После каркас был отредактирован, был подобран цвет и расставлены источники освещения. Таким образом, была получена модель здания корпуса №5.

В.С. Белошедов (ГГУ имени Ф. Скорины, Гомель)
Науч. рук. **Е.А. Дей,** канд. физ.-мат. наук, доцент

ИСПОЛЬЗОВАНИЕ КОЛЛАБОРАТИВНОЙ ФИЛЬТРАЦИИ ДЛЯ ВОССТАНОВЛЕНИЯ РЕЗУЛЬТАТОВ ОПРОСОВ НА ЯЗЫКЕ PYTHON

Современные web-приложения становятся сложней с каждым днем. В качестве примеров можно указать web-сервисы Netflix/Megogo (фильмы/сериалы), Spotify/Boom (музыка), Amazon/AliExpress (товары). При этом вместо выдачи всего доступного контента сервисам выгоднее рекомендовать пользователю контент, который соответствует его предпочтениям или сложившейся практике (в частности, к материнской плате будут рекомендоваться часто выбираемые процессоры/видеокарты). Один из методов реализации таких рекомендаций является метод коллаборативной фильтрации.

Коллаборативная фильтрация – это процесс прогнозирования, основанный на анализе существующей информации об объектах (пользователи, группы людей) и их оценках контента (песни, фильмы, компьютеры и так далее). В расчет может браться еще и другая информация о пользователе: возраст, страна проживания, пол.

Стоит отметить, что оценки, возраст можно считать количественным признаками, страну проживания и пол - номинальным признаками. Известно, что сервис Netflix, зная пол пользователей, будет подбирать разные обложки фильмов: для женщин это могут быть более романтические, для мужчин более брутальные.

Предлагаемая работа посвящена реализации коллаборативной фильтрации. Наиболее подходящим в данном случае является язык программирования Python. Python − это высокоуровневый язык программирования универсального назначения. В настоящее время язык Python широко используется не только в промышленных задачах (WEB приложения, DevOps, Machine Learning), но и в научных расчетах как бесплатная альтернатива коммерческим приложениям (Matlab, Wolfram) при программировании сложных вычислений и отображения их результатов. Существует огромное количество реализаций

этого языка: CPython, Pypy, IronPython (.Net платформа), Jython (JVM платформа) и другие. Для реализации задачи был использован CPython.

Принцип работы программы следующий: программа считывает данные о пользователях и их оценках из заранее подготовленного файла (базы данных), строит матрицу, где строки — это пользователи, а столбцы – это признаки – оценки для товара, вопроса и т.п. Выбирается пользователь и признак, который требуется вычислить. После этого следует узнать меру близости нескольких пользователей. Для этого можно представить оценки пользователей как векторы и находить косинус угла между двумя векторами. Полученное значение будет находится в пределах [0;1]. Отсортировав значения по убыванию меры значимости и выбрав первых К пользователей, которые имеют оценку данного признака, мы вычислим предполагаемую оценку. Вычислив оценки для всех неизвестных признаков, можно выбрать максимальную оценку и рекомендовать данный признак пользователю. Важно отметить, что чем больше данных присутствует, тем более точный ответ будет вычислен.

Таким образом, такую задачу можно классифицировать как восстановление результатов опроса.

В качестве примера приводится реализация класса Similarity на языке Cpython (рисунок 1). Внутри класса находится статический метод cosine, который принимает два параметра: a и b. Это представление искомых векторов – два списка. В качестве результата метод возвращает косинус угла между векторами.

```
from math import sqrt
class Similarity:
    @staticmethod
    def cosine(a, b):
        numerator = denominator = 0
        sum_a = sum_b = 0
        for i in range(a.__len__()):
            sum_a += a[i] * a[i]
            sum_b += b[i] * b[i]
            numerator += a[i] * b[i]
        denominator = sqrt(sum_a) * sqrt(sum_b)
        return numerator / denominator
```

Рисунок 1 – Реализация класса Similarity

В интернете имеются открытые базы данных реальных проектов. В работе была выполнена тестовая реализация коллаборативной

фильтрации на основе одной из таких баз данных. База представляет из себя объекты и признаки. Часть признаков отсутствует, их следует рассчитать.

Использованная база состояла из 1000 объектов, 1700 признаков и 100000 оценок. Тестирование проводилось для 20000 случаев.

При количестве пользователей К = 5 результат совпадения прогнозирования с реальными данными оказался равен 36%. В 65% случаев результат совпадения прогнозирования данных с реальными данными лежал в промежутке [-1;1] от реального значения.

Изменяя количестве пользователей К и увеличивая количество входных данных, мы можем увеличить точность вычислений.

Литература

1. Маккинли, У. Python и анализ данных / У. Маккинли. – пер. с англ. − М.: ДМК Пресс, 2015. − 482 с.

2. Бизли, Д.М. Python. Подробный справочник / Д.М. Бизли. – 4-е издание. – пер. с англ. − СПб.: Символ-Плюс, 2010. − 864 с.

3. Лутц, М. Изучаем Python / М. Лутц. − 4-е издание. − пер. с англ. − СПб.: Символ-Плюс, 2010. − 1280 с.

Н.В. Берещенко (ГГУ имени Ф. Скорины, Гомель)
Науч. рук. **В.Д. Левчук,** канд. техн. наук, доцент

ИНСТРУМЕНТЫ РЕАЛИЗАЦИИ МОБИЛЬНОГО КЛИЕНТА ДЛЯ ИНФОРМАЦИОННОГО ОБСЛУЖИВАНИЯ ИНОСТРАННЫХ СТУДЕНТОВ УО «ГГУ ИМЕНИ Ф. СКОРИНЫ»

Мобильное приложение «Scorina University» выполняет такие функции, как:

– получение базы данных преподавателей;
– отображение корпусов на карте;
– поиск по карте;
– получение новостей;
– поддержка языков: Английский, Русский;
– возможность сообщить о проблемах в корпусе университета.

Разработка состояла из пяти этапов: проектирование приложения, разработка основной логики приложения, создание интерфейса приложения, создание Unit-тестов, доработка дополнительных функций.