ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ

ФГАОУ ВО НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет компьютерных наук
Образовательная программа «Прикладная математика и информатика»

Отчет о программном проекте

на тему Веб-сервис кластеризации цифрового ассистента студента
(промежуточный, этап 1)

Выполнил студент группы БПМИ1910  А.А.Ибаева
Подпись И.О. Фамилия

8 февраля 2020 года
Дата

Принял: руководитель проекта
Андрей Андреевич Паринов
Имя, Отчество, Фамилия

младший научный сотрудник
Должность, ученое звание

МЛ ИССА
Место работы (Компания или подразделение НИУ ВШЭ)

Дата проверки_____2021 _____ _____
Оценка (по 10-тибалльной шкале) Подпись

Москва 2021

Содержание

1. Основные термины и определения

Веб-сервис — идентифицируемая уникальным веб-адресом (URL-адресом) программная система со стандартизированными интерфейсами, а также HTML-документ сайта, отображаемый браузером пользователя.

Граф есть совокупность двух множеств — множества самих объектов, называемого множеством вершин и множеством их парных связей, называемой множеством рёбер.

Матрица смежности графа G с конечным числом вершин n — это квадратная матрица A размера $n*n$, в которой значение элемента $a_{i,j}$ равно числу рёбер из i-й вершины графа в j-ю вершину.

Матрица степеней вершин — это диагональная матрица $D = diag(d_1,...,d_k)$, где $d_i=\sum_{j=1}^{k} w_{i,j}$.

Пусть дана матрица A, ненулевой вектор x, скаляр l. Если $Ax=lx$, то x - **собственный вектор**, а l - **собственное значение** графа.

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

Алгоритм кластеризации — функция $a:X\to Y$, которая любому объекту $x\in X$ ставит в соответствие идентификатор кластера $y\in Y$.

Ассоциативные правила представляют собой механизм нахождения логических закономерностей между связанными элементами (событиями или объектами).

2. Введение

В начале второго курса каждому студенту ФКН предстоит сложная задача: выбрать тему проекта, который он будет выполнять на протяжении всего учебного года. Но как определиться, какой проект стоит выбрать? Ведь за первый курс обучения далеко не все поняли, что конкретно им интересно, а что нет. А если ошибиться в этой сложной задаче, вероятно, работать над проектом будет крайне тяжело. И ведь всем было бы гораздо легче и удобнее, если бы существовал сервис, позволяющий справиться с этой трудной задачей!

К тому же студенты нашего факультета сталкиваются с тем, что нет единой площадки, которая позволяет отслеживать релевантную информацию о подходящих им проектах.

Смысл цифрового ассистента состоит в том, чтобы решить данные задачи. Одна из главных функций ассистента - отображать рекомендации проектов на основании интересов студента и различных исторических данных. Как раз-таки она и поможет студентам определиться с наиболее подходящим проектом, анализируя множество факторов.

Поэтому в рамках данного проекта мы с членами команды разрабатываем веб-сервис, который и будет выполнять все эти важные задачи. Он будет представлять рекомендации, используя реализацию нескольких алгоритмов кластеризации, ассоциативных правил и нейросетей.

Цель проекта — это разработка сайта и мобильного приложения рекомендательной системы проектной деятельности.

Задачи проекта:

1) Реализация клиента Android, iOS

2) Разработка сайта

3) Веб-сервис кластеризации (задача, решаемая мной)

4) Веб-сервис рекомендательной системы

5) Разработка архитектуры

3. Обзор и сравнительный анализ алгоритмов кластеризации

Название алгоритма	Входные данные	Масштабируемость	Вариант использования	Геометрия
K-Means	Число кластеров	Очень большое количество образцов, среднее количество кластеров	Плоская геометрия, небольшое количество кластеров	Расстояния между точками
Spectral clustering	Число кластеров	Среднее количество образцов, маленькое количество кластеров	Небольшое количество кластеров, неплоская геометрия	Граф расстояний
Birch	Фактор ветвления, порог расстояния, необязательный глобальный кластеризатор	Большое количество образцов и кластеров	Большой набор данных, удаление выбросов, сокращение данных	Евклидово расстояние между точками

3.1. K-Means

Алгоритм K-Means — это итеративный алгоритм, который пытается разделить набор данных на *K* отдельных непересекающихся кластеров, где каждая точка принадлежит только одной группе. Он присваивает точки кластеру таким образом, чтобы сумма квадратов расстояния между точками и центроидом кластера была минимальной.

Описание алгоритма:

1) Надо указать число *K* - количество кластеров
2) Случайно выбрать *K* центроидов
3) *repeat*
   - присвоить точку к ближайшему центроиду
   - вычислить среднее значение всех точек для каждого кластера и выбрать новый центроид

   until
   позиция центроидов не меняется

Методы оценки числа кластеров:

1) Elbow Method

Несколько раз запускаем K-Means, каждый раз увеличивая число кластеров. Строим график, зависящий от суммы квадратов расстояний и числа кластеров. Найдется точка, в которой прямая начнет изгибаться — это и будет наиболее подходящее число кластеров.

2) Silhouette Analysis

Он используется для определения степени разделения кластеров. Для каждого образца:
- вычисляем среднее расстояние от всех точек в одном кластере (a^i)
- вычисляем среднее расстояние от всех точек в ближайшем кластере (b^i)
- вычисляем следующий коэффициент: $\frac{b^i - a^i}{\max(a^i, b^i)}$

Этот коэффициент принимает значения в интервале [-1, 1]. Если 0, то образец очень близок к соседним кластерам, если 1 - то образец далеко от соседних кластеров, если -1 - образец отправлен к неправильному кластеру.

3.2. Spectral Clustering

Спектральная кластеризация — это метод, уходящий корнями в теорию графов, где этот подход используется для идентификации групп узлов в графе на основе соединяющих их ребер. Этот метод является гибким и позволяет кластеризовать и те данные, которые не являются графами. Спектральная кластеризация использует информацию из собственных значений (спектра) специальных матриц, построенных из графа или набора данных.

Описание алгоритма:
Пусть дан граф с l вершинами.
1) Строим лапласиан графа, равный разности матрицы степеней и матрицы смежности.
2) Из полученной матрицы находим нормированную систему из собственных векторов $u_1,...,u_k$, соответствующую минимальным собственным значениям.
3) Строим новую матрицу $U = (u_1|...|u_k)$. В ней по строчкам будет расположено описание l объектов, а каждый собственный вектор f_i записан в столбец.
4) Кластеризируем U с помощью, например, K-Means.

Спектральная кластеризация произвольных данных
Есть несколько способов рассматривать наши данные как граф. Самый простой способ - построить граф k-ближайших соседей. Граф k-ближайших соседей обрабатывает каждую точку данных как узел в графе. Затем от каждого узла к его ближайшим k соседям в исходном пространстве проводится ребро. Как правило, алгоритм не слишком чувствителен к выбору k. Меньшие числа, такие как 5 или 10, обычно работают очень хорошо.
Более общий подход - построить матрицу сходства. Матрица сходства похожа на матрицу смежности, за исключением того, что значение для пары точек выражает, насколько эти точки похожи друг на друга. Если пары точек очень непохожи, то сходство должно быть нулевым. Если точки идентичны, то сходство может быть равным 1. Таким образом, сходство действует как веса для ребер в графе.

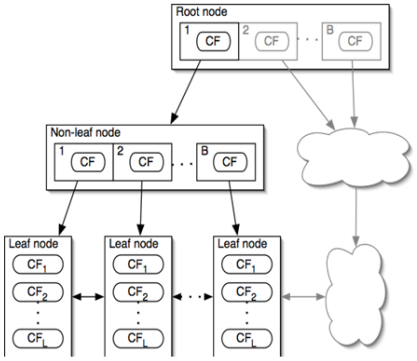
3.3.Birch

Алгоритм работает с большими наборами данных, сначала генерируя более компактную сводку, которая сохраняет как можно больше информации о распределении, а затем кластеризирует сводку данных вместо исходного набора. Birch дополняет другие алгоритмы кластеризации, так как различные алгоритмы могут быть применены к сводке, произведенной им.

Clustering Feature (CF):
Birch пытается минимизировать требования к памяти для больших наборов данных, суммируя информацию, содержащуюся в плотных областях, в виде записей *CF*.
$CF = (N, LS, SS)$, где N - число точек в кластере, LS - сумма всех точек, SS - сумма квадратов точек.
CF может быть составлен из других. Тогда $CF_1+CF_2=(N_1+ N_2, LS_1 + LS_2, SS_1 + SS_2)$.

CF-tree:
CF-дерево представляет собой очень компактное представление набора данных, поскольку каждая запись в листовом узле является не отдельной точкой данных, а подкластером. Каждый нелистовой узел содержит не более B записей. Одна запись

содержит указатель на дочерний узел и *CF*, состоящий из суммы *CF* в дочернем узле. Листовой узел содержит не более *L* записей, и каждая запись является *CF*. Все записи в листовом узле должны удовлетворять пороговым требованиям. То есть диаметр каждой листовой записи должен быть меньше порогового значения. Кроме того, каждый листовой узел имеет два указателя, *prev* и *next*, которые используются для объединения всех листовых узлов в цепочку для эффективного сканирования.



Описание алгоритма вставки:

1) Определить подходящий лист. Начиная с корня, рекурсивно спускаться по дереву, выбирая ближайший дочерний узел в соответствии с выбранной метрикой расстояния.

2) Изменить лист. По достижении листового узла найти ближайший элемент листа и проверить, может ли он вместить *CF* без нарушения порогового условия. Если это возможно, обновить запись *CF*, в противном случае добавьте новую запись *CF* в лист. Если в листе недостаточно места для этой новой записи, надо разделить листовой узел. Разделение узлов выполняется путем выбора двух наиболее удаленных друг от друга записей в качестве начальных значений и перераспределения оставшихся записей в зависимости от расстояния.

3) Преобразовать путь к листу. После вставки записи *CF* в лист надо обновить информацию для каждой нелистовой записи на пути к листу. В случае разделения надо вставить новую запись, не являющуюся листом, в родительский узел и указать ей на вновь сформированный лист. Если согласно выбранному *B*, у родителя недостаточно места, тогда надо разделить и родителя, и так далее до корня.

Описание алгоритма кластеризации:

1) Алгоритм начинается с начального порогового значения, он сканирует данные и вставляет точки в дерево. Если ему не хватает памяти до того, как он завершит сканирование данных, он увеличивает пороговое значение и перестраивает новое меньшее CF-дерево, повторно вставляя листовые записи старого дерева в новое. После того, как все старые листовые записи были повторно вставлены, сканирование данных и вставка в новое дерево возобновляются с точки, в которой оно было прервано. Правильный выбор порогового значения может значительно сократить количество перестроек. Однако, если начальный порог слишком высок, мы получим менее подробное дерево, чем это возможно с доступной памятью.

2) Применяем любой подходящий алгоритм глобальной кластеризации. Существующие алгоритмы кластеризации наборов точек могут также работать с наборами подкластеров, каждый из которых представлен своим *CF* вектором. Зная *CF* вектор, можно вычислить центроид и в дальнейшем заменить всю информацию о кластере этим значением (этого достаточно для вычисления большинства необходимых метрик). После фазы 3 мы получаем набор кластеров, который отражает основные характеристики распределяемых данных.

3) Улучшение кластеров. Использует центры тяжести кластеров, как основы. Перераспределяет данные между близкими кластерами. Данный этап гарантирует попадание одинаковых данных в один кластер.

Достоинства:
Двухступенчатая кластеризация, кластеризация больших объемов данных, работает на ограниченном объеме памяти, является локальным алгоритмом, может работать при одном сканировании входного набора данных.

Недостатки:
Работа с только числовыми данными, хорошо выделяет только кластеры сферической формы, есть необходимость в задании пороговых значений.

4. Архитектура программной системы

Архитектура системы состоит из следующих частей:

1. База данных (информация об объектах, информация о пользователях)
    a. Подсистема интеграции
2. Подсистема аналитики
    a. Алгоритмы (классификации)
3. REST веб-сервис
4. Клиент (веб-сайт - django, flask, vue.js, desktop, mobile)

5. Описание функциональных и нефункциональных требований к программному проекту

5.1. Функциональные требования

1. Предоставления возможности запуска и остановки работы программы
2. Поддержка загрузки конфигурации из файла конфигурации
3. Возможность авторизации с помощью логина и пароля
4. Отображение информации о подключенных устройствах
5. Отображение информации о носимых устройствах

5.2. Нефункциональные требования

1. Пользователю должен быть предоставлен непрерывный доступ к веб-приложению
2. Веб-сервис не должен непредвиденно прерывать свою работу
3. Надежное хранение информации о пользователях
4. Климатические условия эксплуатации, при которых должны обеспечиваться заданные характеристики, должны удовлетворять требованиям, предъявляемым производителем устройства, где используется программа

6. Список источников

1. K-Means Clustering in Python: A Practical Guide [Электронный ресурс] Режим доступа: https://realpython.com/k-means-clustering-python/, свободный. (дата обращения: 06.02.2021)
2. 2.3 Clustering [Электронный ресурс] Режим доступа:, https://scikit-learn.org/stable/modules/clustering.html, свободный. (дата обращения: 06.02.2021)
3. Birch Clustering algorithm example in Python [Электронный ресурс] Режим доступа: https://towardsdatascience.com/machine-learning-birch-clustering-algorithm-clearly-explained-fb9838cbeed9, свободный. (дата обращения: 06.02.2021)
4. Spectral Clustering [Электронный ресурс] Режим доступа: https://towardsdatascience.com/spectral-clustering-aba2640c0d5b#:~:text=Spectral%20clustering%20is%20a%20technique,non%20graph%20data%20as%20well, свободный. (дата обращения: 06.02.2021)
5. sklearn.cluster.SpectralClustering [Электронный ресурс] Режим доступа: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.SpectralClustering.html, свободный. (дата обращения: 07.02.2021)
6. sklearn.cluster.Birch [Электронный ресурс] Режим доступа: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.Birch.html, свободный. (дата обращения: 07.02.2021)
7. sklearn.cluster.KMeans [Электронный ресурс] Режим доступа: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html, свободный. (дата обращения: 07.02.2021)

7. Приложение 1. Календарный план

Месяц	Январь				Февраль				Март				Апрель				Май			
Неделя	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Обзор алгоритмов кластеризации																				
Разработка программы для выполнения подзадачи																				
Эксперименты на данных																				
Интеграция подсистем в единую систему																				