуклы, Сарапала, Маметьево, Савалеево, Светлое Озеро, Калейкино, Васильевка, Сармашбаш, Абдрахманово, Верхние Шипки, Старое Утяганово, Девятерня, Исенбаево, Ямурзино, Крынды, Янга-Аул, Биктово, Табарле, Тойгузино, Камаево, Татарская Тумбарла, Новое Алимово, Кзыл-Яр, Сарабикулово, Федотовка, Акбаш, Дым-Тамак, Нижние Чершилы, Иж-Бобья, Янчиково, Кряш-Шуран, Старое Шугурово, Сугушла, Татар. Шуган, Старый Каразерик, Салауз-Мухан, Учалле, Нижний Табын, Баюково, Старые Карамалы, Байряки-Тамак, Верхнее Стярле, Сарлы, Каракашлы, Карамалы, Асеево, Большой Сухояш, Тойгильдино, Балтачево, Масягутово, Кавзияково, Мальбагуш, Иляксаз, Сулеево, Бишмунча, Новый Имян, Старое Байсарово, Старое Сафарово, Татарские Суксы, Старое Курмашево, Атясево, Исансупово, Семяково, Старый Иркеняш, Аю, Ст.Мелькен, Верхние Юшады, Сарсаз Такермень, Ляки, Средний Багряж, Кабан-Бастрык, Кадырово, Бурды, Средние Пинячи, Мелекес, Верхний Такермен, Калмия, Мунайка, Рысово, Верхнее Абдулово, Янга-Аул, Нижние Гордалы, Штырь, Псеево, Бизяки, Старое Гришкино, Абалачи, Морты, Большой Шурняк, Альвидино, Большой Салтан, Большой Ошняк, Татарский Янтык, Большая Елга, Ямашурма, Шали, Пелево, Именьково, Кирби, Курманаково, Татарский Кабан, Нижний Искубаш, Кугунур, Арбор, Нуринер, Верхний Субаш, Янгулово, Мамашир, Бурбаш, Норма, Карадуван, Туембаш, Байлянгар, Сиза, Верхняя Корса, Измя, Евлаштау, Псяк, Урясьбаш, Кильдебяк, Старый Мичан, Большие Верези, Сардебаш, Апазово, Средний Пшалым, Мендюш, Качелино, Байкал, Ашитбаш, Утар-Аты, Большой Менгер, Ключи-Сап, Кубян, Коморгузя, Ташлы-Ковали, Ислейтар, Малый Сулабаш, Альдермыш, Большие Ковали, Большие Кургузи, Березняк, Большой Сардек, Чишма, Куюк Ерыкса, Крещеный Пакшин, Верхняя Ошма, Нижний Таканыш, Средние Кирмени, Уразбахты// Татарский Омар, Тяберлины Челны, Янцевары, Большие Кибя-Кози, Кукча, Ишкеево, Верхняя Сунь, Нижняя Сунь, Ятмас Дусай, Татарская Икшурма, Албай, Тюбяк, Ядыгерь, Сатышево, Баландыш, Субаш, Новый Арыш |
| Кластер 7 | Ерыклинский, Мокрые Курнали, Татарские Шатрашаны, Старые Чукалы, Большая Цильна, Старая Задоровка, Мочалей, Старое Шаймурзино, Новые Ишли, Татарские Тюки, Энтуганы, Чутеево, Бикулово, Нижний Нурлат, Савиново, Кривое Озеро, Курманаево, Ст. Баран//Иске Рязап, Новый Баран, Нижняя Каменка, Лашманка, Татар. Волчье, Нов. Кадеево, Шахмайкино, Старое Ибрайкино, Старые Киязлы, Исляйкино, Средние Тиганы, Старое Алпарово, Старое Ямкино, Сухие Курнали, Новые Челны, Нижнее Алькеево, Татарское Ахметьево, Каргополь, Тяжбердино, Татарское Бурнаево, Ямбухтино, Старые Салманы, Татарское Тюгульбаево, Татарская Тахтала, Измери, Средний Юрткуль, Большие Кармалы, Большие Янгасала, Балтачево, Большие Болгояры, Большие Атряси, Большая Турма, Старое Барышево, Чуру-Барышево, Татарская Беденьга, Большая Карланга, Деушево, Черки-Кильдуразы, Старый Студенец, Сюндюково, Старые Тинчали, Бик-Утеево, Верхние Лащи, Новое Узеево, Тубулга, Татарский Елтан, Старая Киреметь, Новое Демкино, Сунчелеево, Муслюмкино, Акбулатово, Ст.Татар. Адам, Верхняя Татарская Майна, Новая Амзя, Степная Шентала, Кутлушкино, Кульбаево-Мараса, Кичкальня |

Параметр repulsion = 3

|  |  |
| --- | --- |
| Кластер 1 | Большие Ачасыры, Бурундуки, Молвино, Сатламышево, Кугеево, Мамадыш-Акилово, Большие Кушманы, Акзигитово, Старые Чечкабы |
| Кластер 2 | Салихово, Новые Чути |
| Кластер 3 | Татарская Тумбарла, Кзыл-Яр, Сарабикулово, Федотовка, Акбаш, Дым-Тамак, Нижние Чершилы, Сугушла |
| Кластер 4 | Старые Бугады, Тюгеевка, Урмышла, Ильтень-Бута, Чубуклы, Сарапала, Маметьево, Савалеево, Светлое Озеро, Калейкино, Васильевка, Сармашбаш, Абдрахманово, Верхние Шипки, Старое Утяганово, Ямурзино, Новое Алимово, Кряш-Шуран, Татар. Шуган, Старый Каразерик, Салауз-Мухан, Учалле, Нижний Табын, Баюково, Старые Карамалы, Байряки-Тамак, Верхнее Стярле, Сарлы, Каракашлы, Карамалы, Асеево, Большой Сухояш, Тойгильдино, Балтачево, Масягутово, Кавзияково, Иляксаз, Сулеево, Бишмунча, Новый Имян, Старое Байсарово, Старое Сафарово, Татарские Суксы, Старое Курмашево, Атясево, Исансупово, Семяково, Старый Иркеняш, Аю, Ст.Мелькен, Верхние Юшады, Сарсаз Такермень, Ляки, Средний Багряж, Кабан-Бастрык, Кадырово, Бурды, Средние Пинячи, Мелекес, Верхний Такермен, Калмия, Верхнее Абдулово, Нижние Гордалы, Морты, Большой Шурняк |
| Кластер 5 | Мальбагуш, Мокрые Курнали |
| Кластер 6 | Татарские Шатрашаны, Старые Чукалы, Большая Цильна, Мочалей, Старое Шаймурзино, Новые Ишли, Татарские Тюки |
| Кластер 7 | Хозесаново, Старое Тябердино |
| Кластер 8 | Тавель |
| Кластер 9 | Мордовский Каратай |
| Кластер 10 | Девятерня, Исенбаево, Крынды, Янга-Аул, Биктово, Табарле, Тойгузино, Камаево, Иж-Бобья, Мунайка, Рысово, Псеево, Бизяки, Старое Гришкино, Абалачи, Чишма |
| Кластер 11 | Старый Юмралы, Ст. Баран//Иске Рязап, Новый Баран, Татарское Бурнаево, Старые Салманы, Измери, Средний Юрткуль, Большие Кармалы, Азимово-Курлебаш, Большие Янгасала, Балтачево, Большие Болгояры, Большие Атряси, Большая Турма, Старое Барышево, Чуру-Барышево, Татарская Беденьга, Большая Карланга, Шамбулыхчы, Деушево, Черки-Кильдуразы, Сюндюково, Бакрче, Бик-Утеево, Верхние Лащи |
| Кластер 12 | Янга-Сала, Янга-Аул, Ямашурма, Кугунур, Арбор, Нуринер, Верхний Субаш, Янгулово, Мамашир, Бурбаш, Карадуван, Туембаш, Байлянгар, Большие Верези, Сардебаш, Апазово, Средний Пшалым, Байкал, Ашитбаш, Большой Менгер, Ключи-Сап, Кубян, Коморгузя, Ташлы-Ковали, Ислейтар, Малый Сулабаш, Альдермыш, Большой Битаман, Большие Ковали, Большие Кургузи, Большой Сардек, Ядыгерь |
| Кластер 13 | Кугарчино, Татарское Ходяшево, Кибячи, Янчиково, Штырь, Альвидино, Большой Салтан, Большой Ошняк, Татарский Янтык, Большая Елга, Шали, Пелево, Именьково, Кирби, Курманаково, Татарские Саралы, Татарский Кабан, Нижний Искубаш, Норма, Сиза, Верхняя Корса, Измя, Евлаштау, Псяк, Урясьбаш, Кильдебяк, Старый Мичан, Мендюш, Качелино, Утар-Аты, Березняк, Ташкирмень, Нижний Шандер, Куюк Ерыкса, Крещеный Пакшин, Верхняя Ошма, Нижний Таканыш, Средние Кирмени, Уразбахты// Татарский Омар, Тяберлины Челны, Янцевары, Большие Кибя-Кози, Кукча, Ишкеево, Верхняя Сунь, Нижняя Сунь, Ятмас Дусай, Татарская Икшурма, Албай, Тюбяк, Шеморбаш, Козяково-Челны, Сатышево, Баландыш, Субаш, Новый Арыш |
| Кластер 14 | Старое Шугурово, Ерыклинский, Старая Задоровка, Энтуганы, Чутеево, Бикулово, Нижний Нурлат, Савиново, Кривое Озеро, Курманаево, Нижняя Каменка, Лашманка, Татар. Волчье, Нов. Кадеево, Шахмайкино, Старое Ибрайкино, Старые Киязлы, Исляйкино, Средние Тиганы, Старое Алпарово, Старое Ямкино, Сухие Курнали, Новые Челны, Нижнее Алькеево, Татарское Ахметьево, Каргополь, Тяжбердино, Ямбухтино, Татарское Тюгульбаево, Татарская Тахтала, Старый Студенец, Старые Тинчали, Новое Узеево, Тубулга, Татарский Елтан, Старая Киреметь, Новое Демкино, Сунчелеево, Муслюмкино, Акбулатово, Ст.Татар. Адам, Верхняя Татарская Майна, Новая Амзя, Степная Шентала, Кутлушкино, Кульбаево-Мараса, Кичкальня |

# Приложение 2. Алгоритм ROCK: Полученные кластеры

Number\_of\_clusters = 2, Theta = 0.2

|  |  |
| --- | --- |
| Кластер 1 | Сугушла, Татар. Шуган, Кряш-Шуран, Старое Шугурово, Учалле, Нижний Табын, Старый Каразерик, Салауз-Мухан, Акбаш, Дым-Тамак, Сарабикулово, Федотовка, Янчиково, Старый Юмралы, Нижние Чершилы, Иж-Бобья, Балтачево, Масягутово, Большой Сухояш, Тойгильдино, Иляксаз, Сулеево, Кавзияково, Мальбагуш, Байряки-Тамак, Верхнее Стярле, Баюково, Старые Карамалы, Карамалы, Асеево, Сарлы, Каракашлы, Маметьево, Савалеево, Чубуклы, Сарапала, Сармашбаш, Абдрахманово, Светлое Озеро, Васильевка, Большие Кушманы, Акзигитово, Кугеево, Мамадыш-Акилово, Тюгеевка, Урмышла, Старые Чечкабы, Старые Бугады, Камаево, Салихово, Табарле, Тойгузино, Новое Алимово, Кзыл-Яр, Новые Чути, Татарская Тумбарла, Девятерня, Исенбаево, Верхние Шипки, Старое Утяганово, Янга-Аул, Биктово, Ямурзино, Крынды, Пелево, Именьково, Ямашурма, Шали, Татарские Саралы, Татарский Кабан, Кирби, Курманаково, Мокрые Курнали, Альвидино, Морты, Большой Шурняк, Татарский Янтык, Большая Елга, Большой Салтан, Большой Ошняк, Бурбаш, Норма, Верхний Субаш, Янгулово, Байлянгар, Сиза, Карадуван, Туембаш, Чутеево, Хозесаново, Старая Задоровка, Энтуганы, Арбор, Нуринер, Старое Тябердино, Кугунур, Аю, Ст.Мелькен, Семяково, Старый Иркеняш, Ляки, Средний Багряж, Верхние Юшады, Сарсаз Такермень, Старое Байсарово, Старое Сафарово, Бишмунча, Новый Имян, Атясево, Исансупово, Татарские Суксы, Старое Курмашево, Ерыклинский, Нижние Гордалы, Верхнее Абдулово, Янга-Аул, Бизяки, Старое Гришкино, Штырь, Псеево, Средние Пинячи, Мелекес, Кабан-Бастрык, Кадырово, Мунайка, Рысово, Верхний Такермен, Калмия |
| Кластер 2 | Черки-Кильдуразы, Сюндюково, Чишма, Деушево, Бик-Утеево, Верхние Лащи, Старые Тинчали, Бакрче, Старое Барышево, Чуру-Барышево, Большие Атряси, Большая Турма, Большой Сардек, Шамбулыхчы, Татарская Беденьга, Большая Карланга, Нижняя Сунь, Ятмас Дусай, Ишкеево, Верхняя Сунь, Шеморбаш, Козяково-Челны, Албай, Тюбяк, Нижний Таканыш, Средние Кирмени, Крещеный Пакшин, Верхняя Ошма, Янцевары, Кукча, Уразбахты// Татарский Омар, Тяберлины Челны, Большие Кибя-Кози, Татарская Икшурма, Нижний Шандер, Старые Чукалы, Мочалей, Псяк, Куюк Ерыкса, Ильтень-Бута, Нижний Искубаш, Татарские Шатрашаны, Муслюмкино, Новые Ишли, Татарские Тюки, Старое Шаймурзино, Ст.Татар. Адам, Большая Цильна, Сухие Курнали, Мамашир, Большой Менгер, Бурды, Абалачи, Старый Студенец, Татарский Елтан, Калейкино, Кичкальня, Татарское Ходяшево, Кибячи, Кугарчино, Янга-Сала, Молвино, Сатламышево, Большие Ачасыры, Бурундуки, Акбулатово, Верхняя Татарская Майна, Новое Демкино, Сунчелеево, Кутлушкино, Кульбаево-Мараса, Новая Амзя, Степная Шентала, Баландыш, Субаш, Ядыгерь, Сатышево, Тубулга, Старая Киреметь, Новый Арыш, Новое Узеево, Средние Тиганы, Старое Алпарово, Старые Киязлы, Исляйкино, Нижнее Алькеево, Татарское Ахметьево, Старое Ямкино, Новые Челны, Нижняя Каменка, Лашманка, Березняк, Тавель, Шахмайкино, Старое Ибрайкино, Татар. Волчье, Нов. Кадеево, Мордовский Каратай, Большие Кармалы, Ташкирмень, Средний Юрткуль, Балтачево, Большие Болгояры, Азимово-Курлебаш, Большие Янгасала, Татарское Бурнаево, Ямбухтино, Каргополь, Тяжбердино, Татарская Тахтала, Измери, Старые Салманы, Татарское Тюгульбаево, Мендюш, Качелино, Апазово, Средний Пшалым, Утар-Аты, Ключи-Сап, Байкал, Ашитбаш, Евлаштау, Урясьбаш, Верхняя Корса, Измя, Большие Верези, Сардебаш, Кильдебяк, Старый Мичан, Нижний Нурлат, Савиново, Большие Кургузи, Бикулово, Ст. Баран//Иске Рязап, Новый Баран, Кривое Озеро, Курманаево, Ташлы-Ковали, Ислейтар, Кубян, Коморгузя, Большой Битаман, Большие Ковали, Малый Сулабаш, Альдермыш |

Number\_of\_clusters = 7, Theta = 0.2

|  |  |
| --- | --- |
| Кластер 1 | Средние Тиганы, Старое Алпарово, Старые Киязлы, Исляйкино, Нижнее Алькеево, Татарское Ахметьево, Старое Ямкино, Новые Челны, Нижняя Каменка, Лашманка, Березняк, Тавель, Шахмайкино, Старое Ибрайкино, Татар. Волчье, Нов. Кадеево, Мордовский Каратай, Большие Кармалы, Ташкирмень, Средний Юрткуль, Балтачево, Большие Болгояры, Азимово-Курлебаш, Большие Янгасала, Татарское Бурнаево, Ямбухтино, Каргополь, Тяжбердино, Татарская Тахтала, Измери, Старые Салманы, Татарское Тюгульбаево, Мендюш, Качелино, Апазово, Средний Пшалым, Утар-Аты, Ключи-Сап, Байкал, Ашитбаш, Евлаштау, Урясьбаш, Верхняя Корса, Измя, Большие Верези, Сардебаш, Кильдебяк, Старый Мичан, Нижний Нурлат, Савиново, Большие Кургузи, Бикулово, Ст. Баран//Иске Рязап, Новый Баран, Кривое Озеро, Курманаево, Ташлы-Ковали, Ислейтар, Кубян, Коморгузя, Большой Битаман, Большие Ковали, Малый Сулабаш, Альдермыш |
| Кластер 2 | Черки-Кильдуразы, Сюндюково, Чишма, Деушево, Бик-Утеево, Верхние Лащи, Старые Тинчали, Бакрче, Старое Барышево, Чуру-Барышево, Большие Атряси, Большая Турма, Большой Сардек, Шамбулыхчы, Татарская Беденьга, Большая Карланга, Нижняя Сунь, Ятмас Дусай, Ишкеево, Верхняя Сунь, Шеморбаш, Козяково-Челны, Албай, Тюбяк, Нижний Таканыш, Средние Кирмени, Крещеный Пакшин, Верхняя Ошма, Янцевары, Кукча, Уразбахты// Татарский Омар, Тяберлины Челны, Большие Кибя-Кози, Татарская Икшурма, Нижний Шандер, Старые Чукалы, Мочалей, Псяк, Куюк Ерыкса, Ильтень-Бута, Нижний Искубаш, Татарские Шатрашаны, Муслюмкино, Новые Ишли, Татарские Тюки, Старое Шаймурзино, Ст.Татар. Адам, Большая Цильна, Сухие Курнали |
| Кластер 3 | Мамашир, Большой Менгер, Бурды, Абалачи, Старый Студенец, Татарский Елтан, Калейкино, Кичкальня, Татарское Ходяшево, Кибячи, Кугарчино, Янга-Сала, Молвино, Сатламышево, Большие Ачасыры, Бурундуки, Акбулатово, Верхняя Татарская Майна, Новое Демкино, Сунчелеево, Кутлушкино, Кульбаево-Мараса, Новая Амзя, Степная Шентала, Баландыш, Субаш, Ядыгерь, Сатышево, Тубулга, Старая Киреметь, Новый Арыш, Новое Узеево |
| Кластер 4 | Пелево, Именьково, Ямашурма, Шали, Татарские Саралы, Татарский Кабан, Кирби, Курманаково, Мокрые Курнали, Альвидино, Морты, Большой Шурняк, Татарский Янтык, Большая Елга, Большой Салтан, Большой Ошняк, Бурбаш, Норма, Верхний Субаш, Янгулово, Байлянгар, Сиза, Карадуван, Туембаш, Чутеево, Хозесаново, Старая Задоровка, Энтуганы, Арбор, Нуринер, Старое Тябердино, Кугунур |
| Кластер 5 | Аю, Ст.Мелькен, Семяково, Старый Иркеняш, Ляки, Средний Багряж, Верхние Юшады, Сарсаз Такермень, Старое Байсарово, Старое Сафарово, Бишмунча, Новый Имян, Атясево, Исансупово, Татарские Суксы, Старое Курмашево, Ерыклинский, Нижние Гордалы, Верхнее Абдулово, Янга-Аул, Бизяки, Старое Гришкино, Штырь, Псеево, Средние Пинячи, Мелекес, Кабан-Бастрык, Кадырово, Мунайка, Рысово, Верхний Такермен, Калмия |
| Кластер 6 | Сугушла, Татар. Шуган, Кряш-Шуран, Старое Шугурово, Учалле, Нижний Табын, Старый Каразерик, Салауз-Мухан, Акбаш, Дым-Тамак, Сарабикулово, Федотовка, Янчиково, Старый Юмралы, Нижние Чершилы, Иж-Бобья, Балтачево, Масягутово, Большой Сухояш, Тойгильдино, Иляксаз, Сулеево, Кавзияково, Мальбагуш, Байряки-Тамак, Верхнее Стярле, Баюково, Старые Карамалы, Карамалы, Асеево, Сарлы, Каракашлы |
| Кластер 7 | Маметьево, Савалеево, Чубуклы, Сарапала, Сармашбаш, Абдрахманово, Светлое Озеро, Васильевка, Большие Кушманы, Акзигитово, Кугеево, Мамадыш-Акилово, Тюгеевка, Урмышла, Старые Чечкабы, Старые Бугады, Камаево, Салихово, Табарле, Тойгузино, Новое Алимово, Кзыл-Яр, Новые Чути, Татарская Тумбарла, Девятерня, Исенбаево, Верхние Шипки, Старое Утяганово, Янга-Аул, Биктово, Ямурзино, Крынды |

Number\_of\_clusters = 14, Theta = 0.2

|  |  |
| --- | --- |
| Кластер 1 | Аю, Ст.Мелькен, Семяково, Старый Иркеняш, Ляки, Средний Багряж, Верхние Юшады, Сарсаз Такермень, Старое Байсарово, Старое Сафарово, Бишмунча, Новый Имян, Атясево, Исансупово, Татарские Суксы, Старое Курмашево, Ерыклинский, Нижние Гордалы, Верхнее Абдулово, Янга-Аул, Бизяки, Старое Гришкино, Штырь, Псеево, Средние Пинячи, Мелекес, Кабан-Бастрык, Кадырово, Мунайка, Рысово, Верхний Такермен, Калмия |
| Кластер 2 | Сугушла, Татар. Шуган, Кряш-Шуран, Старое Шугурово, Учалле, Нижний Табын, Старый Каразерик, Салауз-Мухан, Акбаш, Дым-Тамак, Сарабикулово, Федотовка, Янчиково, Старый Юмралы, Нижние Чершилы, Иж-Бобья, Балтачево, Масягутово, Большой Сухояш, Тойгильдино, Иляксаз, Сулеево, Кавзияково, Мальбагуш, Байряки-Тамак, Верхнее Стярле, Баюково, Старые Карамалы, Карамалы, Асеево, Сарлы, Каракашлы |
| Кластер 3 | Маметьево, Савалеево, Чубуклы, Сарапала, Сармашбаш, Абдрахманово, Светлое Озеро, Васильевка, Большие Кушманы, Акзигитово, Кугеево, Мамадыш-Акилово, Тюгеевка, Урмышла, Старые Чечкабы, Старые Бугады, Камаево, Салихово, Табарле, Тойгузино, Новое Алимово, Кзыл-Яр, Новые Чути, Татарская Тумбарла, Девятерня, Исенбаево, Верхние Шипки, Старое Утяганово, Янга-Аул, Биктово, Ямурзино, Крынды |
| Кластер 4 | Большие Кибя-Кози, Татарская Икшурма, Нижний Шандер, Старые Чукалы, Мочалей, Псяк, Куюк Ерыкса, Ильтень-Бута, Нижний Искубаш, Татарские Шатрашаны, Муслюмкино, Новые Ишли, Татарские Тюки, Старое Шаймурзино, Ст.Татар. Адам, Большая Цильна, Сухие Курнали |
| Кластер 5 | Мамашир, Большой Менгер, Бурды, Абалачи, Старый Студенец, Татарский Елтан, Калейкино, Кичкальня, Татарское Ходяшево, Кибячи, Кугарчино, Янга-Сала, Молвино, Сатламышево, Большие Ачасыры, Бурундуки |
| Кластер 6 | Пелево, Именьково, Ямашурма, Шали, Татарские Саралы, Татарский Кабан, Кирби, Курманаково, Мокрые Курнали, Альвидино, Морты, Большой Шурняк, Татарский Янтык, Большая Елга, Большой Салтан, Большой Ошняк |
| Кластер 7 | Бурбаш, Норма, Верхний Субаш, Янгулово, Байлянгар, Сиза, Карадуван, Туембаш, Чутеево, Хозесаново, Старая Задоровка, Энтуганы, Арбор, Нуринер, Старое Тябердино, Кугунур |
| Кластер 8 | Мендюш, Качелино, Апазово, Средний Пшалым, Утар-Аты, Ключи-Сап, Байкал, Ашитбаш, Евлаштау, Урясьбаш, Верхняя Корса, Измя, Большие Верези, Сардебаш, Кильдебяк, Старый Мичан |
| Кластер 9 | Нижний Нурлат, Савиново, Большие Кургузи, Бикулово, Ст. Баран//Иске Рязап, Новый Баран, Кривое Озеро, Курманаево, Ташлы-Ковали, Ислейтар, Кубян, Коморгузя, Большой Битаман, Большие Ковали, Малый Сулабаш, Альдермыш |
| Кластер 10 | Средние Тиганы, Старое Алпарово, Старые Киязлы, Исляйкино, Нижнее Алькеево, Татарское Ахметьево, Старое Ямкино, Новые Челны, Нижняя Каменка, Лашманка, Березняк, Тавель, Шахмайкино, Старое Ибрайкино, Татар. Волчье, Нов. Кадеево |
| Кластер 11 | Мордовский Каратай, Большие Кармалы, Ташкирмень, Средний Юрткуль, Балтачево, Большие Болгояры, Азимово-Курлебаш, Большие Янгасала, Татарское Бурнаево, Ямбухтино, Каргополь, Тяжбердино, Татарская Тахтала, Измери, Старые Салманы, Татарское Тюгульбаево |
| Кластер 12 | Черки-Кильдуразы, Сюндюково, Чишма, Деушево, Бик-Утеево, Верхние Лащи, Старые Тинчали, Бакрче, Старое Барышево, Чуру-Барышево, Большие Атряси, Большая Турма, Большой Сардек, Шамбулыхчы, Татарская Беденьга, Большая Карланга |
| Кластер 13 | Нижняя Сунь, Ятмас Дусай, Ишкеево, Верхняя Сунь, Шеморбаш, Козяково-Челны, Албай, Тюбяк, Нижний Таканыш, Средние Кирмени, Крещеный Пакшин, Верхняя Ошма, Янцевары, Кукча, Уразбахты// Татарский Омар, Тяберлины Челны |
| Кластер 14 | Акбулатово, Верхняя Татарская Майна, Новое Демкино, Сунчелеево, Кутлушкино, Кульбаево-Мараса, Новая Амзя, Степная Шентала, Баландыш, Субаш, Ядыгерь, Сатышево, Тубулга, Старая Киреметь, Новый Арыш, Новое Узеево |