Вишинська Т.

Науковий керівник – доц. Габрусєв В.Ю.

ВИРОБЛЕННЯ РЕКОМЕНДАЦІЙ НА ОСНОВІ ВЕЛИКИХ МАСИВІВ ДАНИХ З ВИКОРИСТАННЯМ АЛГОРИТМІВ КОЛЛАБОРАТИВНОЇ ФІЛЬТРАЦІЇ

Актуальність. В час масового поширення інформації з різних джерел створює певні труднощі в оцінці ситуації та прийнятті відповідних рішень. Тому на допомогу приходять технології прийняття рішень для вироблення персоналізованих рекомендацій, які грунтуються на коллаборативній фільтрація та відборі подібних зразків.

Коллаборативна фільтрація – це один з методів побудови прогнозів (рекомендацій) в рекомендаційних системах, що використовує відомі вподобання (оцінки) групи користувачів для прогнозування невідомих вподобань іншого користувача. Його основне припущення полягає в наступному: ті, хто однаково оцінювали які небудь предмети в минулому, схильні давати схожі оцінки іншим предметам і в майбутньому. Тим самим коллаборативна фільтрація відрізняється від більш простого підходу, що дає усереднену оцінку для кожного об'єкта інтересу, наприклад, що базується на кількості поданих за нього голосів. Дослідження в даній області активно ведуться і в наш час, що також обумовлюється і наявністю невирішених проблем у цій галузі.

У даній статті розглядаються окремі методи коллаборативної фільтрації на основі схожості груп, а також їх реалізація на мові програмування Python.

Зазвичай алгоритм коллаборативної фільтрації працює таким чином: переглядає велику групу результатів аналізів і відшукує в ній меншу групу з такими ж результатами. Він дивиться, які ще аналізи є у групі, об'єднує спільне і створює ранжований список пропозицій. Є кілька способів вирішити, які групи схожі, і об'єднати їх спільні аналізи в список. У цій статті розглянуто деякі з цих способів.

ЗБІР ІНФОРМАЦІЇ ПРО РЕЗУЛЬТАТИ АНАЛІЗІВ

Перше, що нам потрібно, – це спосіб представлення груп і їх результатів аналізів. У мові Python це робиться дуже просто за допомогою вкладеного словника.

Словник зручний для експериментів з алгоритмами і для ілюстрації. У ньому легко проводити пошук і зміни.

Хоча в словнику, що знаходиться в пам'яті, можна зберегти багато результатів аналізів, великі набори даних краще зберігати в базі.

ВІДШУКАННЯ СХОЖИХ РЕЗУЛЬТАТІВ АНАЛІЗІВ

Зібравши дані про те, які групи мають подібні аналізи, потрібно якось визначити, наскільки ці аналізи схожі. Для цього кожна група порівнюється з усіма іншими і обчислюється коефіцієнт подібності (або оцінка подібності). Для цього є кілька способів. Серед них: евклідова відстань і коефіцієнт кореляції Пірсона.

ОЦІНКА НА ОСНОВІ ОБЧИСЛЕННЯ ЕВКЛІДОВІЙ ВІДСТАНІ

Евклідова відстань визначає відстань між двома точками в багатовимірному просторі. Це та відстань, яку ви вимірюєте за допомогою звичайної лінійки. Відстань між точками з координатами (p_1, p_2, p_3, p_4, ...) і (q_1, q_2, q_3, q_4, ...) виражається за формулою В.1.

$$\sqrt{(p_1-q_1)^2+(p_2-q_2)^2+\cdots+(p_n-q_n)^2}=\sqrt{\sum_{i=1}^{n}(p_i-q_i)^2} \quad (1)$$

Один з найпростіших способів обчислення оцінки подібності – це евклідова відстань. У цьому випадку результати аналізів, які є спільними для груп, представляються у вигляді координатних осей. Тепер в цій системі координат можна розташувати точки, відповідні групам, і подивитися, наскільки вони виявилися близькі. Чим ближче дві групи в просторі аналізів, тим більше вони схожі.

Щоб обчислити відстань між групами на діаграмі, візьмемо різниці координат по кожній осі, піднесемо їх до квадрата, додамо, а потім винесемо квадратний корінь з суми. У Python для піднесення до квадрата можна скористатися функцією pow(n,2), а для обчислення квадратного кореня служить функція sqrt модуля math:

```
>> from math import sqrt
>> sqrt(pow(p1-q1)+pow(p2-q2))
```

Відстань, обчислена за цією формулою, буде тим менше, чим більше схожості між групами. Однак нам потрібна функція, значення якої тим більше, чим групи більш схожі одна на одну. Цього можна домогтися, додавши до обчисленої відстані 1 (щоб ніколи не ділити на 0) і взявши зворотну величину:

```
>> 1/(1+sqrt(pow(p1-q1)+pow(p2-q2)))
```

Нова функція завжди повертає значення від 0 до 1, причому 1 виходить, коли результати аналізів двох груп в точності збігаються. Тепер можна зібрати все докупи і написати функцію для обчислення оцінки подібності:

```
from math import sqrt
def sim_distance(prefs,person1,person2):
si={}
for item in prefs[person1]:
if item in prefs[person2]:
si[item]=1
if len(si)==0: return 0
sum_of_squares=sum([pow(prefs[person1][item]-prefs[person2][item],2)
for item in prefs[person1] if item in prefs[person2]])
return 1/(1+sum_of_squares)
```

Цією функцією при виклику передаються імена двох груп, для яких потрібно обчислити оцінку подібності.

ОЦІНКА НА ОСНОВІ ОБЧИСЛЕННЯ КОЕФІЦІЄНТ КОРЕЛЯЦІЇ ПІРСОНА

Коефіцієнт кореляції Пірсона – це міра кореляції двох змінних. Він приймає значення від 1 до -1, де 1 означає, що кореляція між змінними ідеальна, 0 – що кореляції немає, а -1 – що мається ідеальна зворотна кореляція. Кореляція Пірсона розраховується за формулою (2).

$$r=\frac{\sum XY-\frac{\sum X\sum Y}{N}}{\sqrt{\left(\sum X^2-\frac{(\sum X)^2}{N}\right)\left(\sum Y^2-\frac{(\sum Y)^2}{N}\right)}} \quad (2)$$

Трохи складніший спосіб визначити рівень схожості аналізів груп дає коефіцієнт кореляції Пірсона. Коефіцієнт кореляції – це міра того, наскільки добре два набори даних лягають на пряму. Формула складніша, ніж для обчислення евклідової відстані, але вона дає кращі результати, коли дані погано нормалізовані.

Програма для обчислення коефіцієнта кореляції Пірсона спочатку знаходить аналізи, які є спільними для двох груп, і обчислює суму і суму квадратів числових значень цих аналізів, а також суму добутків аналізів. На останньому етапі знайдені значення використовуються для обчислення коефіцієнта кореляції.

```
def sim_pearson(prefs,p1,p2):
si={}
for item in prefs[p1]:
if item in prefs[p2]: si[item]=1
n=len(si)
if n==0: return 0
sum1=sum([prefs[p1][it] for it in si])
sum2=sum([prefs[p2][it] for it in si])
sum1Sq=sum([pow(prefs[p1][it],2) for it in si])
sum2Sq=sum([pow(prefs[p2][it],2) for it in si])
pSum=sum([prefs[p1][it]*prefs[p2][it] for it in si])
num=pSum-(sum1*sum2/n)
den=sqrt((sum1Sq-pow(sum1,2)/n)*(sum2Sq-pow(sum2,2)/n))
if den==0: return 0
r=num/den
return r
```

Ця функція повертає значення від -1 до +1. Значення 1 означає, що дві групи мають в точності однакові результати аналізів. На відміну від евклідової метрики, масштабувати повернене значення для приведення до потрібного діапазону не потрібно.

РАНЖУВАННЯ ОТРИМАНИХ РЕЗУЛЬТАТІВ

Маючи функції для порівняння двох груп, можна написати функцію, яка буде обчислювати оцінку подібності всіх існуючих груп з даною групою і шукати найкращу відповідність.

Функція, яка створює список груп, аналізи яких схожі на аналізи заданої групи:

```
def topMatches(prefs,person,n=5,similarity=sim_pearson):
scores=[(similarity(prefs,person,other),other)
for other in prefs if other!=person]
scores.sort( )
scores.reverse( )
return scores[0:n]
```

Ця функція порівнює дану групу з усіма іншими, які зберігаються в словнику за допомогою однієї з раніше визначених метрик, застосовуючи для цього трансформацію списку і повертає перші n елементів відсортованого списку результатів.

ВИСНОВКИ

Ми розглянули дві різні метрики, але є й багато інших способів виміряти подібність двох наборів даних. Який з них оптимальний – залежить від конкретних випадків. На практиці доцільно спробувати коефіцієнт Пірсона, і метрику на основі обчислення відстані Евкліда, і визначити, яка з них дає найкращий результати.

Для обчислення подібності можна брати інші функції, наприклад коефіцієнт Жаккарда або Манхеттенська відстань, за умови що у них така ж сигнатура і значення яке повертається з плаваючою крапкою тим більше, чим вища подібність.

Техніка, яку ми застосовували, називається коллаборативною фільтрацією за подібністю груп. Альтернатива назва «коллаборативної фільтрація на основі схожості зразків». Коли набір даних дуже великий, коллаборативна фільтрація за схожістю зразків може давати кращі результати, причому чимало обчислень можна виконати заздалегідь, тому користувач отримає рекомендації швидше.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Воронцов К. В. Методы коллаборативной фильтрации и их применение. Сборник материалов семинара Б. Г. Миркина, М: ВШЭ, 2008.
2. Сегаран. Т. Программируем коллективный разум. / Т. Сегеран; [пер. с англ. А. Слинкина]. – СПб.: Символ-Плюс, 2008. – 368 с.
3. Xiaoyuan Su and Taghi M. Khoshgoftaar A Survey of Collaborative Filtering Techniques A Survey of Collaborative Filtering Techniques (англ.) // Hindawi Publishing Corporation, Advances in Artificial Intelligence archive, USA : журнал. — 2009. — С. 1 - 19.
4. Fleder D., Hosanagar K. Blockbuster Culture's Next Rise or Fall: The Impact of Recommender Systems on Sales Diversity (англ.) // Management Science, Vol. 55, No. 5, May 2009, pp. 697-712 : журнал. — 2009. — С. 1 - 49.