Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «**Информатика и системы управления**»

КАФЕДРА «**Программное обеспечение ЭВМ и информационные технологии**»

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ

НА ТЕМУ:

«Рекомендательная система новостей на основе нечеткой кластеризации»

Студент группы **ИУ7-82Б**	_____	**Чалый А. А.**
	(Подпись, дата)	(И.О. Фамилия)
Руководитель ВКР	_____	**Русакова З. Н.**
	(Подпись, дата)	(И.О. Фамилия)
Нормоконтроллер	_____	**Мальцева Д. Ю.**
	(Подпись, дата)	(И.О. Фамилия)

***2022** г.*

РЕФЕРАТ

Расчетно-пояснительная записка 70 с., 17 рис., 5 табл., 16 ист., 2 прил.

Ключевые слова: нечеткая кластеризация, рекомендательные системы, машинное обучение, обучение без учителя. Объектом исследования данной работы являются текста новостей.

Цель работы — разработать и реализовать рекомендательную систему на основе нечеткой кластеризации.

Для достижения поставленной цели необходимо решить следующие задачи:

- проанализировать предметную область;
- проанализировать существующие подходы в рекомендательных системах;
- на основе полученных во время анализа данных разработать собственную рекомендательную систему на основе нечеткой кластеризации;
- реализовать разработанный метод в программном продукте.

СОДЕРЖАНИЕ

ТЕРМИНЫ И УСЛОВНЫЕ ОБОЗНАЧЕНИЯ

Машинное обучение (Machine Learning, ML) — подраздел искусственного интеллекта, изучающий различные способы построения обучающихся алгоритмов. Среди множества парадигм и подходов в машинном обучении выделяются нейронные сети [1].

Обучение без учителя (Unsupervised learning) — один из разделов машинного обучения. Изучает широкий класс задач обработки данных, в которых известны только описания множества объектов (обучающей выборки), и требуется обнаружить внутренние взаимосвязи, зависимости, закономерности, существующие между объектами.

Кластеризация (Clustering) — 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.

Нечеткая кластеризация (Fuzzy clustering) — форма кластеризации, в которой каждая точка данных может принадлежать более чем одному кластеру.

Рекомендательные системы (Recommendation system) — комплекс алгоритмов, программ и сервисов, задача которого предсказать, что может заинтересовать того или иного пользователя.

Холодный старт (Cold start) — ситуация, когда система не может делать никаких выводов для пользователей или объектов, о которых она еще не собрала достаточно информации.

Датасет (Dataset) — набор данных, выборка.

ВВЕДЕНИЕ

За последние несколько десятилетий, с появлением Youtube, Amazon, Netflix и многих других подобных веб–сервисов, системы рекомендаций стали занимать все больше места в нашей жизни. Начиная с электронной коммерции (предлагая статьи, которые могут заинтересовать людей) и заканчивая рекламой в Интернете.

Многие современные сервисы создают рекомендательные системы, которые основываясь на информации о пользователе и его поведении в системе, пытаются определить какие объекты ему интересны, будь то товары, новости, услуги и т.д. Яркими примерами служат такие сервисы или сайты, как «КиноПоиск», «Яндекс.Дзен», «Яндекс.Новости» и многие другие. «КиноПоиск» — российский веб-сайт, предлагающий пользователю к просмотру фильмы на основе его предпочтений. «Яндекс.Дзен» — веб-сайт и расширение для браузера от компании «Яндекс», ищущее в интернете информацию, которая может быть интересна пользователю, и собирающее ее в персональную ленту. «Яндекс.Новости» — российский веб-сайт, предлагающий к просмотру новости от партнеров службы, в числе которых ведущие российские и зарубежные СМИ. Поступающая информация автоматически группируется в сюжеты. На их основе формируется информационная картинка дня.

Как видно из примеров, рекомендательные системы, улучшают пользовательский опыт, упрощают нахождение наиболее интересного для пользователя контента. Поэтому со временем, количество сервисов применяющих рекомендательные системы растет и начинает широко применятся во многих сферах, таких как электронная коммерция, при поиске фильмов, музыки, научных статей, а также на новостных сайтах и в справочных центрах, а задача разработки эффективных рекомендательных систем является актуальной.

Выделяют два основных метода построения рекомендательных систем — метод фильтрации на основе содержания и метод коллаборативной фильтрации.

Методы фильтрации на основе содержания основаны на описании объекта и профиле предпочтений пользователя. Данный подход пытается подобрать объекты, похожие на те, что нравились пользователю ранее, и опирается на методы информационного поиска и машинного обучения.

Метод коллаборативной фильтрации базируется на информации об истории поведения всех пользователей в системе. К примеру, если это сайт по продаже электроники, то рекомендация по покупке товаров основывается на пользователях со схожей историей и их отношениях к объекту.

Одним из направлений обработки данных различной структуры и свойств является кластеризация. Существует множество методов кластеризации, которые можно классифицировать как четкие и нечеткие. Четкие методы кластеризации разбивают исходное множество объектов на несколько непересекающихся подмножеств. При этом любой объект принадлежит только одному кластеру. Нечеткие методы кластеризации позволяют одному и тому же объекту принадлежать одновременно нескольким (или даже всем) кластерам, но с различной степенью принадлежности. Нечеткая кластеризация во многих ситуациях более “естественна”, чем четкая, например, для объектов расположенных на границе кластеров.

В данной работе должны быть проанализированы методы нечеткой кластеризации, существующие подходы к рекомендательным системам, а также потребуется разработать рекомендательную систему с применением алгоритма нечеткой кластеризации.

Целью работы является разработка и реализация рекомендательной системы новостей на основе нечеткой кластеризации методом Гауссовой смеси, а также проведение исследования работоспособности реализованной рекомендательной системы и разработанного алгоритма нечеткой кластеризации.

Для достижения поставленной цели необходимо решить следующие задачи:

- провести анализ предметной области, выделить основные определения;

- провести анализ существующих подходов реализации рекомендательных систем;
- выделить основные критерии для сравнения и выбора наиболее подходящего подхода рекомендательной системы и алгоритма нечеткой кластеризации для решения проблемы;
- в результате полученных во время анализа данных разработать рекомендательную систему на основе нечеткой кластеризации;
- реализовать выбранный алгоритм нечеткой кластеризации;
- реализовать рекомендательную систему в программном продукте;
- провести исследование работоспособности реализованной рекомендательной системы.

1 Аналитический раздел

1.1 Постановка задачи

Рассмотрим задачу рекомендательной системы новостей. На вход алгоритм получает данные о прочитанных ранее пользователям новостях. Выходом рекомендательной системы являются новости, рекомендованные к прочтению. Таким образом, задача рекомендации новостей представляет собой анализ ранее прочитанных пользователем новостей и рекомендации новостей на их основе.

Постановка задачи рекомендательной системы новостей представлена на рисунке 1.1.

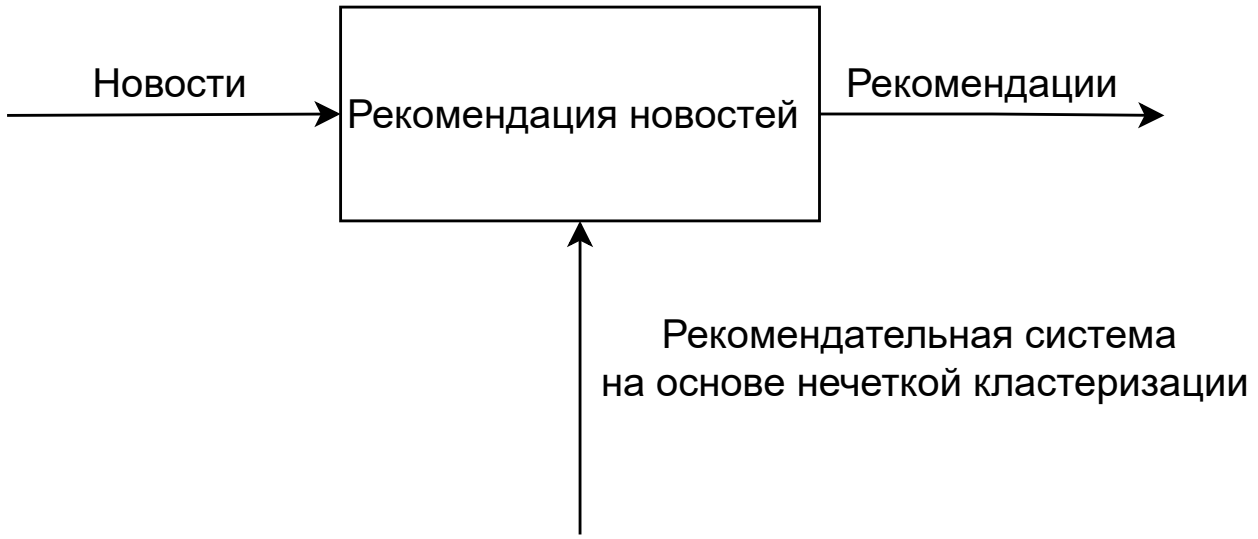

Рисунок 1.1 – Постановка задачи. Диаграмма верхнего уровня

1.2 Рекомендательная система

Рекомендательная система — это очень большая поддоменная область интеллектуального анализа данных. Выделяются два основных вида рекомендательных систем: персонализированная и неперсонализированная. Неперсонализированная рекомендательная система — это система, делающая рекомендации на основе рейтинга элемента. Преимуществом данного вида является простота реализации. Однако, предпочтения отдельного пользователя не учитыва-

ются, следовательно, сделанная рекомендация с маленько вероятностью понравится большей части пользователей.

Другой вид — персонализированная рекомендательная система учитывает предпочтения каждого пользователя, поэтому работает эффективнее. Такие системы основаны на расчете сходства между пользователями или элементами. Для построение персонализированной рекомендательной системы существует в основном два метода: фильтрация на основе содержания (контентная фильтрация) и коллаборативная фильтрация. [2]

1.3 Обучение без учителя

Обучение без учителя использует алгоритмы машинного обучения для анализа и кластеризации немаркированных наборов данных. Эти алгоритмы обнаруживают скрытые шаблоны или группы данных без необходимости вмешательства человека. Способность алгоритмов обнаруживать сходства и различия в информации делает их идеальным решением для исследовательского анализа данных, сегментации данных и рекомендаций на их основе.

Обучение без учителя используется в основном, для трех задач — кластеризации, ассоциации и уменьшения размерности. Ниже будет приведено описание кластеризации и уменьшения размерности и выделены общие алгоритмы и подходы для эффективной работы с ними.

1.3.1 Кластеризация

Кластеризация — это мощный инструмент машинного обучения для обнаружения структур и закономерностей в размеченных и неразмеченных наборах данных. Алгоритмы кластеризации используются для обработки необработанных, неклассифицированных объектов данных в группы, представленные структурами или шаблонами.

Цель кластерного анализа заключается в разделении объектов на группы с учетом сходств объектов. Кластеризацию можно считать наиболее важным методом обучения без учителя. Как и любой другой метод обучения без учителя, кластеризация не использует идентификаторы класса для выявления ос-

новной структуры в сборе данных. Кластер может быть определен как совокупность объектов, которые являются подобными друг для друга и неподобными для объектов, принадлежащих другим кластерам. Определение кластера может быть сформулировано по-разному, в зависимости от цели кластеризации. Можно дать общее определение кластера, согласно которому кластер представляет собой группу объектов, которые больше подобны друг другу, чем представителям других кластеров. Термин «подобие» может быть истолкован как математическое подобие, измеряемое в некотором определенном смысле. В метрических пространствах подобие часто определяется с помощью нормы расстояния или меры расстояния. Данные могут формировать кластеры с различными геометрическими формами, размерами и плотностями, как показано на рис. 1.2. Кластеры могут быть сферическими, вытянутыми и полыми.

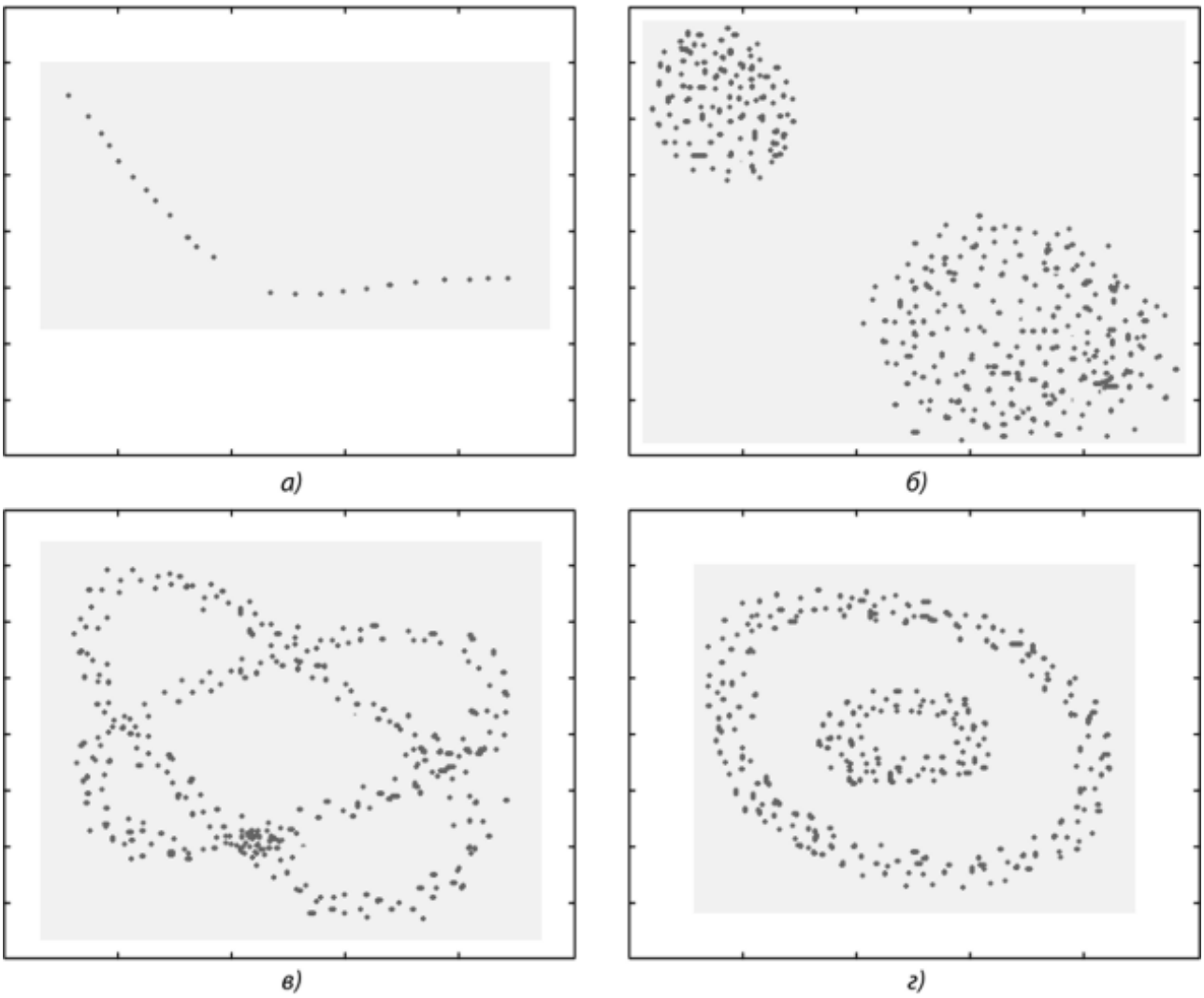

Рисунок 1.2 – Различные формы кластеров в R^2: *а* — вытянутые кластеры; *б* — сферические кластеры; *в*, *г* — полые кластеры

Кластеры могут существовать в любом д-мерном пространстве. Кластеры *а*, *в* и *г* можно охарактеризовать как линейные и нелинейные подпространства в пространстве данных (в данном случае R^2). Алгоритмы кластеризации способны обнаруживать подпространства в пространствах данных, поэтому они надежны для идентификации. [3]

Эффективность большинства алгоритмов кластеризации зависит не только от геометрической формы и плотности отдельных кластеров, но и от положения в пространстве и расстояния между кластерами. Кластеры могут быть разделены, связаны друг с другом или наложены друг на друга. Существует множество методов кластеризации, которые можно классифицировать как четкие и

нечеткие. Четкие методы кластеризации разбивают исходное множество объектов на несколько непересекающихся подмножеств. При этом любой объект принадлежит только одному кластеру. Нечеткие методы кластеризации позволяют одному и тому же объекты принадлежать одновременно нескольким (или даже всем) кластерам, но с различной степенью принадлежности. Нечеткая кластеризация во многих ситуациях более “естественна”, чем четкая, например, для объектов расположенных на границе кластеров. [4]

1.3.2 Нечеткая кластеризация

Нечеткая кластеризация (вероятностная) это метод, который помогает решать задачи оценки плотности. При нечеткой кластеризации точки данных группируются на основе вероятности их принадлежности к определенному распределению. Модель Гауссовой смеси (GMM) является одним из наиболее часто используемых вероятностных методов кластеризации.

Применение нечеткой кластеризации в качестве метода для рекомендательной системы новостей является актуальной задачей, так как каждый человек имеет свои предпочтения в прочтении новостей, вследствие чего он хочет видеть в рекомендациях новости, которые он хотел бы прочитать, но проблема состоит в том, что при использовании обычной кластеризации, где отсутствует связь кластеров, пользователю в скором времени начнут предлагаться новости одного типа, когда как при использовании нечеткой кластеризации пользовательские рекомендации бы обладали большей гибкостью. [5]

1.3.3 Принцип понижения размерности

Производительность алгоритмов машинного обучения может ухудшиться при слишком большом количестве входных переменных.

Если ваши данные представлены с помощью строк и столбцов, то входные переменные — это столбцы, которые передаются в качестве входных данных модели для прогнозирования целевой переменной. Столбцы данных можно рассматривать как измерения в n-мерном пространстве признаков, а строки данных как точки в этом пространстве. В таком случае набор данных, обладает

геометрической интерпретацией.

Наличие большого количества измерений в пространстве признаков может означать, что объем этого пространства очень велик, и, в свою очередь, точки, которые у нас есть в этом пространстве (строки данных), часто представляют собой небольшую и нерепрезентативную выборку. Это может существенно повлиять на производительность алгоритмов машинного обучения, подходящих для данных со многими входными характеристиками, обычно называемых «проклятием размерности». Поэтому зачастую прибегают к уменьшению количества входных признаков.

Многомерность может означать сотни, тысячи или даже миллионы входных переменных. Меньшее количество входных измерений часто означает соответственно меньшее количество параметров или более простую структуру в модели машинного обучения, называемую степенями свободы. Модель со слишком большим количеством степеней свободы, будет переобучаться на обучающем наборе данных и, следовательно, может плохо работать с новыми данными и предсказывать неправильный результат. В данном случае и прибегают к методам понижения размерности, которые, тем не менее, хорошо обобщают набор входных данных. Одними из основных методов понижения размерности являются метод главных компонент (PCA) и метод сингулярного разложения (SVD). Ниже будет приведено изложение данных методов.

1.3.4 Метод главных компонент (PCA)

Метод главных компонент (PCA) — это тип алгоритма уменьшения размерности, который используется для уменьшения избыточности и сжатия наборов данных посредством извлечения признаков. Этот метод использует линейное преобразование для создания нового представления данных, что дает набор «основных компонентов». Первый главный компонент — это направление, которое максимизирует дисперсию набора данных. Второй главный компонент также обнаруживает максимальную дисперсию данных, но совершенно не коррелирует с первым главным компонентом, что дает направление, перпен-

дикулярное или ортогональное первому компоненту. Этот процесс повторяется в зависимости от количества измерений, где следующий главный компонент является направлением, ортогональным предыдущим компонентам с наибольшей дисперсией.

1.3.5 Метод сингулярного разложения (SVD)

Метод сингулярного разложения (SVD) — это подход к уменьшению размерности, который раскладывает матрицу A на три матрицы низкого ранга. SVD приведено в формуле 1.

$$A = USV^T, \tag{1}$$

где U и V — ортогональные матрицы. S — диагональная матрица, а значения S считаются сингулярными значениями матрицы A. Подобно PCA, он обычно используется для уменьшения шума и сжатия данных.

Сингулярное разложение является удобным методом при работе с матрицами. Оно показывает геометрическую структуру матрицы и позволяет наглядно представить имеющиеся данные. Сингулярное разложение используется при решении самых разных задач — от приближения методом наименьших квадратов и решения систем уравнений до сжатия изображений. При этом используются разные свойства сингулярного разложения, например, способность показывать ранг матрицы, приближать матрицы данного ранга. SVD позволяет вычислять обратные и псевдообратные матрицы большого размера, что делает его полезным инструментом при решении задач регрессионного анализа.

1.4 Обработка естественных языков

Обработка естественного языка (Нейролингвистическое программирование (NLP)) относится к области компьютерных наук, в частности к области искусственного интеллекта или ИИ. Ее цель заключается в предоставлении компьютерам возможности понимать текст и произносимые слова почти так же, как люди.

Обработка естественного языка сочетает в себе вычислительную лингвистику — моделирование человеческого языка на основе правил — со статистическими моделями, машинным обучением и моделями глубокого обучения. Вместе эти технологии позволяют компьютерам обрабатывать человеческий язык в виде текстовых или голосовых данных и «понимать» его полное значение, включая намерения и чувства говорящего или пишущего. Она управляет компьютерными программами, которые переводят текст с одного языка на другой, реагируют на голосовые команды и быстро резюмируют большие объемы текста — даже в режиме реального времени.

Для использования текста в алгоритмах машинного обучения, первоначально следует обработать сам текст, а именно произвести токенизацию, после чего произвести стемминг или лемматизацию.

1.4.1 Токенизация

Токенизация — это первый шаг при разработке NLP приложения. Токенизатор разбивает неструктурированные данные и текст на естественном языке на фрагменты информации, которые можно рассматривать как отдельные элементы. Вхождения токена в документе можно использовать непосредственно как вектор, представляющий этот документ. Токенизация превращает неструктурированную строку (текстовый документ) в числовую структуру данных, пригодную для машинного обучения.

Существуют различные инструменты токенизации текста, каждый из которых подходит под определенные задачи, основными токенизаторами являются: NLTK, TextBlob, spacy, gensim и Keras.

1.4.2 Стемминг

Стемминг является более простым подходом, по сравнению с лемматизацией. Это способ подготовки текста для использования в модели машинного обучения, сокращение слов до своих грамматических основ (основа слова ”Африки”– ”Африк”). Основа слова — стем, не обязательно совпадает с корнем, он может включать и суффиксы. Это неизменяемая при склонении часть.

Алгоритмы стемминга обычно основаны на правилах: слово проходит через ряд условных предложений, которые определяют, как его сократить. Например, существует правило суффиксов: в английском языке «-ed» и «-ing» отрезают, чтобы сопоставить ”cooking”и ”cooked”с одной и той же основой ”cook”.

Поскольку стемминг обычно основан на эвристике, он далек от совершенства. Данный подход обладает такими недостатками как перестемминг и недостемминг.

Перестемминг (англ. overstemming) происходит, когда слишком большая часть слова обрезается. Это может привести к бессмысленным стемам, где значение слова потеряно. Или же к тому, что совершенно неродственные слова будут приведены к одной и той же основе.

Недостемминг — противоположная проблема. Подобная ситуация возникает, когда у нас есть несколько слов, которые на самом деле являются формами друг друга, но их основа слова (stem), оказываются отличными.

1.4.3 Лемматизация

Лемматизация — объединение слов с одним и тем же корнем или леммой, но с разными склонениями или производными значения для дальнейшего анализа как элемента. Цель состоит в том, чтобы выявить присутствие слова в любой из его форм в текстовом блоке (корпусе) и, например, определить частоту его появления.

Подобный подход является более мощным инструментом, так как учитывает морфологический анализ слов. Он возвращает лемму, которая является базовой формой всех ее флективных форм. Для создания словарей и поиска правильной формы слова необходимы глубокие лингвистические знания. Стемминг — это общая операция, а лемматизация — интеллектуальная операция, в которой правильная форма будет выглядеть в словаре. Следовательно, лемматизация помогает в формировании лучших возможностей машинного обучения.

1.4.4 Векторизация текстовых данных

Векторизация — процесс конвертации текста в числа. После предобработки данных, следует шаг векторизации.На данном шаге требуется закодировать текстовые данные в виде чисел, которые в дальнейшем будут использованы в алгоритмах.

Наиболее часто используемыми векторизаторами являются:

- векторизация методом мешка слов (Bag of words (BOW));
- векторизация методом TF-IDF;
- векторизация методом word embedding (векторное представление слов).

Описание каждого из приведенных в списке выше векторизаторов приведено ниже.

1.4.5 Векторизация методом мешка слов (Bag of words (BOW))

Модель мешка слов является одной из самых простых методик векторизации текста. Данный метод создает словарь уникальных слов в корпусе (собрание всех токенов в данных), после чего создается таблица в которой столбцы соответствуют входящим в корпус уникальным словам, а строки предложениями, далее происходит инициализация ячеек 0 если слова в предложении нет и 1 иначе.

1.4.6 Векторизация методом TF-IDF

Векторы слов с использованием данного метода высчитываются из двух коэффициентов, а именно Term Frequency (Частота слова) и Inverse Document Frequency (Обратная частота документа).

Term Frequency высчитывает вероятность встретить слово в документе, по сравнению с общим количеством слов в документе. Пример для терма w_i в документе d_j приведен в формуле 2.

$$\text{Term Frequency}(w_i, d_j) = \frac{\text{количество } w_i \text{ в } d_j}{\text{количество термов в } d_j} \tag{2}$$

Inverse Document Frequency (обратная частота документов) отражает до-

лю документов в корпусе, содержащих этот термин. Слова, уникальные для небольшого процента документов (например, термины технического жаргона), получают более высокие значения важности, чем слова, общие для всех документов, к примеру местоимения, если они не были удалены на предобработке. Вычисление IDF приведено в формуле 3.

$$\begin{gathered}\text{Inverse Document Frequency} = \\ = \log\left(\frac{\text{количество вхождений терма в документ}}{\text{количество документов, которые содержат терм в корпусе}}\right)\end{gathered} \tag{3}$$

Итоговая формула 4 представляет собой произведение TF на IDF, формулы которых 2 и 3.

$$\text{TF-IDF} = TF * IDF \tag{4}$$

1.4.7 Векторизация методом word embedding (векторное представление слов)

Векторное представление слов — это представление текста, в котором слова с одинаковым значением имеют сходное представление.

Данный подход к представлению слов и документов можно считать одним из ключевых прорывов в области глубокого обучения при решении сложных задач обработки естественного языка.

Word embedding на самом деле представляет собой класс методов, в которых отдельные слова представлены в виде векторов с действительными значениями в предопределенном векторном пространстве. Каждое слово сопоставляется с одним вектором, а значения векторов изучаются способом, напоминающим нейронную сеть, поэтому этот метод часто относят к области глубокого обучения. Основной подход основан на идее использования плотного распределенного представления для каждого слова.

Каждое слово представлено действительным вектором, часто с десятками или сотнями измерений. Это контрастирует с тысячами или миллионами измерений, необходимых для представления разреженных слов.

Наиболее популярным методом, который преобразует слова в векторное представление является Word2Vec.

1.5 Классификация подходов к реализации рекомендательных систем

Подходы к реализации рекомендательных систем можно классифицировать на:

- фильтрация на основе содержания;
- коллаборативная фильтрация.

Далее будет приведено описание каждого из подходов, а также описаны подразделы подхода к решению рекомендательных систем под названием коллаборативная фильтрация. Для фильтрации на основе содержания будет приведены методы, позволяющие построить рекомендательную систему, основанную на данном подходе.

1.5.1 Фильтрация на основе содержания

Фильтрация на основе содержания (content–based filtering (CBF)) рекомендует элементы на основе профиля пользователя, который он создает при регистрации в системе. CBF сравнивает понравившиеся пользователю элементы, или в случае с новостями, те новости, которые он прочел полностью с другими новостями и выбирает из них аналогичные. Преимущество метода является то, что если у пользователя необычный вкус, то система порекомендует подходящие ему элементы, без учета рейтинга. Кроме того, для достижения высокой точности рекомендаций не нужна большая группа пользователей, а также новые, еще не имеющие рейтинг элементы, могут быть сразу порекомендованы. К недостаткам относят проблему нового пользователя: когда в системе появляется новый пользователь, нет информации о его предпочтениях, и как результат, может быть сделана плохая рекомендация. Другая проблема состоит в том, что система рекомендует только элементы, которые пользователю понравились ранее, и она не сможет порекомендовать что–то нетипичное для пользователя.

Для построения подобных систем применяются наивный байесовский клас-

сификатор и другие различные методы машинного обучения, включая кластеризацию, деревья решения и нейронные сети.

1.5.2 Коллаборативная фильтрация

Коллаборативная фильтрация (collaborative filtering (CF)) — это подход, основанный на предпочтениях группы пользователей. Преимуществом метода является универсальность. Кроме того, для работы данного метода достаточно знать историю оценок целевого пользователя и похожих на него пользователей. Данный тип фильтрации зачастую сталкивается с проблемой холодного старта: система зависит от аналогичных “соседей”, но так как они недоступны на начальном этапе, то система делает плохие рекомендации. Кроме того, отмечается проблема первого оценщика: система не может рекомендовать новость, которая ранее не была просмотрена. Еще одной проблемой является то, что если есть много элементов, то матрица пользователей/просмотров разрежена, и трудно найти пользователей, которые оценили одни и те же элементы. Кроме того, система не может рекомендовать элементы кому–то с уникальными вкусами. [6]

Системы CF могут быть подразделены на системы, основанные на эвристических методах (memory/heuristic–based) и на построении моделей предпочтения (model–based).

В подходе на основе эвристических методов (memory/heuristic–based) прогноз рейтинга делается с учетом всех оцененных пользователем ранее элементов. Данный подход, в свою очередь подразделяется на User–based коллаборативную фильтрацию и Item–based коллаборативную фильтрацию.

Суть User–based коллаборативной фильтрации заключается в выборе подмножества пользователей на основе их сходства, после чего взвешенная комбинация из рейтингов используется для предсказания рейтинга, который поставит отдельный пользователь. В общем случае:

1) Присвоить вес (мера сходства) всем пользователям. Вес $w_{a,u}$ используется для вычисления сходства между пользователем u и пользователем a.

Наиболее распространенный метод — коэффициент корреляции Пирсона для измерения сходства двух пользователей приведен в формуле 5.

$$w_{a,u} = \frac{\sum_{i \epsilon I}(\gamma_{a,i} - \bar{\gamma_a})(\gamma_{u,i} - \bar{\gamma_u})}{\sqrt{\sum_{i \epsilon I}(\gamma_{a,i} - \bar{\gamma_a}) \sum_{i \epsilon I}(\gamma_{u,i} - \bar{\gamma_u})}}, \quad (5)$$

где I — множество элементов, оцененных обоими пользователями, $\gamma_{u,i}$ — рейтинг, поставленный элементу i пользователем u, а γ_u — средняя оценка, которую ставит пользователь u.

2) Выбрать число K — количество похожих пользователей.
3) Рассчитать предсказание для целевого пользователя на основе весовой функции и оценок K — ближайших пользователей. Рассчет предсказания приведен в формуле 6.

$$P_{a,i} = \bar{\gamma_a} + \frac{\sum_{u \epsilon K}(\gamma_{u,i} - \bar{\gamma_u}) * w_{a,u}}{\sqrt{\sum_{u \epsilon K} w_{a,u}}}, \quad (6)$$

где $P_{a,i}$ — прогноз рейтинга, поставленного пользователем a на элемент i, $w_(a,u)$ — сходство между пользователями a и u, а K — соседи, то есть набор наиболее похожих пользователей.

По мере роста системы количество пользователей увеличивается, и соответственно сложность поиска похожих пользователей возрастает. Потому был предложен новый подход, который вместо похожих пользователей ищет похожие элементы.

Метод Item–based коллаборативной фильтрации также основан на коэффициенте корреляции Пирсона, однако рассчитывается схожесть пары элементов i и j.

Подход, основанный на модели предпочтений, предполагает, что сходство между пользователями и элементами вызвано некоторой скрытой низкоразмерной структурой данных. В данном подходе одна предварительная модель разрабатывается на основе имеющихся данных. Когда появляется запрос от пользователя, этот подход дает быстрый ответ о предпочтениях пользовате-

ля.

Поскольку построение модели часто является трудоемким и ресурсоемким процессом, обычно сложнее добавлять данные в системах, основанных на моделях, что делает их негибкими. Кроме того, в связи с тем, что не используется весь набор данных, полученные прогнозы могут быть менее точными, чем в системах с эвристическим подходом.

1.5.3 Сравнение подходов в рекомендательных системах

В таблице 1 приведено сравнение основных подходов к построению рекомендательных систем. Для достижения лучших результатов некоторые рекомендательные системы сочетают различные методы коллаборативных подходов и подходов, основанных на содержании.

Таблица 1 – Подходы к построению рекомендательных систем

Подход	Преимущества	Недостатки
Неперсонализированный подход	Простота реализации.	Не учитывает предпочтения отдельного пользователя.
Фильтрация на основе содержания	Учет необычных вкусов пользователей. Не нужна большая группа пользователей. Возможность рекомендовать неоцененные/непросмотренные элементы.	Проблема нового пользователя. Отсутствие разнообразия в рекомендациях.
Коллаборативная фильтрация	Универсальность. Разнообразие рекомендации. Не нужно много информации о пользователях и элементах.	Проблема холодного старта. Проблема первого оценщика. Разреженность. Смещения популярности.

1.6 Классификация алгоритмов нечеткой кластеризации

Алгоритмы нечеткой кластеризации полезны, когда существует набор данных с подгруппами точек, имеющих нечеткие границы и перекрывающихся между кластерами. Стандартные методы нечеткой кластеризации требуют, чтобы пользователи заранее обладали знаниями о ожидаемом результате, чтобы

определить, сколько кластеров искать. Также существуют итерационные алгоритмы нечеткой кластеризации, которые выбирают оптимальное количество кластеров, основываясь на одном из нескольких показателей производительности. [7]

Далее будет приведено описание и сравнительный анализ двух алгоритмов (алгоритм нечетких средних, алгоритм Гауссовой смеси).

1.6.1 Метод нечетких средних (Fuzzy C-means (FCM))

Алгоритм нечетких средних (Fuzzy C-Means (FCM)) позволяет разбить имеющееся множество элементов мощностью N на заданное число нечетких множеств k. Метод нечеткой кластеризации С–средних можно рассматривать как усовершенствованный метод k–средних, при котором для каждого элемента из рассматриваемого множества рассчитывается степень его принадлежности каждому из кластеров.

Данный алгоритм работает, присваивая каждой точке данных определенный кластер, соответствующей каждому центру кластера на основе расстояния между центром кластера и точкой данных. Чем ближе данные к центру кластера, тем больше принадлежность к конкретному кластерному центру. Сумма членства каждой точки ко всем кластерам равна 1. После каждой итерации членство и центры кластеров обновляются по формулам 7 и 8.

$$\mu_{ij} = \frac{1}{\sum_{k=1}^{c} \left(\frac{d_{ij}}{dik}\right)^{\left(\frac{2}{m-1}\right)}}, \tag{7}$$

$$v_j = \frac{\sum_{i=1}^{n} (\mu_{ij})^m x_i}{\sum_{i=1}^{n} (\mu_{ij})^m}, \forall j = 1, 2, ...c, \tag{8}$$

где n это количество точек, v_j, центр j-го кластера, m это индекс нечеткости $m \epsilon [1, \text{inf}]$, c это количество центров кластеров, μ_{ij} степень принадлежности i-ой точки к j-ому центру кластера, а d_{ij} представляет собой Евклидово расстояние между i-ой точкой к j-ым центром кластера.

Основной задачей алгоритма является минимизация функции, приведенной в формуле 9.

$$J(U,V) = \frac{i=1}{n}\frac{j=1}{c}(\mu_{ij})^m \left\|x_i - v_j\right\|^2, \tag{9}$$

где $\|x_i - v_j\|$ это Евклидово расстояние между i-ой точкой к j-ым центром кластера.

Минимизацию С–средних можно рассматривать как нелинейную задачу оптимизации, которая может быть решена с помощью различных методов. Примерами методов, позволяющих решать нелинейные задачи оптимизации, являются групповая координатная минимизация и генетический алгоритм. Наиболее простым способом решения этой задачи является использование итерации Пикара через условия первого порядка для стационарных точек уравнения. Этот метод называется алгоритмом оптимизации нечетких С–средних. [8]

1.6.2 Модель Гауссовой смеси (Gaussian mixture model (GMM))

В реальной жизни многие наборы данных можно смоделировать с помощью распределения Гаусса (одномерного или многомерного). Поэтому вполне естественно и интуитивно предполагать, что кластеры происходят из разных распределений Гаусса. Или, другими словами, данная модель пытается смоделировать набор данных как смесь нескольких распределений Гаусса.

GMM можно использовать для поиска кластеров в наборах данных, где кластеры могут быть нечетко определены. Кроме того, модель Гауссовой смеси можно использовать для оценки вероятности того, что новая точка данных принадлежит каждому кластеру. Смешанные модели Гаусса также относительно устойчивы к выбросам, а это означает, что они могут давать точные результаты, даже если есть некоторые точки данных, которые не вписываются точно ни в один из кластеров. Это делает GMM гибким и мощным инструментом для кластеризации данных.

В смешанных гауссовских моделях метод максимизации ожидания

(expectation-maximization (EM)) является инструментом для оценки параметров Гауссовой смешанной модели (GMM). Ожидание (E) используется для нахождения гауссовских параметров, которые используются для представления каждого компонента моделей Гауссовой смеси. Максимизация (M) участвует в определении того, можно ли добавлять новые точки данных или нет.

Метод максимизации ожидания представляет собой двухэтапный итерационный алгоритм, который чередуется между выполнением шага ожидания (E), в котором вычисляются ожидания для каждой точки данных, используя текущие оценки параметров, а затем максимизируем их для получения нового гауссова, за которым следует шаг максимизации (M) в котором происходит обнолвение Гауссовых средних на основе оценки максимального правдоподобия. Метод EM работает, сначала инициализируя параметры GMM, а затем итеративно улучшая эти оценки. На каждой итерации шаг ожидания вычисляет математическое ожидание функции логарифмического правдоподобия по отношению к текущим параметрам. Затем это ожидание используется для максимизации правдоподобия на шаге максимизации. Затем процесс повторяется до сходимости.

1.6.3 Сравнение алгоритмов нечеткой кластеризации

В таблице 2 приведено сравнение алгоритмов нечеткой кластеризации.

Таблица 2 – Сравнение алгоритмов нечеткой кластеризации

Метод	Преимущества	Недостатки
Метод нечетких средних (FCM)	Скорость работы	Необходимо изначально знать количество кластеров, плохо работает с данными разных размеров и плотностей, чувствителен к выбросам.
Модель Гауссовой смеси (GMM)	Выявляет кластеры различных форм, размеров и плотностей.	Работает медленнее чем FCM.

1.7 Выводы из аналитического раздела

В данном разделе была произведена постановка задачи, после чего приведено обьяснение рекомендательных систем, методов обучения без учителя, а также приведена основная информация о обработке естественных языков. В качестве векторизатора слов было принято решение взять TF-IDF, т.к. он обладает наилучшим соотношением скорости и качества работы для данной задачи, а в качестве метода понижения размерности выбрано сингулярное разложение матриц так как оно лучше работает с разреженной матрицей и больше подходит в качестве предобработки данных для последующей кластеризации. Также была произведена классификация подходов при разработке рекомендательных систем и их сравнительный анализ, в ходе которого было принято решение разрабатывать рекомендательную систему с использование фильтрации на основе содержания. Далее были рассмотрены методы нечеткой кластеризации и про-

веден их сравнительный анализ, после которого было решено, что следует использовать модель Гауссовой смеси т. к. она является наиболее подходящей для поставленной задачи.

2 Конструкторский раздел

2.1 Декомпозиция разрабатываемой рекомендательной системы

Разрабатываемая рекомендательная система состоит из этапов.

- предобработка входных данных;
- векторизация предобработанных данных;
- понижение размерности полученной матрицы;
- разделение новостей на кластеры;
- рекомендация новостей на основе данных, выделенных в кластеры.

Ниже представлена IDEF0-диаграмма разрабатываемого метода на рисунке 2.3.

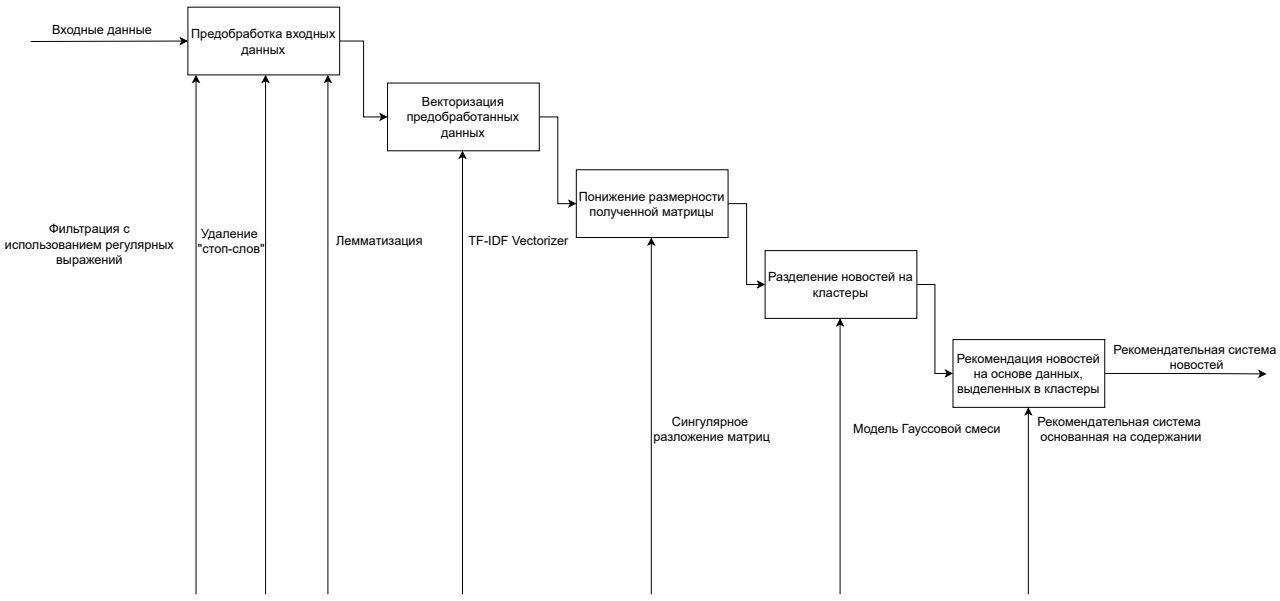

Рисунок 2.3 – IDEF0-диаграмма

2.2 Предобработка входных данных

Данный этап предназначен для подготовки к дальнейшему обучению входных данных. На вход поступают структуры из json-файла, которые содержат информацию о новостях. Пример входных данных приведен в таблице 3.

Таблица 3 – Пример входных данных

ID	Заголовок	Абстрактное описание	Новость	Дата публикации
N45436	Walmart Slashes Prices on Last-Generation iPads	Apple’s new iPad releases bring big deals on last year’s models...	This year, Walmart’s not waiting until to offer steep deals on tech. Right now, you can save big on since new models for 2019...	10/29/2019
N23144	50 Worst Habits For Belly Fat	These seemingly harmless habits are holding you back and keeping you from shedding that unwanted belly fat for good.	When you first start dieting and exercising, the pounds seem to melt off. But, we all hit that stagnant point where the last few pounds of belly fat just don’t want to leave...	5/7/2019

Для того, чтобы корректно произвести векторизацию и последующее обучение модели, данные следует предобработать следующим образом:

- объединить столбцы с заголовком, абстрактным описанием и самой новостью;
- удалить все символы, кроме кириллических;
- удалить все ”стоп-слова”;
- провести лемматизацию.

После выполнения данного этапа будет получен массив предложений в которых содержится только необходимая для обучения информация, а все данные не несущие смысловую нагрузку удалены.

2.3 Векторизация предобработанных данных

Для векторизации полученных данных используется терм-документная частота (TF-IDF). Вычисление TF-IDF состоит из трех этапов:

- вычисление TF;
- вычисление IDF;
- Произведение TF и IDF и получение TF-IDF.

TF (term frequency) позволяет оценить важность терма в отдельно взятом документе. TF вычисляется по формуле 10.

$$TF(w, d) = \frac{n_w}{\sum_i n_i}, \tag{10}$$

где n_w — количество вхождений терма w в документ d, $\sum_i n_i$ — количество слов в документе.

Инверсная документная частота (Inverse Document Frequency (IDF)) необходима для уменьшения веса широко употребляемых слов. Вычисляется по формуле 11.

$$IDF(w_i, D) = \log\left(\frac{|D|}{|D_i|}\right), \tag{11}$$

где $|D|$ — это общее количество документов, а $|D_i|$ — это число докумен-

тов, где w_i встретилось хотя бы раз.

Получаем что TF-IDF вычисляется по формуле 12.

$$TF - IDF(w_i, D) = TF(w, d) * IDF(w_i d, D) \tag{12}$$

Векторизация подобным образом очень эффективна для последующей задачи кластеризации, так как значимые термы, встречающиеся в пределах одного документа, но редко употребляемые во всем корпусе, имеют наибольший вес.

Результатом выполнения данного этапа является матрица, строки которой — это документ (новость), а столбцы — это все термы документов (новостей). В каждой ячейке хранятся TF-IDF для конкретного терма. Так как корпус состоит из большого количества уникальных слов, то матрица получается разреженная, а также слишком большого размера.

2.4 Понижение размерности матрицы признаков

На данном этапе производится сингулярное разложение, полученной на предыдущем шаге матрицы признаков. Данное действие обусловлено тем, что вычислительные мощности моего оборудования не позволяют обрабатывать таблицу подобного размера при проведении кластеризации. Также это сделано для увеличения скорости работы алгоритма нечеткой кластеризации, что немаловажно.

2.5 Разделение новостей на кластеры

После понижения размерности выполняется этап нечеткой кластеризации новостей из полученной матрицы. В качестве алгоритма нечеткой кластеризации используется модель Гауссовой смеси (Gaussian Mixture Model (GMM)), алгоритм работы которого приведен ниже.

В данном алгоритме каждый кластер представляется параметрическим распределением, а весь набор данных моделируется смесью этих распределе-

ний, следовательно для Гауссовой смеси получаем формулу 13.

$$P(\mathbf{x}|\Theta) = \sum_{i=1}^{K} \alpha_i p_i(\mathbf{x}|\theta_i), \tag{13}$$

где параметры $\Theta = (\alpha_1, ..., \alpha_K, \theta_1, ..., \theta_K)$ такие что $\sum_{i=1}^{K} \alpha_i = 1$ и каждое p_i является функцией плотности Гаусса параметризованной по θ_i. Другими словами, мы предполагаем, что у нас есть K плотностей компонентов, смешанных вместе с K коэффициентами смешения α_i.

Пусть $\mathcal{X} = (x_1, ..., x_m)$ — это набор точек данных. Требуется найти такое Θ, чтобы $p(\mathcal{X}|\Theta)$ было максимальным. Подобная задача известна как оценка максимального правдоподобия для Θ. Для оценки Θ обычно вводят логарифмическую функцию правдоподобия, определяемая по формуле 14.

$$\mathcal{L}(\Theta) = \log P(\mathcal{X}|\Theta) = \log \prod_{i=1}^{m} P(\mathbf{x}_i|\Theta) = \sum_{i=1}^{m} \log \left(\sum_{j=1}^{K} \alpha_j p_j(\mathbf{x}_i|\theta_j) \right) \tag{14}$$

Подобную функцию трудно оптимизировать, поскольку она содержит логарифм суммы. Для упрощения выражение правдоподобия, пусть $y_i \epsilon 1, \ldots, K$ обозначает, из какого Гауссиана x_i, и $\mathcal{Y} = (y1, ..., ym)$. Если мы знаем значение $\mathcal{Y}$, получаем формулу 15.

$$\begin{gathered} \mathcal{L}(\Theta) = \log P(\mathcal{X}, \mathcal{Y}|\Theta) = \log \textstyle\prod_{i=1}^{m} P(\mathbf{x}_i, y_i|\Theta) = \\ = \textstyle\sum_{i=1}^{m} \log P(\mathbf{x}_i|y_i) P(y_i) = \sum_{i=1}^{m} \log(\alpha_{y_i} p_{y_i}(\mathbf{x}_i|\theta_{y_i})) \end{gathered} \tag{15}$$

которая впоследствии оптимизируется с помощью различных методов, самым популярным из которых является алгоритм максимизации ожидания.

Исходя из названия становится ясно, что алгоритм состоит из двух частей, а именно вычисления ожидания (E), которое приведено в формулах 17, и вычисления максимизации (M), которое приведено в формулах 19, 20 и 21. Также используются вспомогательные формулы такие как 18 и 16.

$$f(x; \mu, \sum) = \frac{1}{\sqrt{(2\pi)^d * \det(\sum)}} * e^{(\frac{-1}{2} * ((x-\mu)^T * inv(\sum) * (x-\mu)))}, \tag{16}$$

где d — длина вектора x, x это выходной вектор, μ вектор средних, а $\sum$ это матрица ковариаций.

$$r_{n,k} = \frac{\pi_k N(x_n|\mu_k, \sum_k)}{\sum_j \pi_j N(x_n|\mu_j, \sum_j)}, \tag{17}$$

где π_k и pi_j — это отношения количества элементов в кластере k и j соответственно ко всем элементам данных, x_n это входной вектор данных, μ_k и μ_j это вектор средних для столбцов k и j, вычисляемый по формуле 16, а $\sum_k$ и $\sum_j$ это матрица ковариаций элементов в кластерах k и j.

$$N = \sum_k r_{n,k} \tag{18}$$

По формуле ожидания видно, что мы получаем матрицу в которой строки — это каждый элемент данных, а столбец представляет кластер, следовательно каждый элемент данной матрицы это вероятность принадлежности элемента данных к столбцу. После того как алгоритм сойдется, данные из этой матрицы будут использованы в качестве предсказания точки кластера. Также на данном шаге вычисляется N по формуле 18, которое представляет из себя список сумм в столбце матрицы $r_{n,k}$.

Схема алгоритма шага ожидания (E) приведена на рисунке 2.4.

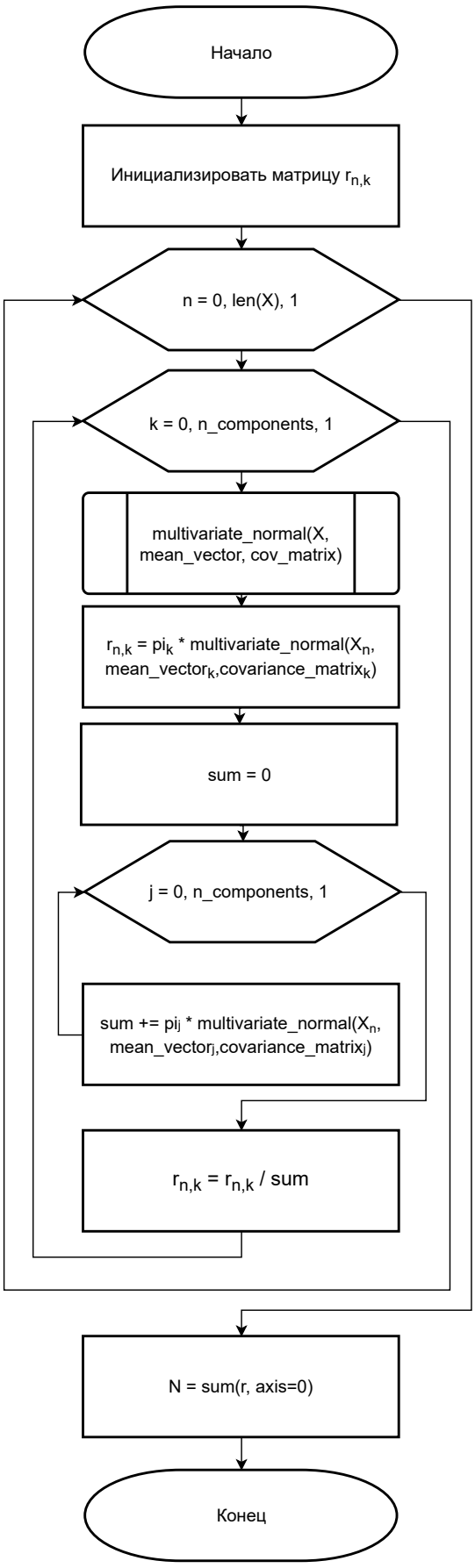

Рисунок 2.4 – Схема алгоритма шага ожидания

Вычисления максимизации (M), приведено в формулах 19, 20 и 21.

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^{N} r_{n,k} x_n, \tag{19}$$

$$\sum_k = \frac{1}{N_k} \sum_{n=1}^{N} r_{n,k} (x_n - \mu_k)(x_n - \mu_k)^T, \tag{20}$$

$$\pi_k = \frac{N_k}{N}. \tag{21}$$

В формуле 19 вычисляется новый вектор средних для каждого столбца, в формуле 20 происходит обновление матрицы ковариаций для каждого столбца, а в формуле 21 обновляется список отношений количества элементов в кластере ко всем элементам данных.

Схема алгоритма шага максимизации (M) приведена на рисунке 2.5.

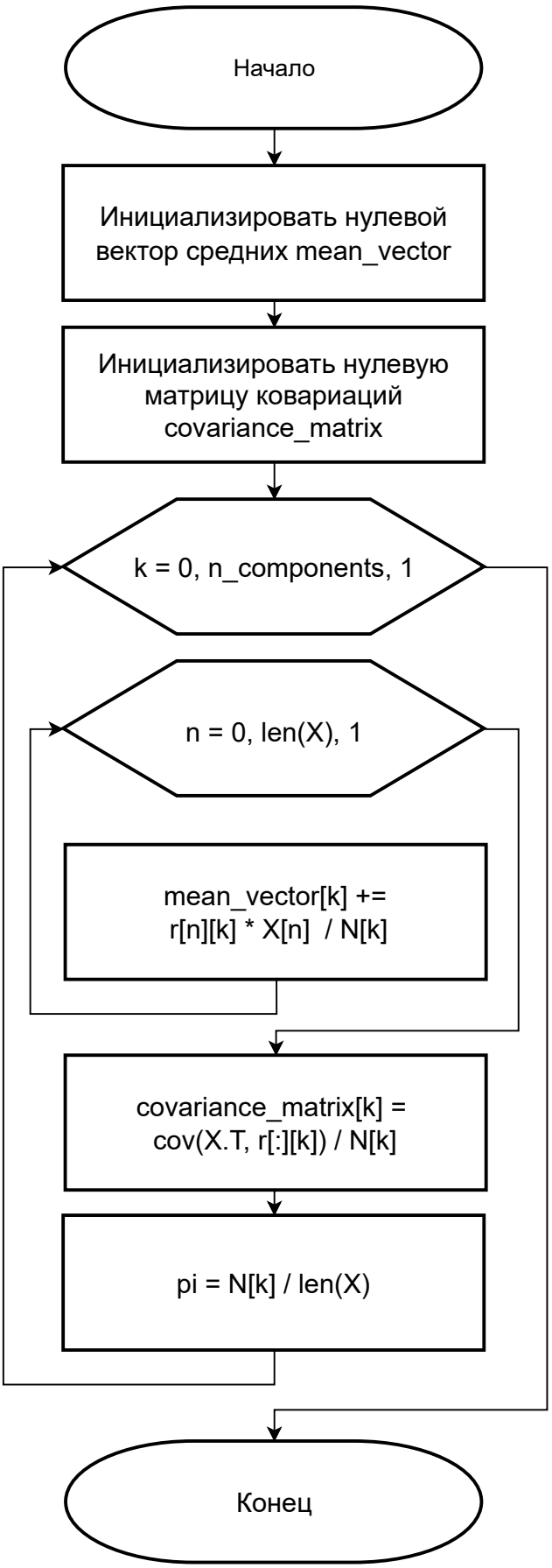

Рисунок 2.5 – Схема алгоритма шага максимизации

Схема алгоритма метода максимизации ожидания для модели Гауссовой смеси представлена на рисунке 2.6.

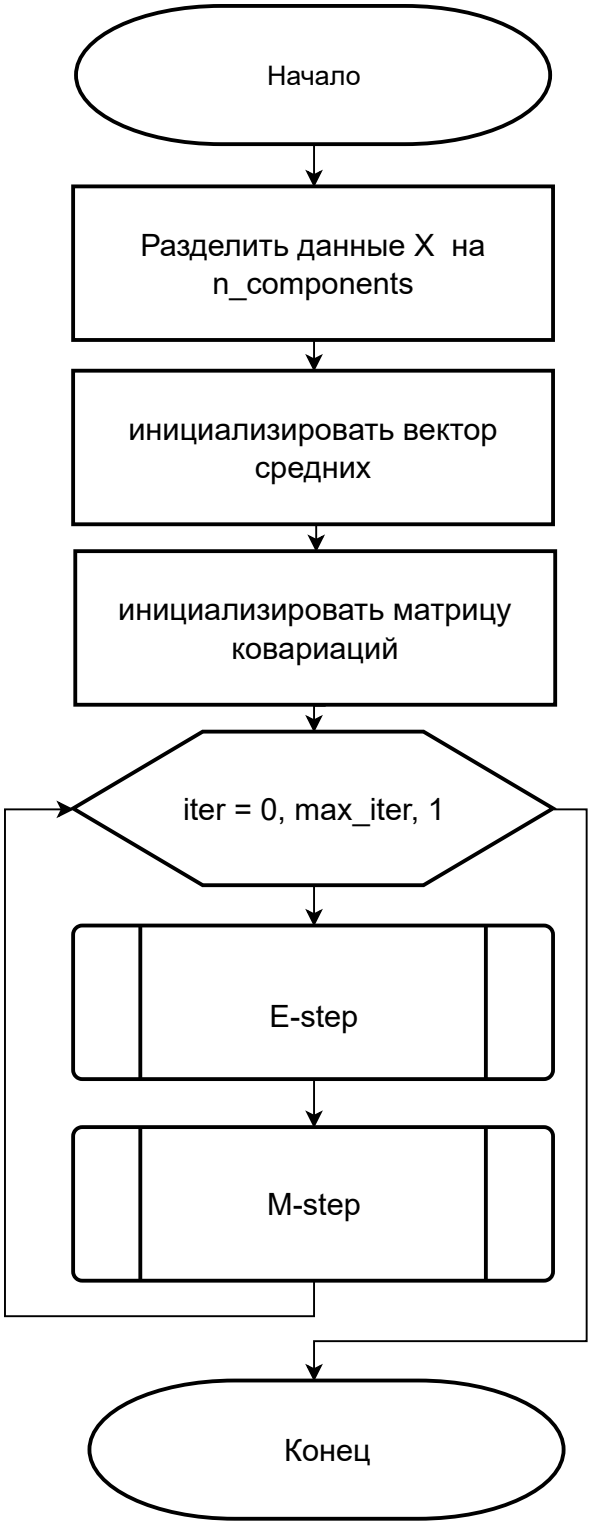

Рисунок 2.6 – Схема алгоритма метода максимизации ожидания для модели Гауссовой смеси

2.6 Рекомендация новостей на основе данных, выделенных в кластеры

Заключительным этапом разработки приложения, является создание рекомендательной системы новостей на основе данных, полученных после проведения нечеткой кластеризации, а именно предоставление пользователю набора новостей из того же кластера, что и выбранная им новость и кластеров, центры которых наиболее близки к текущему. Результатом данного этапа является рекомендательная система, предоставляющая пользователю рекомендации новостей на основе его персональных предпочтений.

2.6.1 Принцип работы рекомендательной системы

Принцип работы рекомендательной системы приведен на схеме алгоритма, изображенной на рисунке 2.7.

Рисунок 2.7 – Схема рекомендательной системы на основе контента

Для того, чтобы выполнить корректную рекомендацию требуется предварительно провести действия, описанные выше. Следовательно перед самой

рекомендацией новости требовалось предобработать для корректной работы алгоритма, а также провести понижение размерности для ускорения скорости работы, пожертвовав незначительным количество информации.

2.7 Тестирование обученной модели

Поскольку у нас нет априорных данных о принадлежности новостей к категориям, требуется выбрать такой критерий оценки, который позволит оценить насколько объект похож на свой кластер по сравнению с другими кластерами. Данную задачу наилучшим образом решает метод оценки качества под названием Силуэт (Silhouette). Метод силуэтов — способ изучения разделительного расстояния между результирующими кластерами наблюдений, данная мера имеет диапазон [-1, 1]. Коэффициенты силуэта около +1 указывают на то, что образец находится далеко от соседних кластеров. Значение, близкое к нулю указывает, что выборка находится на границе принятия решения между двумя соседними кластерами или очень близко к ней, а отрицательные значения указывают на то, что эти выборки могли быть назначены неправильному кластеру.

Оценка для всей кластерной структуры приведена в формуле 22.

$$Sil(C) = \frac{1}{N} \sum_{c_k \epsilon C} \sum_{x_i \epsilon c_k} \frac{b(x_i, c_k) - a(x_i, c_k)}{max\{a(x_i, c_k), b(x_i, c_k)\}}, \tag{22}$$

где $a(x_i, c_k) = \frac{1}{|c_k|} \sum_{x_j \epsilon c_k} ||x_i - x_j||$ — среднее расстояние от $x_i \epsilon c_k$ до других объектов из кластера c_k (компактность),

$b(x_i, c_k) = min_{c_l \epsilon C}\{\frac{1}{|c_l|} \sum_{x_j \epsilon c_l} ||x_i - x_j||\}$ — среднее расстояние от $x_i \epsilon c_k$ до объектов из другого кластера $c_l : k \neq l$ (отделимость).

По формуле 22 можно заметить, что

$-1 \leqslant Sil(C) \leqslant 1.$

И чем ближе данная оценка к 1, тем лучше.

Более того для тестирования реализованной модели нечеткой кластеризации и ее сравнения с моделью из библиотеки scikit-learn, потребуется оценка способная дать более точный результат называемая V-мерой, так как она ис-

пользует информацию о том к каким кластерам принадлежат данные.

V-мера представляет из себя гармоническое среднее оценки однородности и полноты. Вычисление V-меры представлено в формуле 23.

$$V - measure = 2 * \frac{h * c}{h + c}, \tag{23}$$

где h — это однородность, представленная в формуле 25, а c — полнота и представлена она в формуле 26.

Однородность измеряет, насколько образцы в кластере похожи и измеряется с помощью энтропии Шеннона. Вычисление однородности приведено в формуле 25, энтропия Шеннона для образцов с назначенным кластером C в кластере K приведена в формуле 24.

$$H(C|K) = -\sum \frac{n_{ck}}{N} \log(\frac{n_{ck}}{n_k}), \tag{24}$$

где n_{ck} — это количество образцов из кластера c в кластере k, n_k это общее количество образцов в кластере c, а N размер набора данных.

$$h = 1 - \frac{H(C|K)}{H(C)} \tag{25}$$

Как можно заметить, если все образцы в кластере k имеют одинаковый назначенный кластер c, то однородность равна 1.

Полнота же измеряет, сколько похожих образцов объединяется алгоритмом кластеризации. Формула вычисления полноты приведена в формуле 26.

$$c = 1 - \frac{H(K|C)}{H(K)}, \tag{26}$$

где $H(K|C)$ отражает энтропию отношения образцов из кластера c в кластере k к общему количеству образцов c.

Если все образцы с назначенным кластером c назначены одному кластеру k, то полнота равна 1.

2.8 Выводы из конструкторского раздела

В данном разделе была представлена IDEF0 диаграмма разрабатываемой рекомендательной системы, далее были рассмотрены все этапы разработки, а именно:

- этап предобработки входных данных;
- этап векторизации предобработанных данных;
- этап понижения размерности матрицы признаков;
- этап разделения новостей на кластеры;
- этап рекомендации новостей.

Также была приведена схема алгоритма метода максимизации ожидания для модели Гауссовой смеси, а также схема алгоритма как шага ожидания, так и шага максимизации данного алгоритма. Приведена схема рекомендательной системы и описан метод тестирования обученной модели с использованием разных методов оценки.

3 Технологический раздел

3.1 Выбор средств разработки

В данном разделе рассматриваются такие инструменты разработки, как язык программирования, среда разработки и используемые библиотеки.

В первом подразделе будет описан язык программирования, который был выбран для реализации программного продукта, а также были описаны необходимые библиотеки для реализации поставленной задачи. Как для языка, так и для используемых библиотек были описаны преимущества их использования и обоснованность выбора именно данного языка и библиотек.

3.1.1 Язык программирования и используемые библиотеки

В качестве языка программирования для разработки программного продукта было принято решение использовать язык Python v3.9.2. Преимуществом данного языка является большое разнообразие представленных библиотек и фреймворков, а также обладает большой гибкостью и удобством использования.

При разработке программного обеспечения использовались следующие библиотеки и фреймворки:

- NumPy — библиотека, поддерживающая работу с большими многомерными массивами и матрицами, а также предоставляющая набор математических функций для работы с этими данными [10];
- Pandas — библиотека, предоставляющая удобные структуры и операции для работы с табличными даннами [11];
- Scikit-learn — данная библиотека включает в себя различные алгоритмы машинного обучения и подготовку данных для последующей; классификации, поддерживает взаимодействие с NumPy и Pandas [12];
- SciPy — библиотека для математического и числового анализа, такого как вычисление косинусной близости [13];
- NLTK — библиотека для работы с текстовыми данными, предоставляю-

щая возможности обработки и лемматизации текста [14];

- Django — бесплатный высокоуровневый веб-фреймворк [15].

3.1.2 Среда разработки

В качестве среды разработки модуля рекомендательной системой было решено использовать IDE Jupyter Notebook так как он позволяет выполнять код по ячейкам, что очень удобно для разработки модулей, использующих методы машинного обучения. Для разработки графического интерфейса рекомендательной системы была использована IDE PyCharm, которая предоставляется студентам бесплатно по лицензии, а также предоставляет возможности создания виртуальной среды разработки для быстрой установки новых библиотек и модулей.

3.2 Структура разработанного ПО

Структура разработанного ПО в виде UML диаграммы представлена на рисунке 3.8.

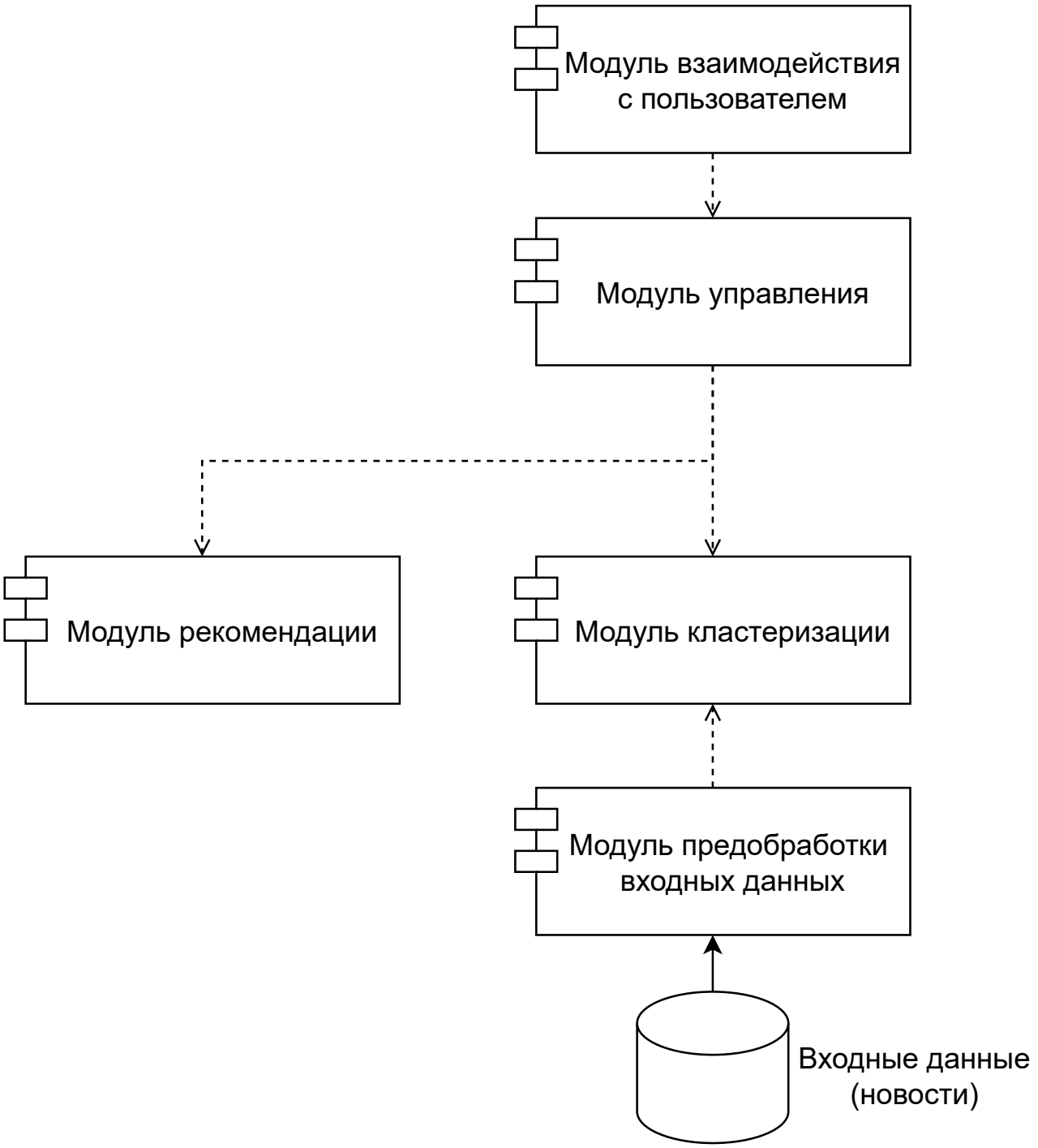

Рисунок 3.8 – UML диаграмма разработанного ПО

Каждый из модулей, изображенный на диаграмме, содержит сгруппированные по функциональному значению соответствующие классы. Модуль взаимодействия с пользователем отвечает за пользовательский интерфейс. Модуль управления объединяет модуль рекомендации и модуль кластеризации и координирует их работу.

3.3 Пользовательский интерфейс

Интерфейс представляет из себя две веб-страницы с новостями. Веб-страница реализована на HTML с использованием CSS стилей. В качестве веб-фреймворка использовался Django.

Пример работы программы в случае если пользователь находится на главной странице и в случае если он находится на странице с новостью, представлены на рисунках 3.9, 3.10.

Рисунок 3.9 – Пример работы программы главной страницы новостного сайта

Рисунок 3.10 – Пример работы программы страницы с новостью

Далее будет приведен пример работы программы в случае если пользователь добавляет новость, к примеру про бои UFC и страница с рекомендациями на основе, добавленной новости. На рисунке 3.11 страница добавления новости, а на рисунке 3.12 страница добавленной новости и рекомендаций на ее основе.

Добавление новости

Абстрактное описание:

Strongman champ flatlines opponent in latest MMA b

Заголовок:

Mariusz Pudzianowski scored a crushing first-round k

Новость:

Poland's Mariusz Pudzianowski,
considered by many to be among the
finest strongmen competitors in history,
stunned his former world champion
opponent Michal Materla with a crushing
first-round knockout at KSW 70 in Lodz,
Poland on Saturday night.

Pudzianowski, the 45-year-old multiple
time winner of the title of the World's
Strongest Man, transitioned to mixed

Добавить

Рисунок 3.11 – Пример работы программы страницы добавления новости

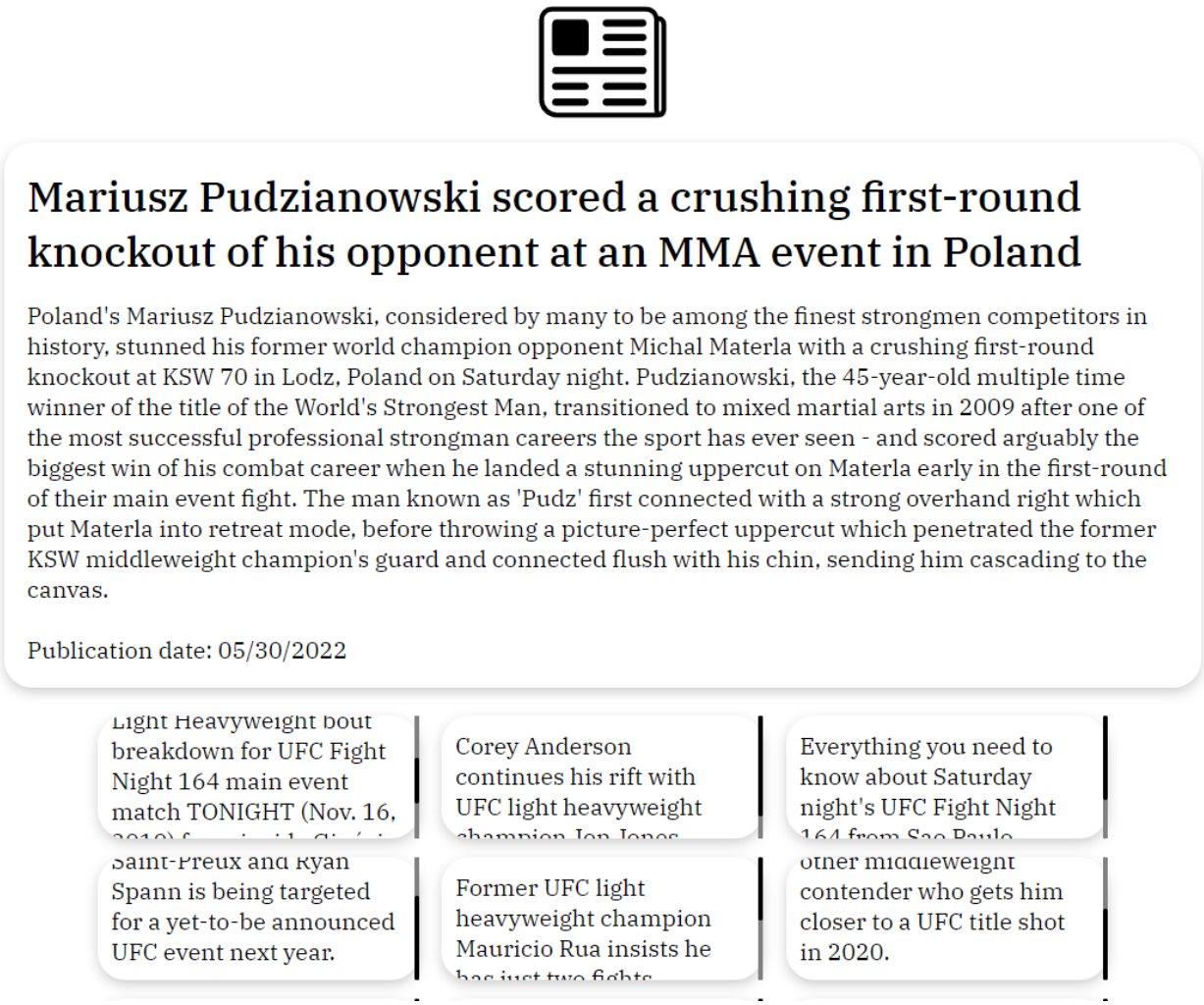

Рисунок 3.12 – Страница с добавленной новостью и рекомендациями к ней

Как можно заметить, все рекомендации так или иначе связаны с боями UFC, что говорит о хорошем качестве предсказания новостей к рекомендации.

3.4 Выводы из технологического раздела

В данном разделе были описаны средства реализации, которые были использованы для разработки ПО, после чего были приведены выбранные инструменты, обоснованы причины их использования и преимущества.

Была описана структура разработанного программного обеспечения и приведено описание каждого из модулей, приведенного на UML диаграмме. Также был приведен пользовательский интерфейс программы и продемонстрирован пример работы программы в случае просмотра и добавления новости.

4 Исследовательский раздел

4.1 Выборка данных

Датасет собран из новостей на английском языке с сайта MSN.com. [9]

На вход векторизатору подается выборка данных размером 98247 документов, каждая из которых представляет собой предобработанную строку на английском языке, а также дату публикации.

Данные хранятся в файле имеющем формат tsv — это формат для представления таблиц баз данных. Для последующего обучения модели нечеткой кластеризации использовалась полная выборка данных, так как каждой новости требуется сопоставить кластер для корректной работы алгоритма.

4.2 Сравнение методов

В данном разделе будет произведено сравнение собственной реализации модели Гауссовой смеси и модели из библиотеки scikit-learn. Будет создан искусственный набор данных имеющий 3 кластера. После выполнения кластеризации собственным методом и методом из библиотеки scikit-learn будет произведена оценка кластеризации V-мерой, которая представляет из себя гармоническое среднее оценки однородности и полноты.

На рисунке 4.13 приведен исходный размеченный набор данных размером 400, а на рисунках 4.14 и 4.15 приведен набор данных кластеризованный собственным методом Гауссовой смеси и методом из библиотеки scikit-learn.

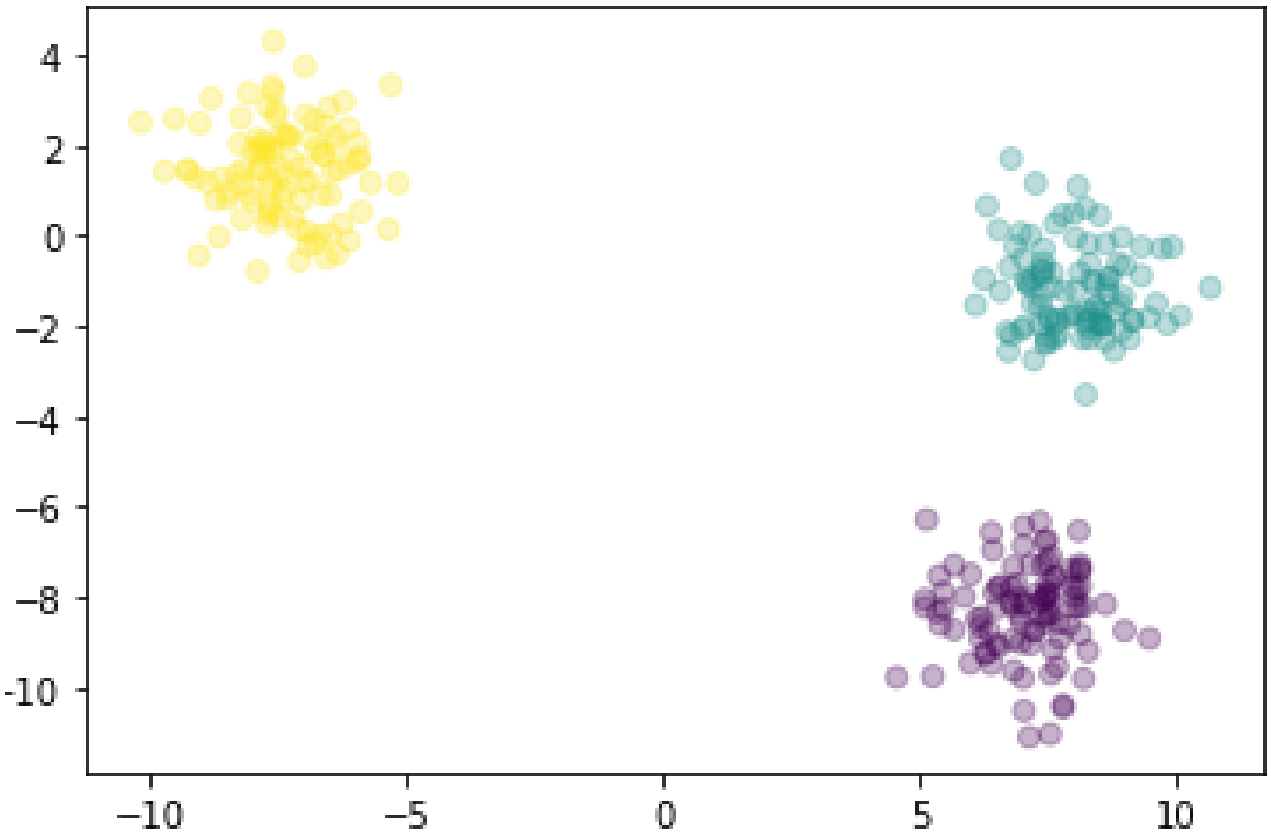

Рисунок 4.13 – График исходного набора данных. Цветами выделены кластеры

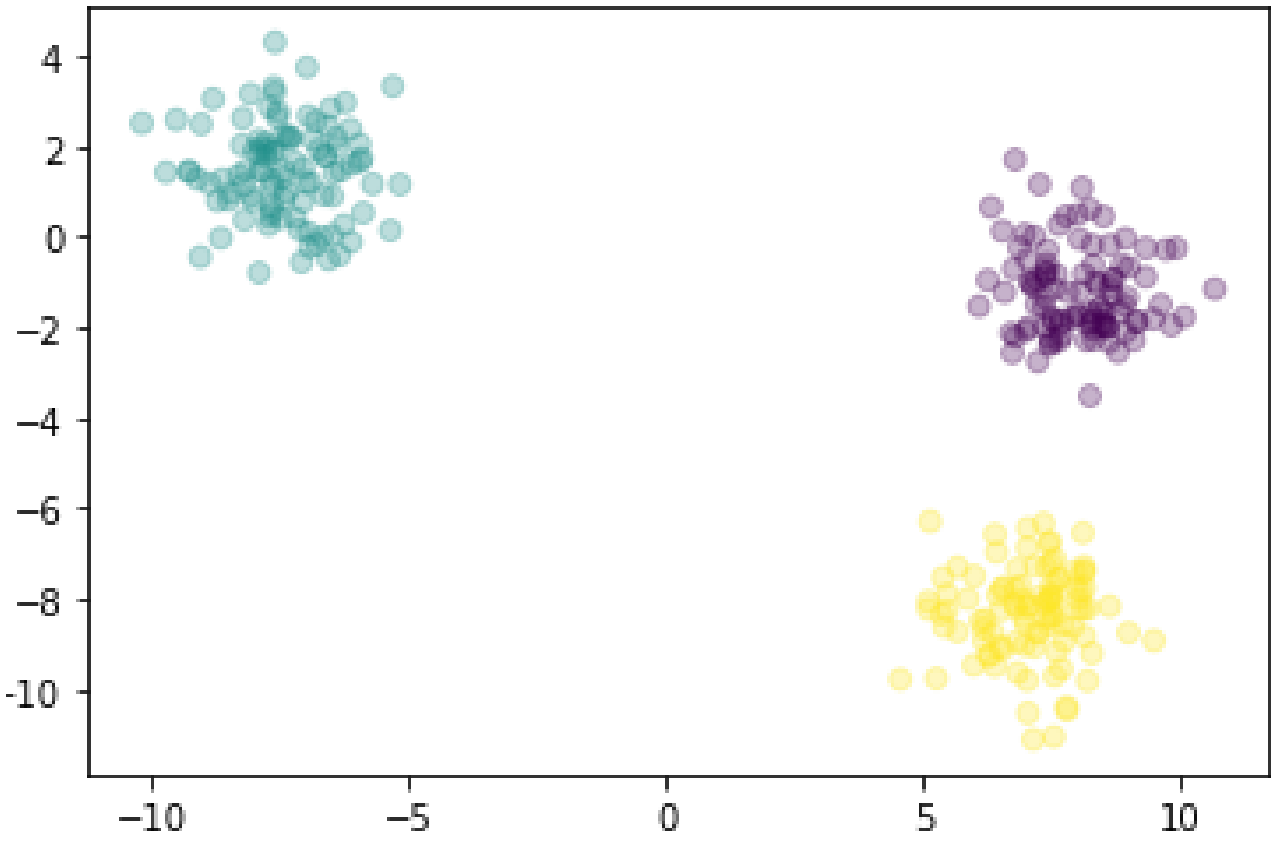

Рисунок 4.14 – График набора данных, кластеризованный собственным методом Гауссовой смеси

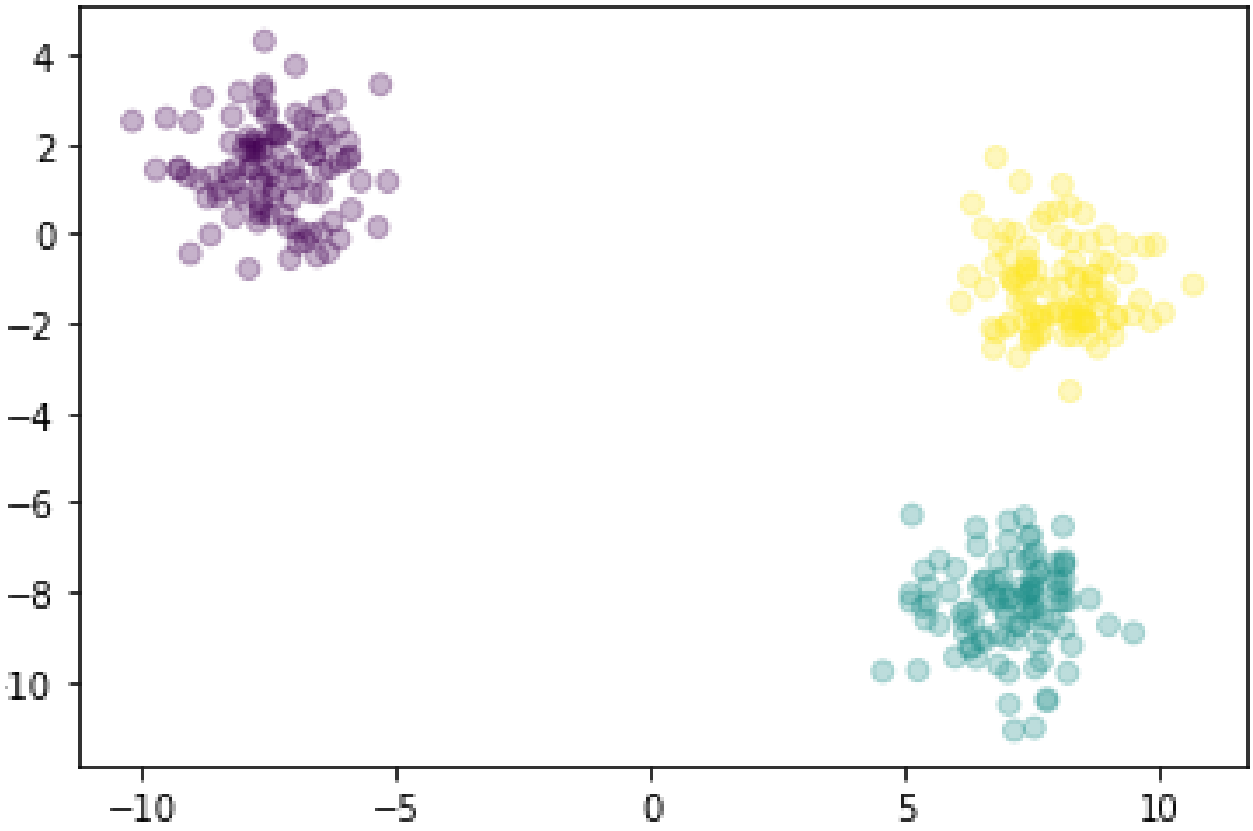

Рисунок 4.15 – График набора данных, кластеризованный методом Gaussian Mixture Model из библиотеки scikit-learn

Как можно заметить алгоритм собственной реализации корректно разделяет данные. Заключительным этапом является оценка кластеризации V-мерой, результат оценки приведен в таблице 4.

Таблица 4 – Сравнение реализованного метода Гауссовой смеси с методом из библиотеки scikit-learn

Количество образцов	Значение V-меры реализованного метода	Значение V-меры метода из scikit-learn
100	0.930	0.965
250	0.930	0.930
300	0.940	0.965

Как можно заметить по таблице, метод собственной реализации лишь незначительно уступает в точности методу из библиотеки scikit-learn, а при значении 2500 совпадает.

4.3 Порядок параметризации

Для достижения наилучшего качества нечеткой кластеризации и последующих рекомендаций в первую очередь следует определить оптимальные параметры векторизатора TF-IDF, после чего требуется определить оптимальное количество компонентов, которые следует оставить для того, чтобы доля объясненной дисперсии после понижения размерности была не меньше 0.95. Заключительным этапом является подбор оптимальных параметров модели Гауссовой смеси для достижения наилучшего качества.

4.3.1 Параметризация векторизатора TF-IDF

Для получения наилучших параметров векторизатора будем варьировать максимальную величину параметра DF (документная частота, то есть максимальное допустимое число документов, в которых встретился термин t), термины выше порогового значения не будут использоваться при векторизации, также будет варьироваться диапазон N-грамм, то есть будут ли признаки формироваться из отдельных слов или из нескольких.

Качество векторизатора будем проверять с помощью оценки последующей нечеткой кластеризации мерой Силуэт. Параметры метода понижения размерности и нечеткой кластеризации заданы по умолчанию. Результат исследования приведен на таблице 5.

Таблица 5 – Значение критерия Силуэт от параметров векторизатора TF-IDF

используемые n-граммы / Max_df	Слова по отдельности	Слова по отдельности и пары слов
0.25	0.912	0.923
0.30	0.905	0.919
0.35	0.903	0.918

Посмотрев на таблицу выше можно сделать вывод о том, что наилучшими значениями параметров векторизатора являются 0.25 для параметра max_df и использование как всех слов по отдельности так и пар слов.

4.3.2 Параметризация метода понижения размерности SVD

Так как исходная выборка обладает большой размерностью, а также является разреженной требуется применить метод понижения размерности в целях экономии памяти и увеличения скорости работы. Целевым показателем является доля объясненной дисперсии, ее значение должно равняться не менее 0.95.

На рисунке 4.16 приведена зависимость количества компонентов после понижения размерности и долю объясненной дисперсии для данного количества компонент.

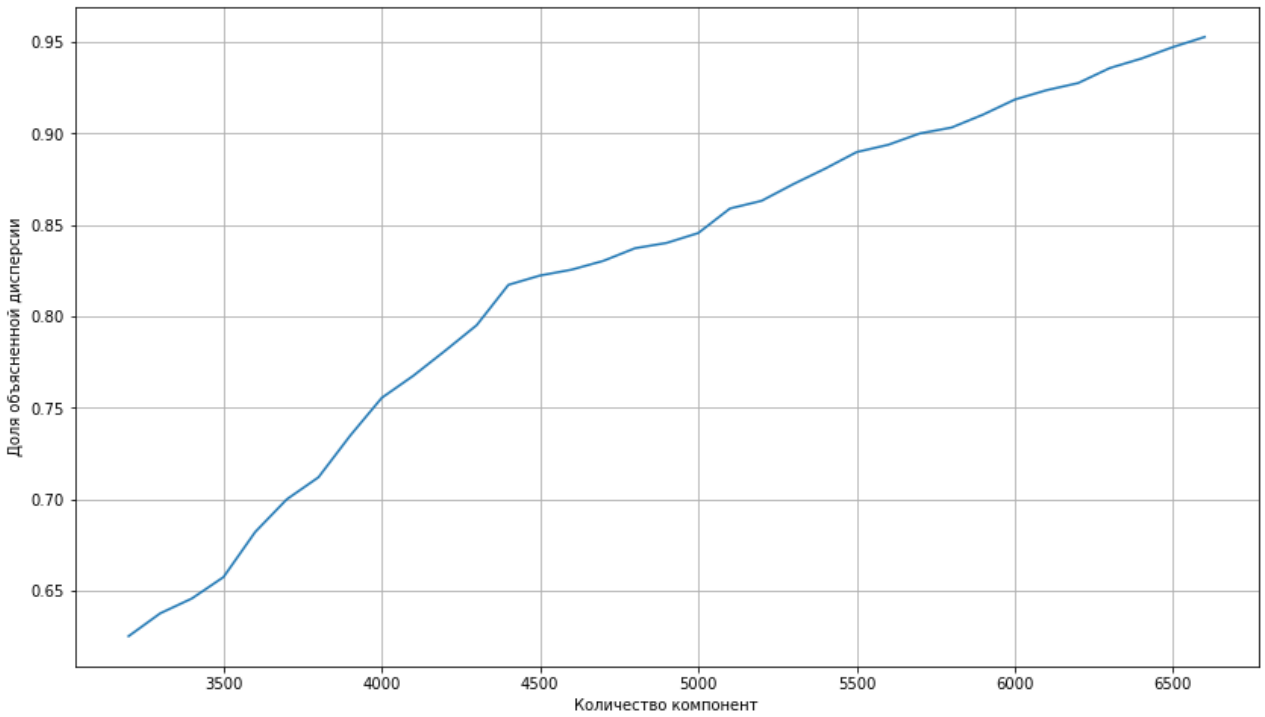

Рисунок 4.16 – Доля объясненной дисперсии в зависимости от количества компонент

Как можно заметить на картинке вышее необходимое количество компонент составляет 6700, при данном количество доля объясненной дисперсии составляет чуть выше 0.95. Данный этап позволил сохранить информативность данных и сэкономил большой количество ресурсов памяти.

4.3.3 Параметризация метода Гауссовой смеси

Для достижения наилучшего результата нечеткой кластеризации требуется варьировать ключевой параметр, которым является количество кластеров, требуется получить такое количество кластеров при котором оценка методом Силуэта будет давать наивысший результат (быть как можно более близким к 1). Для достижения необходимого результата был проведен эксперимент при котором варьировалась количество кластеров. Зависимость полученной оценки от количества кластеров можно увидеть на рисунке 4.17.

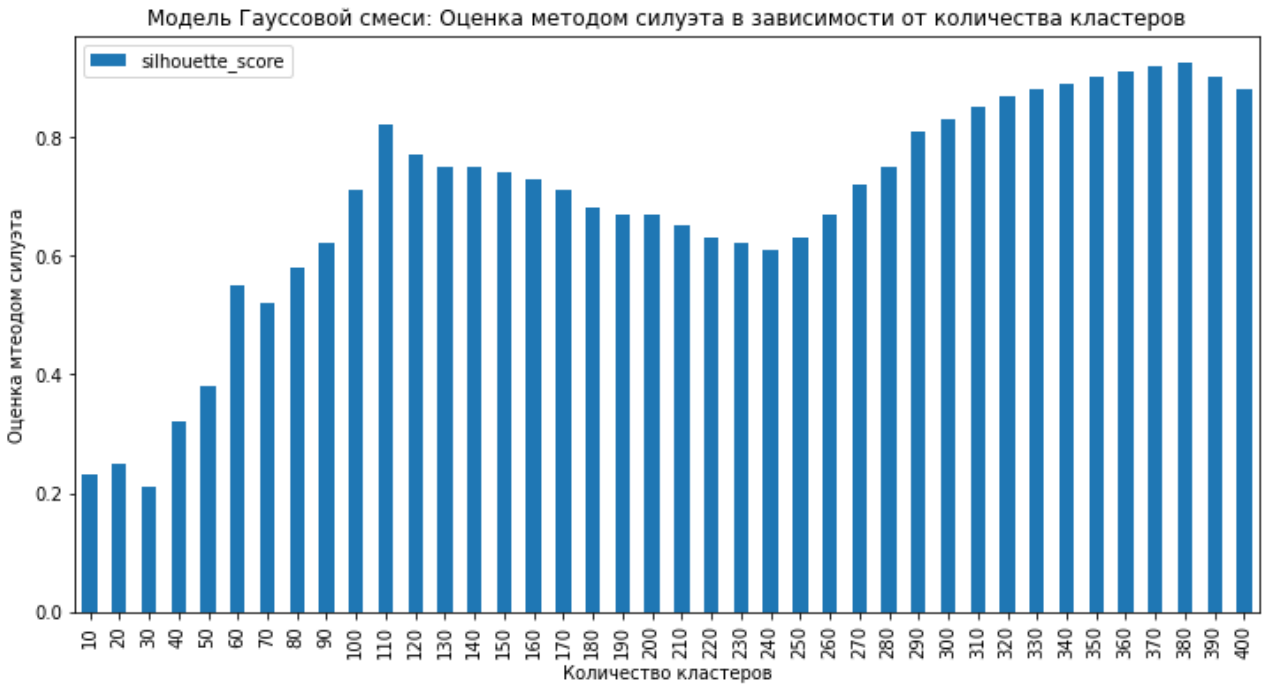

Рисунок 4.17 – Полученная оценка методом Силуэта в зависимости от количества кластеров

Видно, что явные пики на графике это 110 кластеров и 380 кластеров. При количестве кластеров равным 380 оценка достигает порядка 0.92 по методу силуэта, что является очень хорошим показателем, именно данное количество кластеров и было использовано при построении финальной версии рекомендательной системы.

4.4 Рекомендации к применению рекомендательной системы

После проведения исследования и параметризации, полученной рекомендательной системы можно сделать вывод о том, что она работает эффективно и исправно. Данную систему можно применять при создании собственных новостных сайтов, но стоит учесть, что первоначальное обучение модели на уже имеющихся данных должно выполняться на мощной машине и кластеры новостям должны быть определены заранее, поскольку векторизатор, метод понижения размерности и сама нечеткая кластеризация методом Гауссовой смеси являются трудоемкими операциями в случае большого объема данных.

4.5 Выводы из исследовательского раздела

В данном разделе была проведена парметризация алгоритмов из ключевых этапов построения рекомендательной системы, а также проведена проверка качества работы разработанного метода в сравнении с методом из библиотеки scikit-learn с помощью V-меры, в ходе которой было выявлено, что разработанный метод в большинстве случаев не уступает методу из библиотеки. Параметризовав алгоритм векторизации был сделан вывод о том, что наилучшим значением для параметра max_df векторизатора Tf-IDF является 0.25, а в качестве n-грамм следует брать как слова по отдельности так и пары слов. Для метода понижения размерности наилучшим количеством компонент является 6700, поскольку доля объясненной дисперсии при данном количестве превышает 0.95, что очень хорошо описывает данные, экономя при этом память. Заключительным этапом было выявление оптимального количества кластеров в модели Гауссовой смеси, в качестве результата была получена зависимость оценки метода Силуэта от количества кластеров, исходя из которой можно сделать вывод о том, что наилучшее количество кластеров для построения рекомендательной системы на исходных данных равно 380. В конце данного раздела была описана применимость разработанной рекомендательной системы.

ЗАКЛЮЧЕНИЕ

В рамках данной выпускной квалификационной работы была разработана и реализована рекомендательная система новостей на основе нечеткой кластеризации.

В результат проделанной работы были выполнены следующие задачи и достигнуты следующие результаты:

- рассмотрены существующие подходы в рекомендательных системах;
- проведен анализ существующих методов нечеткой кластеризации;
- разработана рекомендательная система на основе нечеткой кластеризации методом Гауссовой смеси;
- сконструировано и разработан программное обеспечение, демонстрирующее работы данного метода;
- проведена параметризация ключевых алгоритмов, используемых при разработке программного обеспечения;
- разработанная система исследована на предмет применимости.

В качестве направлений дальнейшей работы над методом можно выделить следующие:

- использовать другие, более ресурсоемкие, но потенциально более эффективные способы векторизации, такие как Doc2Vec;
- адаптировать алгоритм рекомендательной системы под новости, написанные на русском языке.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Clustering Algorithms III: Schemes Based on Function Optimization / Sergios Theodoridis, Konstantinos Koutroumbas, in Pattern Recognition (Fourth Edition), 2009.

2. News recommender system: a review of recent progress, challenges, and opportunities [Электронный ресурс]. — Режим доступа https://link.springer.com/article/10.1007/s10462-021-10043-x#auth-Chen-Ding, свободный — (22.12.2021).

3. Нейский И. М. Классификация и сравнение методов кластеризации. [Электронный ресурс]. — Режим доступа http://it-claim.ru/Persons/Neyskiy/Article2_Neiskiy.pdf, свободный — (22.12.2021).

4. Б. Мингликулов. Алгоритм решения задачи нечеткой кластеризации // Статья: центр разработки программных продуктов и аппаратно–программных комплексов при Ташкентском университете информационных технологий, Стр 1–5.

5. Д. С. Кутуков. Применение методов кластеризации для обработки новостного потока / Д. С. Кутуков. — Текст : непосредственный // Технические науки: проблемы и перспективы : материалы I Междунар. науч. конф. (г. Санкт-Петербург, март 2011 г.). — Санкт-Петербург: Реноме, 2011. — С. 77-83. — URL: https://moluch.ru/conf/tech/archive/2/207/ (дата обращения: 22.12.2021).

6. Я. С. Погорельская. Обзор подходов к построению рекомендательных систем // Статья: кафедра информационных технологий, Российский университет дружбы народов, Стр 1–5.

7. David J. Miller, Carl A. Nelson, Molly Boeka Cannon, Kenneth P. Cannon, ”Comparison of Fuzzy Clustering Methods and Their Applications to Geophysics

Data Applied Computational Intelligence and Soft Computing, vol. 2009, Article ID 876361, 16 pages, 2009.

8. M.-S. Yang, A survey of fuzzy clustering, Mathematical and Computer Modelling, Volume 18, Issue 11, 1993, Pages 1-16, ISSN 0895-7177, https://doi.org/10.1016/0895-7177(93)90202-A.

9. MSNnews [Электронный ресурс]. – Режим доступа: https://msnews.github.io – Дата обращения: 03.04.2022

10. NumPy [Электронный ресурс]. – Режим доступа: https://numpy.org – Дата обращения: 05.04.2022

11. Pandas [Электронный ресурс]. – Режим доступа: https://pandas.pydata.org – Дата обращения: 05.04.2022

12. Scikit-learn [Электронный ресурс]. – Режим доступа: https://scikit-learn.org – Дата обращения: 05.04.2022

13. SciPy [Электронный ресурс]. – Режим доступа: https://www.scipy.org – Дата обращения: 05.04.2022

14. NLTK [Электронный ресурс]. – Режим доступа: https://www.nltk.org – Дата обращения: 05.04.2022

15. Django [Электронный ресурс]. – Режим доступа: https://www.djangoproject.com – Дата обращения: 05.04.2022

ПРИЛОЖЕНИЕ А

Листинг 1: Предобработка текста.

```
# Оставляем только кириллические символы
regex = re.compile(u"[A-Za-z]+")

def words_only(text, regex=regex):
return " ".join(regex.findall(str(text)))

df.body = df.body.str.lower()
df.loc[:, 'body'] = df.body.apply(words_only)

# Удаляем стопслова-
mystopwords = stopwords.words('english') + ['-', '-']

def  remove_stopwords(text, mystopwords = mystopwords):
try:
return u" ".join([token for token in text.split() if not token in mystopwords
    ])
except:
return u""

df.body = df.body.apply(remove_stopwords)

# нормализуем текст
wordnet_lemmatizer = WordNetLemmatizer()
def lemmatize(text, lemmatizer=wordnet_lemmatizer):
word_list = nltk.word_tokenize(text)
lemmatized_output = ' '.join([lemmatizer.lemmatize(w) for w in word_list])
return lemmatized_output

df.body = df.body.apply(lemmatize)
```

Листинг 2.1: Реализация модели Гауссовой смеси.

```
class GMM:
    def __init__(self, n_components, max_iter = 100, comp_names=None):
        # инициализируем начальные параметры
        # n_components - количество кластеров
        # max_iter - максимальное количество итераций алгоритма
        self.n_componets = n_components
        self.max_iter = max_iter
        if comp_names == None:
            self.comp_names = [f"comp{index}" for index in range(self.
    n_componets)]
        else:
            self.comp_names = comp_names
        # массив с отношениями колиества элементов в каждом кластере ко всем
    элементам данных
        self.pi = [1/self.n_componets for comp in range(self.n_componets)]

    def predict(self, X):
        # выбор кластера к которому образец имеет наибольшую вероятность
    попадания
        probas = []
        for n in range(len(X)):
            probas.append([self.multivariate_normal(X[n], self.mean_vector[k
    ], self.covariance_matrixes[k])
            for k in range(self.n_componets)])
        cluster = []
        for proba in probas:
            cluster.append(self.comp_names[proba.index(max(proba))])
        return cluster

    def multivariate_normal(self, X, mean_vector, covariance_matrix):
        # вычисление вектора определяющего многомерное нормальное
    распределение
        return (2*np.pi)**(-len(X)/2)*np.linalg.det(covariance_matrix)
    **(-1/2)*np.exp(-np.dot(np.dot((X-mean_vector).T, np.linalg.inv(
    covariance_matrix)), (X-mean_vector))/2)
```

Листинг 2.2: Реализация модели Гауссовой смеси.

```
    def fit(self, X):
        # разделение входных данных на количество кластеров
        new_X = np.array_split(X, self.n_componets)
        # получение вектора средних для каждого столбца каждого кластера
        self.mean_vector = [np.mean(x, axis=0) for x in new_X]
        # получение матрицы ковариаций для каждого кластера
        self.covariance_matrixes = [np.cov(x.T) for x in new_X]
        del new_X
        # начало итеративного алгоритма максимизации ожидания
        for iteration in range(self.max_iter):
        ''' ----------------------   E - STEP   ---------------------- '''
        '''
        вычисление матрицы в которой строки - это элемент данных,
        а столбцец это кластер, следовательно каждый элемент матрицы
        это вероятность принадлежности элемента к столбцу т. е. кластеру
        '''
            self.r = np.zeros((len(X), self.n_componets))
            for n in range(len(X)):
                for k in range(self.n_componets):
                # вычисление вероятности принадлежности элемента к кластеру
                    self.r[n][k] = self.pi[k] * self.multivariate_normal(X[n
], self.mean_vector[k], self.covariance_matrixes[k]) / sum([self.pi[j]*
self.multivariate_normal(X[n], self.mean_vector[j], self.
covariance_matrixes[j]) for j in range(self.n_componets)])
            # аычисление списка сумм столбцов матрицы
            N = np.sum(self.r, axis=0)
```

Листинг 2.3: Реализация модели Гауссовой смеси.

```
        ''' ----------------------   M - STEP   ---------------------- '''
            # инициализация вектора средних
            self.mean_vector = np.zeros((self.n_componets, len(X[0])))
            # обновление векторов средних
            for k in range(self.n_componets):
                for n in range(len(X)):
                    self.mean_vector[k] += self.r[n][k] * X[n]
            self.mean_vector = [1/N[k]*self.mean_vector[k] for k in range(
    self.n_componets)]
            # инициализация матрицы ковариации
            self.covariance_matrixes = [np.zeros((len(X[0]), len(X[0]))) for
    k in range(self.n_componets)]
            # обновление матриц ковариаций
            for k in range(self.n_componets):
                self.covariance_matrixes[k] = np.cov(X.T, aweights=(self.r[:,
     k]), ddof=0)
            self.covariance_matrixes = [1/N[k]*self.covariance_matrixes[k]
    for k in range(self.n_componets)]
            # обнолвение списка долей кластера
            self.pi = [N[k]/len(X) for k in range(self.n_componets)]
```