UNIVERSITEIT VAN AMSTERDAM

Verslag 2 – Collectieve Intelligentie

Groep 4: Dennis de Buck, Cesar Groot Kormelink, Enzo Delaney-Lamour en Maik Larooij

2.0 Taakverdeling en planning (update)

De verdeling van de taken zijn niet veranderd ten opzichte van vorige week. We hebben nog steeds twee personen die aan één algoritme werken. We helpen elkaar natuurlijk wel waar dat nodig is en kan. De planning is wel aangepast ten opzichte van vorige week. Dit komt omdat de planning waarbij we volgende week (acht) pas zouden evalueren erg krap werd. We hadden dan maar 2 dagen om te evalueren en eigenlijk bijna geen tijd om nog echt aanpassingen te doen. We hebben dus nu besloten deze week (zeven) de algoritmes te bouwen en te evalueren en volgende week vooral bezig te gaan met de puntjes op de i en de presentatie voorbereiden.

2.1 Algoritmes

Content based:

1. Laad de training en test data voor een bepaalde stad
2. Genereer een dataframe met alleen restaurants
3. Kies een threshold voor het aantal reviews dat de gebruiker moet hebben om een voorspelling te kunnen doen.
4. Genereer een lijst met alle users met meer reviews dan de gekozen threshold in de gekozen stad.
5. Maak een utility matrix, waarbij de bedrijven de index, users de columns en de ratings de ingevulde waarden zijn.
6. Zorg dat elke aparte categorie die kenmerkend is voor het bedrijf, in een aparte rij komt te staan, met alleen het identiteitsnummer van het bedrijf. Als een bedrijf het woord “restaurant” heeft staan, wordt deze niet meegenomen omdat de gehele subdataset uit restaurants bestaat.
7. Creëer een binary matrix met de in stap 6 gegenereerde data. Bedrijven als index en de categorieën op de columns. Als een bedrijf binnen de categorie valt staat er een 1, anders een 0.
8. Met deze matrix wordt er een Jaccard matrix gevormd. Deze zal dienen als similarity maat.
9. Per user wordt er gekeken naar het bedrijf uit de reviews die hoger dan een 4 hebben. Van deze bedrijven wordt met de Jaccard matrix gevonden welke andere bedrijven vergelijkbare categorieën hebben, die bedrijven worden returned.
10. Creëer een dataframe met alleen de gebruikte users en restaurants, met als info de gegeven reviews.
11. Vervolgens worden er aan de hand van de similarity matrix, utility matrix en de bruikbare reviews, cijfers voorspeld voor elke user voor elk bedrijf.

Allereerst hebben we de bedrijven die niet als restaurant gecategoriseerd waren er uit gefilterd bij stap 3. Echter, bleven er wel restaurants over die alleen ‘restaurant’ als categorie hadden. Deze bedrijven zouden de werking van het algoritme verslechteren. We hebben daarom al die bedrijven gefilterd uit de dataset bij stap 4. Hierdoor bleven er alleen restaurants over met categorieën die goed te gebruiken waren om similarity scores uit te rekenen.

Bij stap 3 kan de treshold die aangeeft hoeveel reviews een user minimaal moet hebben als parameter worden meegegeven. Per stad kan het verschillen hoeveel reviews een user nodig heeft om een goede voorspelling te doen. Als er te weinig users zijn die minimaal 5 reviews hebben kan de treshold verlaagd worden om een groter aantal users te krijgen. Met een treshold die zo hoog mogelijk is, maar wel met genoeg users, kan je betere voorspellingen doen.

In de users-tabel stond dat sommige users meer dan 100 reviews hadden. Wanneer wij de daadwerkelijke reviews van de user in de gekozen stad wouden vinden, kwamen we er achter dat dit nummer het totaal aantal reviews was, en wij wouden graag gebruikers die meer dan een bepaald aantal reviews hadden in de stad van onze trainingsset. Hierdoor hebben wij een functie geïmplementeerd die de database met restaurants afspeurt en kijkt per gebruiker of hij evenveel of meer reviews dan de treshold heeft achtergelaten.

Collaborative filtering:

1. Laad de training en test data
2. Kies een persoon en restaurant om een rating te voorspellen
3. Filter de data zodat alleen **restaurants** overblijven waarvan reviews zijn van personen die in ieder geval 5 reviews hebben geschreven
4. Filter de data zodat er 200 random gekozen restaurants over blijven die voldoen aan een minimum van k aantal reviews
5. Genereer een utility matrix, met als index de restaurants en als columns de gebruikers
6. Maak op basis van deze utility matrix een ‘mean centered’ utility matrix
7. Genereer een similarity matrix op basis van het gekozen algoritme (Cosine, Euclidian of Minkowski distance)
8. Maak een lijst met restaurants die het meest lijken op het gekozen restaurant bij stap 2, waar de gekozen gebruiker een review voor heeft gegeven
9. Reken een voorspelde rating uit op basis van een gewogen gemiddelde (hoge similarity telt meer mee)
10. Kijk of deze rating boven een vooraf gestelde threshold uitkomt. Zo ja, raadt dit item aan.

Om ervoor te zorgen dat er niet te weinig data is over bepaalde gebruikers hebben we gekozen om gebruikers die minder dan 5 reviews hebben geschreven bij stap 3 niet mee te nemen. Voor deze groep gebruikers is collaborative filtering niet goed uit te voeren. Om nog meer bruikbare data eruit te filteren is gekozen om bij stap 4 200 random bedrijven uit te kiezen die boven een bepaalde threshold uitkomen. De threshold is ingesteld om bedrijven met weinig reviews weg te filteren. Omdat er vaak memory errors ontstonden bij het berekenen van bijvoorbeeld een utility matrix hebben we gekozen om een subselectie te maken van bedrijven. Het random element

geeft wat ons betreft een leuke twist aan het aanbevelingssysteem: er worden niet altijd dezelfde restaurants aangeraden.

Als in een grote stad alle restaurants meegenomen werden ontstond er een memory error van de computer. Daarom is gekozen om een threshold in te stellen die verandert naarmate de stad groter wordt. In een klein stadje is het misschien dus nodig om 10 reviews te hebben op je restaurant, maar in een grote stad wel 50. Daarna is er gekozen om hier 200 restaurants uit te halen op willekeurige basis. We zijn ons ervan bewust dat op deze manier nuttige informatie verloren kan gaan, maar het was noodzakelijk om te doen.

Een andere parameter is de 'treshold recommended'. Dit is waarboven een voorspelde rating moet vallen om het bedrijf aan te raden. Als deze hoger ligt, worden er minder restaurants worden aangeraden. Dit zal het aantal false positives doen dalen, maar kan wel ten koste gaan van een laag aantal false negatives.

Eén probleem waar we tegenaan liepen was dat onze computer niet alle restaurants kon vergelijken en een utility matrix ervan kon maken als er heel veel van waren zoals in een grote stad. Dan gaf hij een memory error en daarom hebben we gekozen niet alle restaurants telkens mee te nemen maar een willekeurig geselecteerde groep.

2.2 Baseline

Content based:

Het door ons gebouwde algoritme is gebaseerd op de stad Ajax. Dit is de gebruikte trainingset. Om het algoritme te evalueren kunnen we een andere stad gebruiken, die ook een subset is van de gehele dataset. Bij Content based filtering hebben wij de MSE berekend. Deze kwam uit op 0.43. Wanneer we de MSE berekenen bij random voorspellingen, komt die uit op 3.71. Ook de random voorspellingen zijn gedaan op basis van de stad Ajax. het grote verschil in de uitkomsten betekent dus dat ons algoritme wel degelijk beter voorspelt dan willekeurige voorspellingen.

Collaborative filtering

Het algoritme is gebouwd door middel van een trainingsset op de stad Toronto. Dit is een subset van de gehele yelp dataset. Om ervoor te zorgen dat de testset niet dezelfde reviews bevat hebben we gekozen om de stad Henderson te laten fungeren als testset. Voor het berekenen van de MSE is er voor alle gefilterde reviews een voorspelde rating berekend. We hebben besloten om de voorspelde ratings die precies hetzelfde waren als de echte rating niet mee te nemen in het berekenen van de MSE, aangezien deze ratings waarschijnlijk maar op één ander restaurant zijn gebaseerd en deze de MSE een stuk beter laten uitkomen dan het misschien is. Voor elke similarity maat is er een MSE uitgerekend die is vergeleken met de random baseline.
Zoals te zien in de table scoren de geïmplementeerde algoritmes een stuk beter dan de simpele random baseline. Alle MSE's zijn berekend over dezelfde data. Voor volgende week willen we

nog graag gaan kijken naar de precision en recall curve om er achter te komen wat de beste threshold is om een restaurant aan te raden.

Tabel 1. Resultaten MSE collaborative filtering

Collaboritive filtering	MSE
Euclidian distance	0,322
Cosine similarity	0,333
Minkowski distance	0,384
Random baseline	3,849

2.3 Verduidelijking verslag 1

Bij vraag 1.3c werd ons gevraagd om uit te leggen of we verwachtten dat content based filtering goed werkte. De redenatie achter ons antwoord mist nog een beetje. Hierbij willen we dat graag verduidelijken.

De verwachting was dat content based filtering goed zou werken omdat we eerder hadden uitgelegd dat collaborative filtering, vooral user-based, wat minder zou werken omdat we in de dataset te maken hadden met verschillende categorieën aan bedrijven. Een nagelsalon en een kiprestaurant kunnen dan een hoge similarity vertonen omdat ze toevallig ongeveer dezelfde ratings hebben gekregen. Via content based filtering kan dit probleem verholpen worden aangezien hier juist gefocust wordt op categorieën. Op deze manier worden er nagelsalons aangeraden als de gebruiker positieve interesse toont in nagelsalons. Bij content based wordt er dus vergeleken op basis van de inhoud van het bedrijf, wat in het geval van de verschillende categorieën van deze dataset een voordeel is.

Bij vraag 1.2c was het bij de plots nog niet helemaal duidelijk hoeveel totale restaurants er waren bij de plot van restaurant types (figuur 4). Het totale aantal restaurants staat in de plot erboven aangegeven (eveneens figuur 4). De categorieën zijn dus gebaseerd op 885 restaurants. Wel hebben restaurants vaak meerdere categorieën. Restaurants met geen categorieën of maar 1, bijvoorbeeld alleen ‘restaurant’ zijn niet zo bruikbaar. De vraag rees dus hoeveel van dit soort restaurants voorkwamen in de gehele dataset. Na een test zijn we er achter gekomen dat er slecht 4 van de 885 restaurants niet bruikbaar zijn. Deze worden op dit moment al automatisch weggehaald door ons algoritme: de lege categorieën worden niet meegenomen met het filteren van restaurants en ook de categorie restaurant wordt in het proces weggehaald om alleen te focussen op de typen restaurants.

```
Number of restaurants:
885
Number of restaurants with 0 or 1 categories:
4
```

Figure 1. Aanvulling verdeling business data