پالایش مشارکتی (Collaborative filtering) یکی از تکنیک‌های پرطرفدار سیستم‌های پیشنهاد دهنده، مناسب شرکت‌های بزرگ با مقدار داده‌های بزرگ است. در این گزارش ابتدا به بررسی یکی از کتابخانه‌های محبوب برای مقایسه سیستم‌های پیشنهاد دهنده و سپس به بررسی چگونگی محاسبه شباهت بین کاربران و آیتم‌ها می‌پردازیم.

شماره‌ی ۲ گزارش پیشرفت پروژه

پالایش مشارکتی سیستم پیشنهاد دهنده | ۱ اردیبهشت ۱۴۰۰

فائزه سادات سعیدی نژاد

در گزارش قبلی دیدیم که چگونه و بر اساس چه متریک‌هایی، الگوریتم‌های پیشنهاد دهنده را ارزیابی می‌کنیم. همانطور که دیدید، نوشتن کدهای ارزیابی الگوریتم‌های پیشنهاد دهنده بدون استفاده از کتابخانه‌های موجود، کار سخت و زمان بری است.

برای ارزیابی بهتر و آسان‌تر و در نهایت مقایسه‌ی الگوریتم‌های پیشنهاد دهنده با یکدیگر، از کتابخانه‌ی SuperiseLib[1] استفاده می‌کنیم. این کتابخانه یک کلاس پدر (AlgoBase) به ما می‌دهد و ما می‌توانیم کلاس‌های تجزیه مقدارهای منفرد[2]، کی-نزدیک‌ترین همسایه[3]، تجزیه مقدارهای منفرد++[4] و الگوریتم پیشنهاد دهنده‌ی خودمان[5] را پیاده سازی کنیم با استفاده از ارث‌بری از کلاس پدر که با SurpriseLib هم سازگار است. برای پیاده‌سازی الگوریتم سفارسی خودمان به‌صورت زیر می‌توانیم عمل کنیم و یک تابع برآورد[6] بنویسیم:

```
class MyOwnAlgorithm(AlgoBase):

    def __init__(self):
        AlgoBase.__init__(self)

    def estimate(self, user, item):
        return 3
```

برای ارزیابی الگوریتم‌ها در این کتابخانه تنها کافی‌ست که از تابع EvaluatedAlgo(AlgoBase) استفاده کنیم و درصورت نیاز به داده برای ارزیابی الگوریتم، می‌توان به راحتی از تابع EvaluationData(Dataset) دو دسته داده‌ی آموزشی[7] و تستی[8] را فراهم کرد. برای مقایسه‌ی دو الگوریتم پیشنهاد دهنده هم می‌توان از تابع Evaluator(DataSet) استفاده کرد. این کتابخانه دقت، Hit rate، گوناگونی، تازگی و پوشش‌دهی، که در گزارش قبل به توضیحشان پرداختیم، را ارزیابی می‌کند.

بخش عملی

کدهای این بخش از گزارش در فولدر Framework هستند. فایل RecsBackOff.py را اگر اجرا کنیم، می‌توانیم نتیجه‌ی مقایسه‌ی الگوریتم SVD (که در حال حاضر یکی از الگوریتم‌های خوب برای سیستم‌های پیشنهاد دهنده است) با یک الگوریتم تصادفی را ببینیم.

درست است که موضوع پروژه پالایش مشارکتی است، اما یکسری موارد بین همه‌ی سیستم‌های پیشنهاد دهنده یکسان است، به همین دلیل کمی به ویژگی‌های پالایش محتوا محور[9] می‌پردازیم.

---

[1] http://surpriselib.com/
[2] Singular value decomposition (SVD)
[3] K-nearest neighbors (K-nn)
[4] SVDpp
[5] Custom
[6] Estimate function
[7] Train set
[8] Test set
[9] Content-base filtering

پالایش محتوا محور

پالایش محتوا محور نوعی از سیستم پیشنهاد دهنده است که در آن بر اساس یکسری از ویژگی‌های محتوای مورد علاقه‌ی کاربر مورد نظر (کاربری که می‌خواهیم به آن محتوای جدید پیشنهاد دهیم) مانند سال تولید، ژانر، کارگردان و... محتواهای شبیه به آن را پیدا کرده و به کاربر پیشنهاد می‌دهد. در این نوع از سیستم پیشنهاد دهنده، اندازه‌گیری میزان شباهت[10] محتواها با یکدیگر یک امر مهم است. برای اندازه‌گیری میزان شباهت، ۶ معیار وجود دارد: شباهت کسینوسی[11]، کسینوس تنظیم‌شده[12]، شباهت پیرسون[13]، همبستگی رتبه‌ای اسپیرمن[14]، اختلاف میانگین مربعات[15]و شباهت ژاکارد[16].

برای مثال، اگر براساس ژانر محتوا بخواهیم شباهتشان را بدست آوریم با استفاده از شباهت کسینوسی به راحتی می‌توان این کار را انجام داد (توضیحات بیشتر در رابطه با چگونگی عملکرد این معیار اندازه‌گیری شباهت، در مقاله‌ی اصلی آورده می‌شود). فرض کنید تعداد ژانرها بیشتر از ۲تا باشد، در اینصورت فرمول بصورت چند بعدی عمل می‌کند، به طوری که هر بعد در این مثال، بیانگر ژانر است. فرمول آن بصورت زیر است:

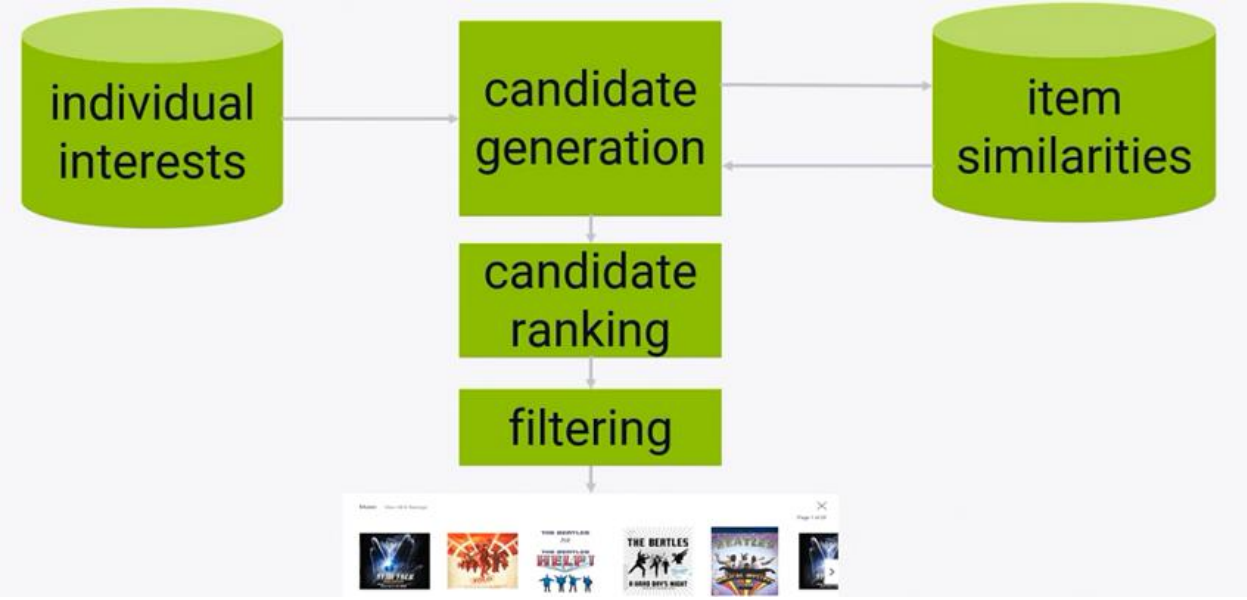
$$CosSim(x,y) = \frac{\sum_i x_i y_i}{\sqrt{\sum_i x_i^2}\sqrt{\sum_i y_i^2}}$$

در این فرمول x و y دو محتوایی هستند که می‌خواهیم شباهت بینشان را محاسبه کنیم و i بُعد است. شباهت کسینونی معیاری است پر استفاده که علاوه بر پالایش محتوا محور، در الگوریتم‌های پیشنهاد دهنده‌ی دیگری هم مورد استفاده قرار می‌گیرد. حال باید از این شباهت بین محتواها استفاده کنیم و پیش بینی امتیازدهی را انجام دهیم، که این کار را می‌توان با استفاده از کی-نزدیکترین همسایه انجام داد.

[10] Similarity score
[11] Cosine similarity
[12] Adjusted cosine
[13] Pearson similarity
[14] Spearman rank correlation
[15] Mean squared difference (MSD)
[16] Jaccard similarity

پالایش مشارکتی

در پالایش مشارکتی، سیستم کاربران شبیه به کاربر مورد نظر را پیدا می‌کند و محتوایی را که کاربران شبیه به کاربر ما دوست داشته‌اند یا خریده‌اند را به او پیشنهاد می‌دهد. این عملیات بصورت زیر انجام می‌شوند:



در قسمت Candidate generation پیشنهادات را تولید می‌کند، در قسمت Candidate ranking محتواهای انتخابی را رده‌بندی می‌کنیم براساس اینکه کاربر مورد نظر چقدر آن محتوا را دوست خواهد داشت و سپس در قسمت Filtering محتواهایی که کاربر قبلا آن‌هارا دیده، از لیست حذف می‌کند.

در پالایش مشارکتی، پیدا کردن محتواها یا کاربران شبیه به کاربر مورد نظر، از چالش‌های اصلی به حساب می‌آید که می‌توان به همان روش‌هایی که برای پالایش محتوامحور ذکر شد، این کار را انجام داد.

برای استفاده از شباهت کسینوسی در پالایش مشارکتی، ابعاد را به جای ژانر محتوا، رفتار کاربران در نظر می‌گیریم. یکی از چالش‌های استفاده از شباهت کسینوسی در پالایش مشارکتی، پراکنده[17] بودن داده‌ها است، داده‌ی کافی از رفتار کاربران نداریم تا با آن کار کنیم، به همین دلیل است که از پالایش مشارکتی در شرکت‌های بزرگ با داده‌های زیاد استفاده می‌شود که مشکل داده ندارند. برای همین در پالایش مشارکتی می‌توان از کسینوس تنظیم‌شده هم استفاده کرد، در کسینوس تنظیم‌شده شباهت کاربران بر اساس امتیازدهی‌شان را بدست می‌آورد و فرمولش بصورت زیر است:

$$CosSim(x,y) = \frac{\sum_i \left((x_i - \bar{x})(y_i - \bar{y})\right)}{\sqrt{\sum_i (x_i - \bar{x})^2}\sqrt{\sum_i (y_i - \bar{y})^2}}$$

[17] Sparsity

همانطور که در فرمول مشاهده کردید، اختلاف امتیاز دهی کاربر به آیتم موردنظر و میانگین امتیاز دهی‌اش به فیلم‌ها است. روش دیگری برای اندازه‌گیری شباهت بین محتواها یا کاربران وجود دارد که در آن اختلاف بین امتیاز کاربر به آن محتوای به‌خصوص و میانگین امتیازهایی که همه کاربران به آن فیلم داده‌اند را محاسبه می‌کند که به این روش شباهت پیرسون می‌گویند. و فرمولش بصورت زیر است:

$$CosSim(x,y) = \frac{\sum_i \left((x_i - \bar{i})(y_i - \bar{i})\right)}{\sqrt{\sum_i (x_i - \bar{i})^2}\sqrt{\sum_i (y_i - \bar{i})^2}}$$

سه روش دیگر هم برای اندازه‌گیری میزان شباهت وجود دارد که زیاد مورد استفاده قرار نمی‌گیرند و من در مقاله‌ی اصلی به آن‌ها می‌پردازم.