ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ БЮДЖЕТНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«ФИНАНСОВЫЙ УНИВЕРСИТЕТ ПРИ ПРАВИТЕЛЬСТВЕ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ»**

**(ФИНАНСОВЫЙ УНИВЕРСИТЕТ)**

Кафедра анализа данных и машинного обучения

Факультета информационных технологий и анализа больших данных

***Дисциплина: «Машинное обучение в семантическом и сетевом анализе»***

*Направление подготовки: «Прикладная математика и информатика»*

*Профиль: «Анализ данных и принятие решений в экономике и финансах»*

*Факультет информационных технологий и анализа больших данных*

*Форма обучения очная*

*Учебный 2023/2024 год, 6 семестр*

**Курсовая работа на тему:**

«Анализ динамики модулярности в многошаговых алгоритмах поиска сообществ в сетях»

*Выполнил:*

студент группы ПМ21-4

Яковец Андрей Игоревич

*Научный руководитель:*

Старший преподаватель Маковейчук К.А.

**Москва 2024**

**Содержание**

[Введение 4](file:///C:\Users\ASUS\Downloads\Курсовая%20Матусков.docx#_heading=h.gjdgxs)

[Глава 1. Описание методов и метрик 5](file:///C:\Users\ASUS\Downloads\Курсовая%20Матусков.docx#_heading=h.30j0zll)

[1.1 Алгоритмы кластеризации на графовых наборах данных 5](file:///C:\Users\ASUS\Downloads\Курсовая%20Матусков.docx#_heading=h.1fob9te)

[1.1.1 Лувенский алгоритм 5](file:///C:\Users\ASUS\Downloads\Курсовая%20Матусков.docx#_heading=h.3znysh7)

[1.1.2 Алгоритм Гирвана-Ньюмана 7](file:///C:\Users\ASUS\Downloads\Курсовая%20Матусков.docx#_heading=h.2et92p0)

[1.1.3 Жадный алгоритм максимизации модулярности 9](file:///C:\Users\ASUS\Downloads\Курсовая%20Матусков.docx#_heading=h.tyjcwt)

[1.1.4 Синхронный метод распространения меток 10](file:///C:\Users\ASUS\Downloads\Курсовая%20Матусков.docx#_heading=h.3dy6vkm)

[1.1.5 Асинхронный метод распространения меток 11](file:///C:\Users\ASUS\Downloads\Курсовая%20Матусков.docx#_heading=h.1t3h5sf)

[1.1.6 Алгоритм асинхронного распространения жидкости 13](file:///C:\Users\ASUS\Downloads\Курсовая%20Матусков.docx#_heading=h.4d34og8)

[1.1.7 Метод k средних 14](file:///C:\Users\ASUS\Downloads\Курсовая%20Матусков.docx#_heading=h.2s8eyo1)

[1.1.8 Пространственная кластеризация основанная на плотности (DBSKAN) 15](file:///C:\Users\ASUS\Downloads\Курсовая%20Матусков.docx#_heading=h.17dp8vu)

[1.1.9 Агломеративная кластеризация 16](file:///C:\Users\ASUS\Downloads\Курсовая%20Матусков.docx#_heading=h.3rdcrjn)

[1.2 Метрики 17](file:///C:\Users\ASUS\Downloads\Курсовая%20Матусков.docx#_heading=h.26in1rg)

[1.2.1 Модулярность 17](file:///C:\Users\ASUS\Downloads\Курсовая%20Матусков.docx#_heading=h.lnxbz9)

[1.2.2 Коэффициент силуэта 19](file:///C:\Users\ASUS\Downloads\Курсовая%20Матусков.docx#_heading=h.35nkun2)

[1.2.3 Индекс Калински-Харабаша 20](file:///C:\Users\ASUS\Downloads\Курсовая%20Матусков.docx#_heading=h.1ksv4uv)

[Глава 2. Применение алгоритмов кластеризации 23](file:///C:\Users\ASUS\Downloads\Курсовая%20Матусков.docx#_heading=h.2jxsxqh)

[2.1 Бизнес задача 23](file:///C:\Users\ASUS\Downloads\Курсовая%20Матусков.docx#_heading=h.z337ya)

[2.2 Сбор данных 24](file:///C:\Users\ASUS\Downloads\Курсовая%20Матусков.docx#_heading=h.3j2qqm3)

[2.3 Характеристика и результаты работы алгоритмов 27](file:///C:\Users\ASUS\Downloads\Курсовая%20Матусков.docx#_heading=h.1y810tw)

[2.4 Анализ динамики модулярности 28](file:///C:\Users\ASUS\Downloads\Курсовая%20Матусков.docx#_heading=h.4i7ojhp)

[2.5 Результаты 40](file:///C:\Users\ASUS\Downloads\Курсовая%20Матусков.docx#_heading=h.32hioqz)

[Заключение 43](file:///C:\Users\ASUS\Downloads\Курсовая%20Матусков.docx#_heading=h.1hmsyys)

[Список использованных источников 45](file:///C:\Users\ASUS\Downloads\Курсовая%20Матусков.docx#_heading=h.41mghml)

[Приложение А. Программный код 47](file:///C:\Users\ASUS\Downloads\Курсовая%20Матусков.docx#_heading=h.2grqrue)

**Введение**

В быстро развивающейся сфере анализа данных изучение графовых наборов данных стало одной из важнейших областей исследований. Графы, характеризующиеся узлами и связями между ними, служат мощным инструментом для моделирования сложных систем в различных областях, включая социальные сети, транспортные сети, биологические сети и многое другое.

Задача кластеризации в этих графах - группировки узлов таким образом, чтобы узлы внутри группы были более связаны друг с другом, чем с узлами в других группах – является достаточно сложной.

Цель данной курсовой работы - получить полное представление о различных методах кластеризации графов, оценить их эффективность и применимость в реальных наборах данных графов, а также изучить основную метрику модулярности. Также будет представлена практическая реализация методов кластеризации на наборах данных графов с помощью современных инструментов и библиотек, таких как NetworkX. Такой практический подход позволит не только теоретически изучить методы кластеризации, но и получить практический опыт применения этих методов к реальным графовым наборам данных.

# Описание методов и метрик

## **Алгоритмы кластеризации на графовых наборах данных**

### **Лувенский алгоритм**

Лувенский алгоритм - это иерархический алгоритм кластеризации, разработанный в Университете Лёвена Винсеном Блонделем в 2008 году и основан на идее максимизации модулярности разбиения, который работает путем итеративного объединения или разделения общин до достижения максимальной модулярности.

Алгоритм состоит из двух фаз:

1. Локальное перемещение

В этой фазе каждый узел рассматривается отдельно, и его кластер обновляется на основе приращения модулярности, которое получается в результате перемещения его в соседний кластер.

Процесс локального перемещения может быть описан следующим образом:

* 1. Для каждого узла i в графе, рассчитывается приращение модулярности ΔQ, которое получается в результате перемещения его в каждый из соседних кластеров.
  2. Для каждого соседнего кластера C узла i, рассчитывается общий вес ребер между i и узлами в C.
  3. Рассчитывается приращение модулярности ΔQ для каждого соседнего кластера C как разница между общим весом ребер между i и узлами в C и общим весом ребер между i и узлами в его текущем кластере.
  4. Выбирается соседний кластер C, который приводит к максимальному приращению модулярности ΔQ.
  5. Узел i перемещается в кластер C, если приращение модулярности ΔQ положительно.

Фаза локального перемещения повторяется для каждого узла в графе до тех пор, пока не будет достигнуто максимальное приращение модулярности.

1. Глобальная агрегация

В этой фазе кластеры агрегируются в новый граф, где каждый узел представляет кластер, а ребра представляют связи между кластерами.

Процесс глобальной агрегации может быть описан следующим образом:

* 1. Создается новый граф G', где каждый узел представляет кластер в исходном графе.
  2. Для каждой пары кластеров C1 и C2 в исходном графе, рассчитывается общий вес ребер между узлами в C1 и узлами в C2.
  3. Добавляется ребро между узлами, представляющими C1 и C2 в новом графе G', с весом, равным общему весу ребер между узлами в C1 и узлами в C2.
  4. Фаза локального перемещения повторяется на новом графе G' до тех пор, пока не будет достигнуто максимальное приращение модулярности.

Фаза глобальной агрегации повторяется до тех пор, пока кластеры не сойдутся, то есть до тех пор, пока не будет достигнуто максимальное приращение модулярности.

Итеративно повторяя фазы локального перемещения и глобальной агрегации, Лувенский алгоритм может обнаружить высококачественные кластеры в графе.

Алгоритм итеративно проходит между этими двумя фазами до достижения максимальной модулярности.

Лувенский алгоритм имеет несколько преимуществ, включая:

* Способность быстро обрабатывать графовые наборы данных больших размеров
* Возможность параллелизации, что делает алгоритм подходящим для графовых наборов данных больших размеров
* Простота реализации, так как существует множество библиотек и инструментов с открытым исходным кодом (NetworkX и др.)                                                                                                                     Однако, Лувенский алгоритм также имеет некоторые ограничения, включая:
* Слабая способность обнаружения небольших сообществ в силу предела разрешения
* Чувствительность к начальному присвоению кластеров, который может повлиять на конечный результат
* Слабая способность работы с графами с высокой плотностью
* Чувствительность к отсутствующим данным

В целом, Лувенский алгоритм является мощным инструментом для обнаружения кластеров в графовых наборах данных и широко используется в многих областях.

### **Алгоритм Гирвана-Ньюмана**

Алгоритм Гирвана-Ньюмана - это алгоритм обнаружения сообществ, разработанный Мишель Гирван и Марком Ньюманом в 2002 году. Это иерархический алгоритм, который использует подход разделения для выявления сообществ в сети.

Алгоритм имеет следующие шаги:

1. Вычисление степеней посредничества всех существующих ребер. Степень посредничества - это мера того, в какой степени ребро лежит на кратчайших путях между другими узлами.
2. Удаление из сети ребра с наивысшей степенью посредничества.
3. Проверка, не разделился ли граф на два или более связанных компонента. Если да, то он идентифицирует эти компоненты как отдельные сообщества.
4. Алгоритм продолжает удалять ребра с наивысшими степенями посредничества и выявлять сообщества до тех пор, пока больше не будет удалено ни одного ребра, или не будет достигнут критерий остановки.

Отсюда можно сделать выводы о том, что:

* Алгоритм Гирвана-Ньюмана не требует предварительных знаний о сообществе
* Использует разделительный подход, начиная работу со всей сетью и итеративно деля ее на более мелкие сообщества

Алгоритм Гирвана-Ньюмана имеет несколько преимуществ, включая:

* Создание высококачественных сообществ, плотно связанных и хорошо разделенных
* Способность работать с взвешенными и невзвешенными графами
* Устойчивость к шуму в данных в силу фокусирования на общей структуре сети, а не на отдельных ребрах

Однако, алгоритм Гирвана-Ньюмана также имеет некоторые ограничения, включая:

* Медленная скорость для больших сетей в силу необходимости итеративных удаления ребер и проверки структуры сообщества
* Чувствительность к начальному присвоению общин, который может повлиять на конечный результат
* Слабая способность работы с графами с очень высокой или очень низкой плотностью
* Неспособность работать с графами, имеющими несколько типов ребер

В целом, алгоритм Гирвана-Ньюмана является мощным инструментом для обнаружения сообществ в сетях, а его иерархический подход и ориентация на степени посредничества между гранями делают его популярным среди исследователей и практиков.

### **Жадный алгоритм максимизации модулярности**

Жадный алгоритм максимизации модулярности [3] - это алгоритм обнаружения сообществ, который использует жадный подход для оптимизации метрики модулярности. Модулярность - это мера силы структуры сообществ в сети, и она определяется следующим образом:

где:

* A – это матрица смежности
* – это суммы весов ребер, присоединенных к узлам i и j соответственно
* – это сумма весов всех ребер в графе
* – это общее количество узлов в графе
* – это сообщества, к которым принадлежат узлы i и j
* – это символ Кронекера

Алгоритм имеет следующие шаги:

1. Каждый узел приписывается к своему сообществу.
2. Алгоритм итеративно объединяет сообщества таким образом, чтобы максимизировать показатель модулярности.
3. Выбор пары узлов, объединение которых приведет к наибольшему увеличению модулярности.
4. Обновление назначений сообществ для объединенных узлов.
5. Продолжение объединения узлов и обновления назначения сообществ до тех пор, пока не будет достигнуто слияние, которое увеличит оценку модулярности.

Жадный алгоритм максимизации модулярности имеет несколько преимуществ, включая:

* Способность быстро обрабатывать графовые наборы данных больших размеров
* Возможность параллелизации, что делает алгоритм подходящим для графовых наборов данных больших размеров
* Простота реализации, так как существует множество библиотек и инструментов с открытым исходным кодом (NetworkX и др.)

Однако Жадный алгоритм максимизации модулярности также имеет некоторые ограничения, включая:

* Чувствительность к начальному присвоению общин, который может повлиять на конечный результат
* Слабая способность работы с графами с очень высокой или очень низкой плотностью
* Неспособность работать с графами, имеющими несколько типов ребер

### **Синхронный метод распространения меток**

Метод синхронного распространения меток (SLP) [4], также известный как алгоритм распространения меток (LPA), - это простой и широко используемый алгоритм для обнаружения сообществ в графах. В отличие от своего асинхронного аналога, SLP обновляет метки всех узлов одновременно на каждой итерации.

Алгоритм имеет следующие шаги:

1. Каждый узел приписывается к своему сообществу
2. На каждой итерации все узлы одновременно обновляют свои метки, основываясь на метках своих соседей. В частности, каждый узел принимает метку, которая наиболее распространена среди его соседей.
3. В случаях, когда несколько меток одинаково часто встречаются среди соседей узла, для выбора новой метки применяются правила устранения ничьей.
4. Продолжение итерационного процесса (2-3 шаги) до тех пор, пока не будет достигнут критерий остановки. Обычно сходимость достигается, когда метки узлов больше не меняются между итерациями или когда достигнуто максимальное количество итераций.

SLP имеет следующие преимущества:

* Эффективен при работе с графами с малых и средних размеров
* Может быть в некоторой степени распараллелен, несмотря на одновременное обновление меток узлов.

Недостатки SLP:

* Слабая способность обнаружения небольших сообществ в силу предела разрешения
* Чувствительность к начальной маркировке узлов. Различные инициализации могут привести к различным структурам сообществ, и нет гарантированного метода выбора наилучших начальных меток.
* Неэффективен при работе с пересекающимися сообществами

### **Асинхронный метод распространения меток**

Метод Asynchronous Label Propagation (ALP) [5] - алгоритм, используемый для обнаружения сообществ в графах, являющийся расширением алгоритма Label Propagation Algorithm (LPA), который известен своей простотой и масштабируемостью.

Алгоритм имеет следующие шаги:

1. Каждый узел приписывается к своему сообществу
2. В отличие от синхронного подхода в LPA, где все узлы обновляют свои метки одновременно, в ALP метки узлов обновляются асинхронно. В каждой итерации узлы обновляют свои метки на основе меток своих соседей, которые могли быть обновлены в предыдущих итерациях. Такой асинхронный процесс обновления обеспечивает параллельное выполнение и более быструю сходимость.
3. Присваивание узлу метки, которая наиболее часто встречается среди его соседей (локальной окрестности).
4. В случаях, когда несколько меток одинаково часто встречаются среди соседей узла, для выбора новой метки применяются правила устранения ничьей.
5. Продолжение итерационного процесса (2-4 шаги) до тех пор, пока не будет достигнут критерий остановки. Обычно сходимость достигается, когда метки узлов больше не меняются между итерациями или когда достигнуто максимальное количество итераций.

Такой алгоритм имеет следующие преимущества:

* В силу асинхронной схемы обновления меток узлов ALP обладает высокой масштабируемостью и эффективностью и способностью работать с крупномасштабными сетями.
* Способность работать с графами с различной плотностью, что делает данный алгоритм адаптивным
* Возможность распараллеливания

Однако ему присущи и некоторые недостатки, такие как:

* Слабая способность обнаружения небольших сообществ в силу предела разрешения

Неспособность выявления иерархических структур сообществ, когда они демонстрирует вложенные или перекрывающиеся отношения на разных уровнях

### **Алгоритм асинхронного распространения жидкости**

Алгоритм асинхронного распределения жидкости [2] – это алгоритм, используемый для обнаружения сообществ в графа, основанный на идее введения ряда жидкостей (т.е. сообществ) в неоднородную среду (т.е. неполный граф), где жидкости будут расширяться и выталкивать друг друга под влиянием топологии среды, пока не будет достигнуто стабильное состояние.

Данный алгоритм выполняется по шагам. На каждом из них он итерируется по всем узлам графа в случайном порядке, обновляя их метки в соответствии с правилом. Когда метки графа перестают обновляться 2 шага подряд, итерационный процесс останавливается. Правило обновления, упомянутое ранее, может быть формально сформулировано следующим образом:

где

* v – обновляемая вершина,
* - набор кандидатов, из которых будет состоять новое сообщество,
* *–* соседи v,
* *–* плотность сообщества c,
* *–* вершина сообщества, к которому принадлежит w
* – это символ Кронекера

Так, если содержит текущую метку узла *v*, то *v* не меняет свою метку. Иначе правило обновления выбирает случайную метку в пределах в качестве нового сообщества для *v*. Данное утверждения может быть формализировано так:

где

* – метка узла v на следующем шагу
* – случайная выборка из дискретного равномерного распределения множества

Данное уравнение гарантирует, что ни одно сообщество не будет удалено из графа, так как когда сообщество *c* сжимается в одну вершину *v*, *c* имеет максимально возможную плотность на правиле обновления *v*, что гарантирует c ∈ , и, следовательно, .

В целом, можно сказать, что алгоритм асинхронного распределения жидкости отлично себя показывает при работе с большими графами и графами с большими параметрами смешивания. Он является самый быстрым алгоритмом с линейной вычислительной сложностью среди своих аналогов, наряду с вышеупомянутым методом синхронного распространения меток (SLP). К тому же, исследования показывают, что производительность SPA быстро снижается для больших параметров смешивания, в то время как данный алгоритм способен обнаруживать значимые сообщества при всех значениях параметров смешивания.

### **Метод k средних**

K-means - алгоритм кластеризации, используемый в машинном обучении, работа которого заключается в разбиении набора данных на множество кластеров, где каждая точка данных принадлежит к кластеру с ближайшим средним (центром), служащим прототипом кластера.

Однако применение k-means непосредственно к кластеризации графов требует некоторой адаптации из-за неотъемлемых различий между данными графов и традиционными векторными данными.

Она включает следующие шаги:

1. Представление узлов графа в виде вектора признаков
2. Преобразование графа в высокоразмерное векторное пространство
3. Подбор параметра k
4. Назначение каждому узлу ближайшего центроида на основе выбранной метрики расстояния (евклидово и др.). Узлы с одинаковыми центроидами образуют сообщества
5. Обновление меток узлов на основе среднего значения векторов признаков узлов, отнесенных к каждому классу (минимизация внутриклассовой дисперсии)
6. Повторение 4 и 5 шагов итеративно до тех пор, пока не будут достигнуты критерии сходимости

Результаты K-means могут меняться в зависимости от начальных центроидов (параметра k), и найти хорошую стратегию инициализации для данных графов довольно сложно. Для подбора k испольуются метод локтя и метод силуэта.

В целом, k-means cпособен обрабатывать большие массивы данных с относительно низкими вычислительными затратами, а также масштабироваться на больших наборах данных.

К его ограничениям относятся:

* Чувствительность к выбросам
* Предположение о выпуклости и изотропности кластеров, что не всегда является правдивым

### **Пространственная кластеризация основанная на плотности (DBSKAN)**

Основанная на плотности пространственная кластеризация для приложений с шумами группирует точки, тесно прилегающие друг к другу (точки с большим количеством ближайших соседей) на основе их плотности в многомерном пространстве.

Имеет 2 ключевых понятия:

* Epsilon - параметр расстояния, определяющий максимальное расстояние между двумя точками, чтобы они считались соседями
* MinPts – минимальное количество точек, необходимое для формирования кластера

В качестве входных данных алгоритм принимает векторное представление графа.

Ключевые моменты в DBSKAN:

* Каждый узел графа считается точкой. Расстояние между узлами определяется на основе структуры графа, часто с помощью расстояния кратчайшего пути или аналогичной меры.
* Плотность узла определяется количеством его соседей в пределах определенного расстояния (eps).
* Узлы, имеющие достаточное количество соседей (minPts), считаются частью одного кластера. Этот процесс является итерационным, когда каждый вновь выявленный кластер расширяется за счет включения в него своих соседей.
* Узлы, не удовлетворяющие критериям плотности, считаются шумом и не включаются ни в один кластер.

Алгоритм имеет такие преимущества, как:

* Автоматическое определения количества кластеров
* Эффективная обработка шумов

Важно уточнить, что подбор параметров сильно влияет на производительность данного алгоритма. Их поиск, как правило, осуществляется с помощью Grid-search и Random-search.

### **Агломеративная кластеризация**

Агломеративная кластеризация строит кластеры пошагово, где на каждом этапе происходит объединение ближайшей пары кластеров. Этот процесс можно представить в виде древовидной структуры, где каждый узел представляет собой кластер, а листья дерева - отдельные объекты.

Заранее граф должен быть преобразован, например, в матрицу расстояний, сходства или же в его векторное представление.

Агломеративная кластеризация имеет следующие шаги:

1. Представление каждого узла графа в виде отдельного кластера

2. Итеративное объединение пары наиболее похожих кластера (например, по количеству общих узлов или расстояния кратчайшего пути между узлами кластеров)

3. Пересчитывание сходства между новым кластером и старыми.

4. Повтор 2 и 3 шагов до тех пор, пока не будет достигнуто заданное количество кластеров

Алгоритм не требует хранения в памяти графа целиком, однако он предполагает вычисления сходства между новым кластером и всеми остальными на каждом итерации, что может приводить к вычислительным сложностям.

В целом агломеративная кластеризация, примененная к кластеризации графов, предлагает иерархический подход к группировке узлов на основе их связности, обеспечивая четкое визуальное представление процесса кластеризации и связей между кластерами. Тем не менее, важными соображениями являются его вычислительная сложность и необходимость тщательной настройки параметров.

## **Метрики**

Оценка эффективности алгоритмов кластеризации предполагает использования метрик, которые количественно определяют качество результатов кластеризации.

### **Модулярность**

Модулярность – это метрика, используемая в сетевом анализе для количественной оценки силы разделения сети на модули или сообщества. Она была введена М. Э. Дж. Ньюманом в 2004 году и с тех пор стала фундаментальным понятием в анализе сложных сетей. Основная цель модулярности - оценить, насколько хорошо сеть разделена на модули, где узлы внутри одного модуля более плотно связаны друг с другом, чем с узлами в других модулях.

Модулярность определяется как разница между количеством ребер, входящих в сообщества, и ожидаемым количеством ребер в случайной сети с тем же распределением степеней. Ее формула имеет вид:

где:

* A – это матрица смежности
* – это суммы весов ребер, присоединенных к узлам i и j соответственно
* – это сумма весов всех ребер в графе
* – это общее количество узлов в графе
* – это сообщества, к которым принадлежат узлы i и j
* – это символ Кронекера

Значения модулярности интерпретируются следующим образом:

* Положительная модулярность указывает на то, что сеть является модулярной, т.е. узлы в одном модуле с большей вероятностью будут связаны друг с другом, чем с узлами в других модулях. Чем выше модулярность, тем сильнее разделение сети на модули.
* Нулевая модулярность указывает на то, что сеть случайным образом разбита на модули, и внутри модулей не больше ребер, чем можно было бы ожидать по случайности.
* Отрицательная модулярность указывает на то, что сеть не является модулярной, при этом узлы с большей вероятностью будут связаны с узлами в других модулях, чем с узлами в своем собственном модуле.

К ограничениям такой метрики относят:

* Предел разрешения: модулярность может быть не в состоянии обнаружить сообщества небольшого размера или сообщества, которые не имеют плотной связи.
* Проблема оптимизации: поиск оптимального разбиения сети, максимизирующего модульность, является трудной задачей, и для приближенного решения часто используются эвристические алгоритмы.
* Субъективность: Выбор количества модулей или сообществ может быть субъективным, и разные алгоритмы могут давать разные результаты

### **Коэффициент силуэта**

Коэффициент силуэта [10] - это метрика, используемая для оценки качества кластеризации при анализе данных. Она измеряет, насколько объект похож на свой кластер по сравнению с другими кластерами. Для того чтобы его вычислить, выполняются следующие шаги:

1. Вычисляется среднее расстояние между i-ым и остальными объектами кластера:

где

* - мощность I-го кластера (т.е. число попавших в него объектов),
* - расстояние между объектами i и j кластера I

2. Вычисляется наименьшее среднее расстояние между i-ым кластера и всеми точками в ближайшем кластере (:

где – объект, содержащийся в любом кластере , отличном от

3. Вычисляется коэффициент силуэта:

Значения коэффициента силуэта интерпретируются следующим образом:

* Высокий балл силуэта указывает на то, что конфигурация кластеризации является подходящей и что объекты хорошо сочетаются с собственным кластером и плохо сочетаются с соседними кластерами.
* Низкий балл силуэта указывает на то, что в конфигурации кластеризации может быть слишком много или слишком мало кластеров и что объекты плохо сочетаются с собственным кластером и/или хорошо сочетаются с соседними кластерами.
* Отрицательная оценка силуэта указывает на то, что конфигурация кластеризации плохая и что объекты больше похожи на объекты в других кластерах, чем на объекты в своем кластере.

К ограничениям такой метрики относят:

* Предположение о сферических кластерах: Оценка силуэта предполагает, что кластеры являются выпуклыми и изотропными, что может быть не так в реальных данных. Это предположение может привести к неоптимальным результатам для несферических кластеров.
* Метрика расстояния: выбор метрики расстояния может существенно повлиять на оценку силуэта. Различные метрики расстояния могут привести к разным оценкам силуэта для одного и того же набора данных.
* Вычислительная сложность: Вычисление оценки силуэта для больших наборов данных может быть вычислительно дорогостоящим, особенно для высокоразмерных данных.

### **Индекс Калински-Харабаша**

Индекс Калински-Харабаша [9], также известный как критерий отношения дисперсий, - это метрика, используемая для оценки качества кластеризации при анализе данных. Он измеряет отношение межкластерной дисперсии к внутрикластерной дисперсии. Индекс назван в честь его создателей, Ежи Калинского и Яцека Харабаша, и был представлен в 1974 году.

Для того чтобы его вычислить, выполняются следующие шаги:

1. Вычисляется межкластерная дисперсия (BCSS – Between-Cluster Sum of Squares):

где

* – это количество точек в кластере
* – это центроид кластера
* - общий центроид данных

2. Вычисляется внтуриклассовая дисперсия:

3. Вычисляется индекс:

Значения индекса Калинского-Харабаша интерпретируются следующим образом:

* Высокий индекс указывает на то, что конфигурация кластеризации является подходящей и что кластеры плотные и хорошо отделены друг от друга.
* Низкий индекс указывает на то, что конфигурация кластеризации может быть неоптимальной и что кластеры плохо разделены или слишком малы.
* Нулевой индекс указывает на то, что конфигурация кластеризации является случайной, и разброс между кластерами не больше, чем можно было бы ожидать по случайности.

К ограничениям такой метрики относят:

* Предположение о сферических кластерах: Индекс Калинского-Харабаша предполагает, что кластеры являются выпуклыми и изотропными, что может быть не так в реальных данных. Это предположение может привести к неоптимальным результатам для несферических кластеров.
* Метрика расстояния: выбор метрики расстояния может существенно повлиять на индекс Калинского-Харабаша. Различные метрики расстояния могут привести к разным значениям индекса для одного и того же набора данных.
* Вычислительная сложность: Вычисление индекса Калинского-Харабаша для больших наборов данных может потребовать больших вычислительных затрат, особенно для высокоразмерных данных.

# Применение алгоритмов кластеризации

## **Бизнес задача**

В качестве задачи собраны данные о страницах Facebook (ноябрь 2017 г.). Эти наборы данных представляют собой подтвержденные синим цветом сети страниц Facebook различных категорий. Узлы представляют страницы, а края являются взаимными сходствами между ними. Выявление таких групп помогает лучше понять структуру рынка, выявить потенциальные риски и возможности для инвестиций, а также улучшить управление корпоративными связями и стратегическими партнёрствами. Связанные компании могут иметь значительное влияние друг на друга в плане финансовой устойчивости и ликвидности. Анализ этих связей помогает выявить потенциальные риски для инвесторов и регуляторов, что особенно важно в условиях глобальных финансовых кризисов.

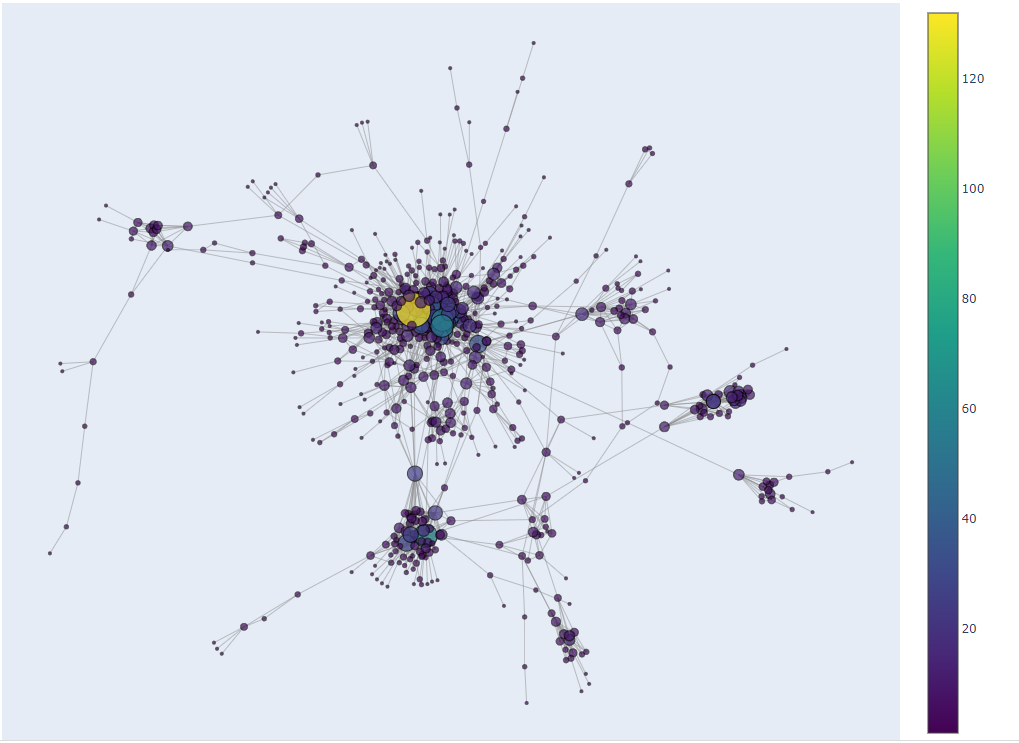
Понимание групп связанных компаний позволяет более эффективно разрабатывать маркетинговые стратегии и планировать рекламные кампании. Это также помогает определить ключевых игроков на рынке и потенциальные возможности для стратегических альянсов.

## **Сбор данных**

Для сбора данных был выбран ресурс https://networkrepository.com/. Первое интерактивное хранилище данных и сетевых данных с визуальной аналитикой в режиме реального времени. Сетевой репозиторий — это не только первый интерактивный репозиторий, но и *крупнейший сетевой репозиторий* с тысячами пожертвований в 30+ доменах (от биологических до данных социальных сетей). Эта большая комплексная коллекция данных сетевого графа полезна для получения важных результатов исследований, а также эталонных наборов сетевых данных для широкого спектра приложений и областей (например, науки о сетях, биоинформатики, машинного обучения, интеллектуального анализа данных, физики и социальных наук) и включает в себя реляционные, атрибутивные, гетерогенные потоковые, пространственные и временные сетевые данные, а также нереляционные данные машинного обучения. Все наборы графовых данных легко загружаются в стандартный согласованный формат.

Собраны данные о страницах Facebook (ноябрь 2017 г.). Эти наборы данных представляют собой подтвержденные синим цветом сети страниц Facebook различных категорий. Узлы представляют страницы, а края являются взаимными сходствами между ними.

Для кластеризации лучше использовать связный граф, поэтому в качестве графа будем использовать крупнейшую связную компоненту. Она состоит из 620 узлов и 2091 ребер (связей) и выглядит следующим образом (размеры узлов пропорциональны степени): (рисунок 3)



*Рисунок SEQ Рисунок \\* ARABIC 3*

## **Характеристика и результаты работы алгоритмов**

## **Лувенский алгоритм**

*Рисунок SEQ Рисунок \\* ARABIC 4*

* Здесь и далее визуализация будет обладать следующими особенностями:
* Каждому кластеру (группе связанных компаний) будет присвоен свой цвет из палитры tab20
* Связи (ребра) внутри кластера будут иметь такой же цвет, как и узлы этого кластера
* Связи (ребра) между разными кластерами имеют черный цвет
* Размер узлов пропорционален степени узла
* Используется пружинная раскладка (spring layout)

Результаты кластеризации:

* Модулярность: 0.653
* Коэффициент силуэта: 0.218
* Индекс Калински-Харабаша: 54.502

## **Алгоритм Гирвана-Ньюмена**

Так как алгоритм является итеративным, необходимо определить оптимальный шаг алгоритма. Для этого будем рассчитывать модулярность на каждом шаге. Получаем, что оптимальным является разбиение после 27 шага

Результаты кластеризации:

* Модулярность: 0.628
* Коэффициент силуэта: 0.279
* Индекс Калински-Харабаша: 41.577

## **Жадный алгоритм максимизации модулярности**

Результаты кластеризации:

* Модулярность: 0.631
* Коэффициент силуэта: 0.24
* Индекс Калински-Харабаша: 46.516

## **Асинхронный метод распространения меток**

Результаты кластеризации:

* Модулярность: 0.566
* Коэффициент силуэта: 0.271
* Индекс Калински-Харабаша: 36.872
* Индекс Дэвиcа-Боулдина: 0.879
* Время выполнения 0.03 сек

## **Синхронный метод распространения меток**

Результаты кластеризации:

* Модулярность: 0.61
* Коэффициент силуэта: 0.29
* Индекс Калински-Харабаша: 41.08

## **Алгоритм асинхронного распространения жидкости**

Результаты кластеризации:

* Модулярность: 0.61
* Коэффициент силуэта: 0.099
* Индекс Калински-Харабаша: 42.729

## **Node2Vec**

Дальнейшие алгоритмы не работают напрямую с графами, поэтому нужно получить матричное представление графа. Будем использовать Node2Vec

Node2vec - это метод, который генерирует векторные представления (эмбеддинги) для узлов графа. Эти эмбеддинги отражают структурные свойства и сходство узлов на основе топологии графа.

Node2vec работает следующим образом:

1. Он выполняет случайные блуждания, чтобы исследовать окрестности каждого узла. При этом происходит компромисс между стратегиями поиска в ширину (BFS) и в глубину (DFS).

* BFS (p → high, q → low): Фокусируется на ближайших соседях, захватывая локальные структуры.
* DFS (p → low, q → high): Исследует дальше от узла, захватывая более глобальные структуры.

2. Рассматривает последовательности, сгенерированные случайными блужданиями, как предложения и применяет модель скип-грамм из обработки естественного языка для изучения эмбеддингов узлов. Модель skip-gram нацелена на предсказание контекста (соседних узлов) данного узла в этих последовательностях.

Node2Vec улавливает разнообразные паттерны и взаимосвязи в графе, в результате чего получаются богатые и информативные вкрапления узлов.

## **K-Means**

Результаты кластеризации:

* Модулярность: 0.625
* Коэффициент силуэта: 0.265
* Индекс Калински-Харабаша: 59.349
* Время выполнения 2.65

## **DBSCAN**

Результаты кластеризации:

* Модулярность: 0.375
* Коэффициент силуэта: 0.257
* Индекс Калински-Харабаша: 35.067
* Время выполнения 0.018

## **Агломеративная кластеризация**

Результаты кластеризации:

* Модулярность: 0.639
* Коэффициент силуэта: 0.314
* Индекс Калински-Харабаша: 57.23
* Время выполнения 0.017

## **Спектральная кластеризация**

Результаты кластеризации:

* Модулярность: 0.62
* Коэффициент силуэта: 0.214
* Индекс Калински-Харабаша: 56.573
* Время выполнения 0.249

## **Анализ динамики модулярности**

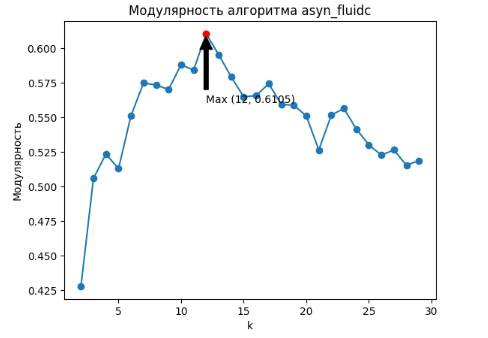
Мы смогли рассмотреть динамику модулярности на алгоритме Гирвана-Ньюмена, Fluid Communities, K-Means, Алгомеративной кластеризации, Спектральной кластеризации. Остальные алгоритмы не имеют этой динамики, так как они сразу выдают итоговое разбиение на сообщества.

## **Алгоритм Гирвана-Ньюмена**

## 

С каждым шагом значение модулярности увеличивается.

## **Fluid Communities**



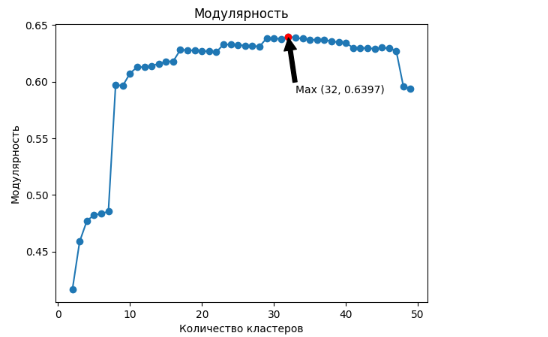
При увеличении числа кластеров до 12 значение модулярности увеличивалось и достигло своего пика, после этого пошло на уменьшение.

## **K-Means**

## 

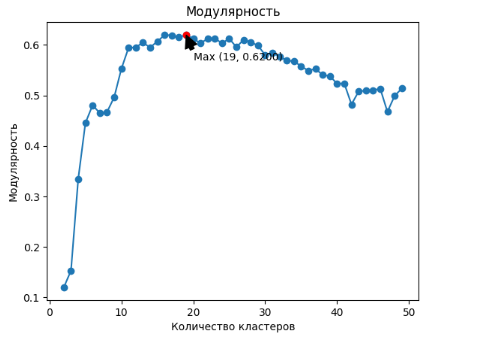
При увеличении числа кластеров значение модулярности увеличивается и достигает своего пика при 24 кластеров.

## **Алгомеративная кластеризация**



При увеличении числа кластеров значение модулярности увеличивается и достигает своего пика при 32 кластеров.

## **Спектральная кластеризация**



При увеличении числа кластеров значение модулярности увеличивается и достигает своего пика при 19 кластеров.

## **Результаты**

Соберем всё в одну таблицу и отсортируем по убыванию модулярности



Можно сделать следующие выводы:

1. Модульность (Modularity):

   - Лувенский алгоритм показывают наивысшую модулярность , что указывает на хорошую способность этих методов выявлять четкие кластеры.

   - Алгоритмы LPA (Asynchronous и Synchronous), DBSCAN имеют значительно более низкую модулярность, что указывает на меньшую способность этих методов выявлять четкие сообщества в графе.

2. Время выполнения:

   - DBSCAN, Лувенский и алгомеративная кластеризация работают наиболее быстро.

   - Алгоритм K-Means Clustering и Spectral Clustering требуют значительно больше времени на выполнение.

**Заключение**

В данной курсовой работе были рассмотрены и проанализированы различные методы кластеризации на графовых наборах данных, такие как Лувенский алгоритм, алгоритм Гирвана-Ньюмана, жадный алгоритм максимизации модулярности, синхронный и асинхронный методы распространения меток, алгоритм асинхронного распределения жидкости, метод k-средних и DBSCAN.

Анализ проводился на примере кластеризации связанных компаний, где каждая компания представлялась узлом в графе, а связи между компаниями – ребрами. Для оценки качества кластеризации использовались различные метрики, такие как модулярность, коэффициент силуэта, индекс Калински-Харабаша и индекс Дэвиса-Боулдина.

Результаты показали, что выбор алгоритма зависит от специфики задачи и требований к качеству кластеров. Лувенский алгоритм продемонстрировал высокие результаты по модулярности, что свидетельствует о хорошей способности этого метода выявлять четко связанные группы компаний.

Время выполнения алгоритмов также было важным фактором. Наиболее быстро работали алгоритмы DBSCAN и Лувенский, тогда как алгоритмы Гирван-Ньюмана, K-Means Clustering и спектральной кластеризации требовали значительно больше времени на выполнение. Это особенно важно при работе с крупными графами, где время выполнения может существенно влиять на практическую применимость алгоритмов.

Таким образом, при выборе метода кластеризации для крупных графовых наборов данных, представляющих связанные компании, следует учитывать баланс между качеством кластеров и временем выполнения алгоритма. Лувенский алгоритм и можно рекомендовать для задач, требующих высокой модулярности и быстрого выполнения.

**Список использованных источников**

1. Cai Z. M. Network community partition based on intelligent clustering algorithm //Компьютерная оптика. – 2020. – Т. 44. – №. 6. – С. 985-989.
2. Parés F. et al. Fluid communities: A competitive, scalable and diverse community detection algorithm //Complex Networks & Their Applications VI: Proceedings of Complex Networks 2017 (The Sixth International Conference on Complex Networks and Their Applications). – Springer International Publishing, 2018. – С. 229-240.
3. Clauset A., Newman M. E. J., Moore C. Finding community structure in very large networks //Physical review E. – 2004. – Т. 70. – №. 6. – С. 066111.
4. Raghavan U. N., Albert R., Kumara S. Near linear time algorithm to detect community structures in large-scale networks //Physical review E. – 2007. – Т. 76. – №. 3. – С. 036106.
5. Cordasco G., Gargano L. Community detection via semi-synchronous label propagation algorithms //2010 IEEE international workshop on: business applications of social network analysis (BASNA). – IEEE, 2010. – С. 1-8.
6. Reichardt J., Bornholdt S. Statistical mechanics of community detection //Physical review E. – 2006. – Т. 74. – №. 1. – С. 016110.
7. Newman M. E. J. Equivalence between modularity optimization and maximum likelihood methods for community detection //Physical Review E. – 2016. – Т. 94. – №. 5. – С. 052315.
8. Davies D. L., Bouldin D. W. A cluster separation measure //IEEE transactions on pattern analysis and machine intelligence. – 1979. – №. 2. – С. 224-227.
9. Caliński T., Harabasz J. A dendrite method for cluster analysis //Communications in Statistics-theory and Methods. – 1974. – Т. 3. – №. 1. – С. 1-27.
10. Rousseeuw P. J. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis //Journal of computational and applied mathematics. – 1987. – Т. 20. – С. 53-65.

**Приложение А. Программный код**

https://github.com/andyak03/NetworkX/blob/efcfd6b69ea8ce16174c5f79eecfc9afa5c11c46/%D0%9A%D1%83%D1%80%D1%81%D0%BE%D0%B2%D0%B0%D1%8F%20%D0%AF%D0%BA%D0%BE%D0%B2%D0%B5%D1%86.ipynb