

TDK-dolgozat

Tasnádi Ervin

Szegedi Tudományegyetem
Természettudományi és Informatikai Kar
Számítógépes Algoritmusok és Mesterséges Intelligencia
Tanszék

**Szociális kapcsolatok feltérképezése gráf alapú
adatbányászati módszerekkel**

Készítette:
Tasnádi Ervin

Témavezető:
Berend Gábor
egyetemi tanársegéd

Témavezető:
Dr. Farkas Richárd
egyetemi adjunktus

Szeged
2014

Tartalomjegyzék

Bevezetés	1
1. A feladat leírása	3
1.1. A rendelkezésre álló adatok	4
2. Alkalmazott technikák	6
2.1. PageRank algoritmus	6
2.1.1. Véletlen séta interpretáció	8
2.1.2. Perszonalizált PageRank	9
2.2. Maximum entrópia osztályozás	9
3. A jellemzőtér bemutatása	11
3.1. A jellemzőcsoportok leírása	11
3.1.1. A felhasználók nyelvezete	12
3.1.2. Az étteremlátogatási preferenciák megállapítása	14
3.1.3. A kapcsolati háló felhasználása	15
3.2. A jellemzők előállítása a tanuláshoz	15
4. Eredmények	17
4.1. Metrikák	17
4.2. A baseline	19
4.3. Az algoritmus eredményei	19
4.4. A gráfok statisztikái	20
5. Kapcsolódó munkák	23
6. Összegzés	25
6.1. Köszönetnyilvánítás	27
Irodalomjegyzék	29

Bevezetés

Jelen dolgozatban egy olyan gépi tanulási módszert mutatunk be, amely alkalmas arra, hogy egy étteremvéleményező portálból kinyert nagymennyiségű adatból következtetéseket vonjon le a felhasználóknak a portálon tanúsított szokásaiból, a kinyert adatoknak az elemzésével és kiértékelésével ezt követően lehetséges barátokat ajánljon fel számukra.

A Web 2.0 olyan Internetes szolgáltatások gyűjtőneve, melyek szembemennek a web eredeti megközelítésével, azaz azzal, hogy a felhasználók csupán az Interneten keresztül elérhető webszervereken tárolt adatokhoz férhetnek hozzá. Főként úgy jellemezhetjük ezt az új megközelítést, hogy itt már nem csak passzív, egyirányú fogyasztói vagyunk az elérhető tartalmaknak, hanem mi hozhatjuk azt létre, tehetjük publikusan elérhetővé, hogy aztán más felhasználók elérhessék azokat.

Tipikus web 2.0-ás szolgáltatásnak nevezhetjük a webnaplót, azaz a blogot. Míg ezen műfaj megjelenése előtt egy szűk felhasználói kör kiváltsága volt az, hogy a weben a megfelelő informatikai szakértelem birtokában, és anyagi ráfordítással a külvilág számára elérhetővé tegyen tartalmakat, addig egy webnapló létrehozása az alkalmas szolgáltatás kiválasztása után pár kattintással elvégezhető. Így az Internetes tartalommegosztás ma már teljesen egyszerű, és bárki által elvégezhető triviális cselekvés lett. A blogon kívül népszerű web 2.0-ás szolgáltatások például a közösségi-, vagy a videómegosztó oldalak.

Ezen új szolgáltatások erőssége abban rejlik, hogy a felhasználóknak lehetőségük van arra, hogy az Interneten keresztül együttműködjenek. Ennek az egyik lehetséges módja, hogy tapasztalatokat, véleményeket osztanak meg egymással, így könnyen profitálni tudnak a mások által megszerzett tudásból.

Ilyen portál a Yelp¹ is, ahonnan az adataink származnak. A portálon éttermeket találhatunk, amiket értékelni tudunk. A portál megengedi, hogy többféleképpen kifejezzük a véleményünket egy étteremről: akár szövegesen le is írhatjuk azt, de azt is választhatjuk, hogy egy 1-5-ig terjedő skálán csillagokkal osztályozzuk így megosztva a tapasztalatainkat. A véleményeket mindenki olvashatja, és a véleményeket is lehet értékelni: ha az egyik felhasználó szerint az étel minősége hagy némi kívánnivalót maga után, de egy másik felhasználó is étkezett már ugyanabban az étteremben, és neki jó tapasztalatai voltak, akkor ezt megoszthatja komment formájában, vagy

¹ A Yelp a következő URL címen érhető el: <http://yelp.com>

pedig a rossz véleményt negatívnak értékelheti. Egy harmadik felhasználónak ezek után már sokkal több információja van arról, hogy érdemes-e ezt az éttermet választani.

Ez sokszor problémás lehet: ha egymásnak ellentmondó információk állnak rendelkezésre egy bizonyos étteremről, akkor érdemes lehet azokat a véleményeket súlyozni azért, hogy javítsunk a véleményező portál használhatóságán: ha valakiről tudjuk, hogy a véleménye a legtöbb esetben csak pozitív, vagy negatív, akkor segíthetjük egy felhasználó számára a döntést úgy, hogy az ilyen véleményeket megjelöljük, vagy hátrébb soroljuk a többi vélemény között. A másik oldalról viszont azt is gondolhatnánk, hogy ha valakivel barátságban vagyunk, akkor az ő véleménye fontos lehet számunkra. Ez alapján jogosan feltételezhetjük, hogy jó ötlet, ha a felhasználók ismerősnek jelölhetik egymást, majd a véleményező portál a barátok véleményét mindig előresorolja, vagy valamilyen módon kiemeli azért, hogy elősegítse a jó döntés meghozatalát.

Ebben a dolgozatban feltesszük a kérdést, amelyet úgy fogalmazhatunk meg: milyen mértékben befolyásolják a szociális kapcsolatokat a felhasználói szokások? A kérdés megválaszolásához felhasználjuk a rendelkezésre álló információt minden felhasználóról, és több szempontból megvizsgáljuk ezeket.

A célunk az, hogy barátokat keressünk a felhasználóknak, majd azokat felajánljuk nekik: ha a felhasználó elfogadja a felajánlást, akkor a további döntések meghozatalánál az új kapcsolat is elősegítheti egy jó döntés meghozatalát.

1. fejezet

A feladat leírása

Gépi tanulás alatt olyan rendszerek összességét értjük, amelyek képesek arra, hogy tapasztalatok gyűjtésével egyre jobban teljesítsenek egy célfeladat végrehajtásában, következtetéseket vonjanak le bemeneti adatokból, melyeket **minták**nak nevezünk. Ezeknél a rendszereknél nem elegendő az, ha egyszerűen memorizálni próbálják a mintapéldákat: képesnek kell lenniük arra, hogy általánosítsanak, és a tanulási folyamat végeztével egy még nem látott mintapéldány elemzése után többnyire helyes következtetéseket vonjanak le arról. A gépi tanulásnak több területe van, mi csak a **felügyelt tanulás**ra fogunk koncentrálni.

Felügyelt tanulás esetén az a célunk, hogy megtanuljuk a leképezést egy x bemeneti vektor és a hozzá rendelt y kimeneti érték között. Ehhez rendelkezésre áll a tanítóhalmaz, amely egy vektorokból, és a hozzárendelt értékekből álló párokból álló N elemű halmaz (N a tanítóhalmaz számosságát jelenti): $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$. Az $\mathbf{x}_i$ jelenti az i -dik tanítópéldát, ami a jellemzőértékek vektora, melyet a legtöbbször valós számokkal reprezentálhatunk, melyet m darab **jellemző** határoz meg. Az y_i -t célváltozónak nevezzük, az értékére nincsen megkötés: lehet valós szám, vagy egész, de diszkrét értéket is felvehet (amelyet általában ugyancsak számokkal kódolunk).

Azt az esetet, amikor y_i folytonos, **regresszió**nak, azt az esetet pedig, amikor diszkrét értékeket vesz fel, **osztályozás**nak nevezzük. Az osztályozáson belül megkülönböztethetjük azt az esetet, amikor egy példának két értéke lehet: vagy egy bizonyos kategóriába tartozik, vagy nem – ezt **bináris osztályozás**nak hívjuk –, illetve azokat az eseteket, amikor egy példa több kategóriába is tartozhat egyszerre, amit **multinomiális osztályozás**nak hívunk.

Jelen dolgozatban pozitív, címkézetlen példányalapú tanulással, a bináris osztályozás egyik speciális esetével foglalkozunk, ahol csak pozitív tanulópéldák állnak rendelkezésre.

Legyen a továbbiakban U a felhasználók halmaza. Azt, hogy az $a \in U$ felhasználónak a $b \in U$ felhasználó a barátja, úgy jelöljük, hogy $\text{barátság}(a, b)$. A barátság jelen esetben nem szimmetrikus reláció, tehát ha $\text{barátság}(a, b)$ teljesül, abból még nem következik, hogy $\text{barátság}(b, a)$ is igaz lesz. Természetesen a barátság több a többhöz kapcsolatot definiál a felhasználók között.

A feladatunk precízebb megfogalmazásban az, hogy válasszunk ki egy $a, b \in U$ felhasználóból álló (a, b) párost, melyről nem tudjuk, hogy $\text{barátság}(a, b)$ teljesül-e, és arra vagyunk kíváncsiak, hogy mekkora annak a valószínűsége, hogy a b felhasználó az a barátja, azaz hogy teljesül-e rájuk a $\text{barátság}(a, b)$ reláció. Így a feladatot visszavezettük egy osztályozási problémára: egy (a, b) felhasználó párhoz kell az „igen” címkét rendelni akkor, ha úgy gondoljuk, hogy ők barátok (azaz a -nak b az egyik barátja), vagy a „nem” címkét, ha úgy gondoljuk, hogy nem azok.

A reláció előrejelzéséhez rendelkezésre állnak pozitív, és címkézetlen tanítópéldák: az adathalmazban minden felhasználónál fel van sorolva az összes barátja, így elő tudunk állítani pozitív tanítópéldákat. A negatív példák meghatározásához véletlenszerűen választunk olyan felhasználó párokat, akik között nincs definiálva a barátság reláció a megfelelő irányban, azaz olyan (a, b) -ket, melyekre $\text{barátság}(a, b)$ -ről nem tudjuk megmondani, hogy teljesül-e. Ennek az az oka, hogy az adathalmazban csak a barátságokról vannak információink.

1.1. A rendelkezésre álló adatok

A rendelkezésre álló adathalmazból a felhasználókat, és az éttermekről alkotott véleményeket leíró adatstruktúrákat fogjuk felhasználni, melyek a bevezetőben említett Yelp-ről származnak.

Minden felhasználót egy JSON objektum ír le, mely a következő formátumban van (a lényeget kiemelve):

```
{
  ...
  "user_id": "MWhR9LvOdRbqtu1I_DRFBg",
  "friends": [
    "8Y2EN4XNNhnwssuPb31sJg",
    "A1jPleJ99kXZ3t9wQ3np-g",
    "resYiOoGkQg6q0qgtj_1GA",
    "sk11OnkjMqD4GdFpVhU88Q",
    ...
    "fdSfg45S-P-FcVF-VIUOew"],
  ...
}
```

Az általunk felhasznált információt a `user_id` nevű felhasználói azonosító és a `friends` tömb tartalmazza. A `friends` a felhasználó barátainak az azonosítója – ez egy karakterláncokból álló tömb értelemszerűen, amely a barátok azonosítóját jelenti. A `user` fájl a később

bevezetésre kerülő barátság gráf felépítéséhez használjuk fel, amely a kapcsolati hálót leíró jellemzőnek a bemenete; ezek a későbbi fejezetekben lesznek részletesen kifejtve.

A véleményeket a `reviews` fájl tartalmazza, melyben egy konkrét vélemény formátuma a következő formában van:

```
{
  ...
  "user_id": "Xqd0DzHaiyRqVH3WRG7hzhg",
  "text": "dr. goldberg offers everything i look for in a general practice. he's nice and easy to talk to without being patronizing; he's always helpful in seeing his patients; he's affiliated with a top-notch hospital (which my parents have explained to me is very important in case something happens and you need surgery; and you can get referrals to see specialists without having to see him first. really, what more do you need? i'm sitting here trying to think of any complaints i have about him, but i'm really drawing a blank.",
  "business_id": "vcNAWiLM4dR7D2nwwJ7nCA"
  ...
}
```

A `reviews` fájlban található objektumok egy felhasználót kapcsolnak össze egy étteremmel, és a felhasználó által írt véleménnyel az adott étteremről. A felhasznált információ ebből a `user_id` mező, ami a véleményt író felhasználónak az azonosítója, a `business_id` mező az étterem azonosítója, illetve a `text` maga a vélemény szöveges formátumban. Ezt a fájlt majd a később bevezetésre kerülő felhasználó-szó, illetve felhasználó-vélemény gráfok felépítéséhez fogjuk felhasználni, a jellemzők értékeinek előállítása érdekében. A céljaink eléréséhez az adatbázisnak csak egy részével rendelkezünk; a teljes adatbázist nem tették nyilvánossá a Yelp üzemeltetői, körülbelül az egy százalékát használhatjuk fel akadémiai célokra. Ez azt jelenti, hogy a vélemény fájlban 1,2 millió rekord van, a felhasználók közül pedig összesen alig több, mint 250 ezer.

2. fejezet

Alkalmazott technikák

2.1. PageRank algoritmus

A jellemzők kialakítása során szükségünk lesz arra, hogy megállapítsuk egy gráfban az összes, vagy bizonyos pontok fontosságát. A probléma egy lehetséges megoldását a PageRank algoritmus adja [7] [8], amely a bemeneti gráf minden pontjához egy nulla, és egy közötti intervallumba eső valós számot rendel, melyeknek az összege 1, így egy valószínűségi eloszlást definiál a csúcsok felett. Egy $G = (V, E)$ irányított, vagy irányítatlan gráf $p \in V$ pontjának PageRank-je s , ha a PageRank algoritmus a p ponthoz az s valós számot rendeli. Minél nagyobb egy pont PageRank-je, annál „fontosabb” a gráfban. Azt a vektort, ami a gráf pontjaihoz a PageRank értéküket jelöli, r -rel fogjuk jelölni, és a **PageRank vektornak** hívjuk.

A PageRank algoritmus szerint egy p pont fontosságát az határozza meg, hogy mely másik pontokból vezet p -be él, figyelembe véve azok fontosságát és közvetlen szomszédjaik számát. Erre úgy tekinthetünk, hogy minden pont rendelkezik bizonyos mennyiségű szavazattal, melyeket továbbadhat a ki-élein keresztül a szomszédos pontoknak, illetve szavazatokat kaphat a be-élein keresztül szomszédos pontoktól.

Minden csúcs a fontosságához mérten kap szavazatot, amit egyenlő arányban oszt szét: jelentse d_j a j pont kimenő éleinek számát. Ha a j pontnak r_j szavazata van, és d_j darab kimenő éle, akkor minden pont, amelybe vezet él j -ből, r_j/d_j szavazathoz fog jutni. Egy pont fontosságát pedig úgy kapjuk meg, hogy összegezzük a bemenő éleken kapcsolódó szomszédos pontok szavazatszámát. Ha d_i jelenti az i pontból induló élek számát, $i \rightarrow j$ pedig azt jelenti, hogy i -ből megy él j -be, akkor ezt a következő formában írhatjuk le röviden:

$$r_j = \sum_{i \rightarrow j} \frac{r_i}{d_i} \quad (2.1)$$

Ez úgy interpretálható a valóságban, hogy ha egy fontos csúccsal vagyunk kapcsolatban, akkor az a csúcs sok szavazattal rendelkezik, így többet tud nekünk továbbosztani, ha pedig

emellett még az a feltétel is fennáll, hogy kevés másik ponttal áll kapcsolatban rajtunk kívül, akkor a szavazata kevés részre oszlik, így a mi fontosságunkat is arányosan növeli.

Ez az algoritmus használható például arra, hogy weboldalakot állítsunk sorba fontosság szerint. Egy gráfot építünk, melynek a csúcsai a weboldalaknak feleltethetők meg, az a pontból b -be mutat egy irányított él, ha az a által reprezentált weboldalon van egy link a b -nek feleltethető oldalra. Ha egy weboldal népszerű, például a Wikipedia ¹, akkor sok másik weboldal hivatkozik rá. Ekkor ezek az oldalak a Wikipédiának adják a szavazataik egy részét. Mivel az sok helyről kap szavazatot, ezért a Wikipédia is sok szavazattal fog rendelkezni azért, mert a szavazatok összeadódnak. Egy weboldal fontosabbá válik, ha egy másik fontos weboldal hivatkozik rá. Ez bekövetkezhet például akkor, ha az indexre kikerül egy link egy idáig kevésbé fontos weboldalról. Ekkor a fontossága nőni fog, hiszen egy sok szavazattal rendelkező oldal továbbosztotta neki a szavazatainak egy részét.

Kézenfekvő megoldás lenne a 2.1 egyenletrendszer egy direkt módszerrel megoldani: például a Gauss elimináció valamelyik változatával. Sajnos látni fogjuk, hogy, a gráf méretének növekedésével az M mátrix mérete is nő, így nagy gráfok esetén nagy lesz az M mátrix is, a nagy mátrixokra pedig a direkt módszerek nem kellőképp hatékonyak [10]. Így más módszerhez kell folyamodnunk, hogy meghatározzuk a PageRank értékeket.

Ehhez vezessük be az M négyzetes szomszédsági mátrixot (ami egyben egy sztochasztikus mátrix), és tegyük fel, hogy az i pontnak d_i kimenő éle van, és ebben az esetben,

$$M_{ij} = \begin{cases} M_{ij} = \frac{1}{d_i}, & \text{ha létezik él } i\text{-ből } j\text{-be} \\ 0, & \text{különben} \end{cases} \quad (2.2)$$

Ekkor az M mátrix minden oszlopa egyre összegződik, tehát teljesíti a sztochasztikus mátrix feltételeit, amit a következőképpen tudunk kiaknázni:

Legyen az $\mathbf{r}$ a fenti $\mathbf{r}$ vektorral megegyező, a csúcsokhoz PageRank értékeket rendelő vektor, és teljesül, hogy

$$\sum r_i = 1$$

Így a PageRank egyenlet a következő mátrixos formában írható fel, amely megegyezik a szummajeles alakkal:

$$\mathbf{r} = M\mathbf{r} \quad (2.3)$$

Ekkor sajátvektora az M mátrixnak, az $\mathbf{r}$ -hez tartozó sajátérték pedig 1. Mivel M egy sztochasztikus mátrix, nem tartalmaz negatív értékeket. Ebből, és az $\mathbf{r}$ vektorra tett meg-megszorításunkból következik, hogy:

¹<http://wikipedia.hu>

$$M\mathbf{r} \leq 1 \quad (2.4)$$

Ha ezt a feltételt, és a PageRank egyenletet összevonjuk, megállapíthatjuk, hogy az M -hez tartozó legnagyobb sajátérték az 1.

Ekkor alkalmazható az M mátrix ismeretében a hatványmódszer, mely pontosan a legnagyobb sajátértékhez tartozó sajátvektort fogja eredményül adni, azaz az $\mathbf{r}$. Így ahelyett, hogy valamely direkt módszerrel próbálnánk megoldani az egyenletet, választhatunk egy iterációs módszert, ha beletörődünk abba, hogy csak a megoldás egy bizonyos pontosságú közelítését kaphatjuk meg.

2.1.1. Véletlen séta interpretáció

Tegyük fel, hogy teszünk a gráfban egy véletlen sétát a következő szabályok szerint:

- Bármely t időpillanatban valamely i ponton vagyunk éppen
- A $t + 1$ időpillanatban az i pontból a ki-élek közül választunk egyet teljesen véletlenszerűen, melyen keresztül a j pontba jutunk.

Ekkor egy olyan pontba fogunk jutni, amelybe el lehet jutni i -ből. Ezt a végtelenségig ismételjük.

Legyen $\mathbf{p}(t)$ egy olyan vektor, amelynek az i . koordinátája azt jelenti, hogy mekkora a valószínűsége annak, hogy a t időpillanatban az i . csúcson vagyunk. Ekkor a $\mathbf{p}(t)$ egy valószínűségi eloszlást definiál a gráf csúcsai felett. Ezek alapján a $\mathbf{p}(t + 1) = M\mathbf{p}(t)$, ahol az M a fentebb definiált sztochasztikus átmenetmátrix. Tegyük fel, hogy $\mathbf{p}(t + 1) = M\mathbf{p}(t) = \mathbf{p}(t)$. Ekkor $\mathbf{p}(t)$ a stacionárius eloszlás a véletlen séta felett, és teljesül az eredeti $\mathbf{r} = M\mathbf{r}$ egyenlet.

Ezzel az algoritmussal két fő probléma adódhat:

1. Ha alkalmazzuk a véletlen sétát egy olyan gráfra, amiben létezik egy olyan pont, aminek csak bemenő élei vannak, de kimenő nincsen, akkor sosem fogja majd $\mathbf{p}$ a szavazatát továbbosztani, így a szavazatok elvesznek.
2. A másik probléma akkor lép fel, ha a gráfban van egy olyan kör, amiből nincsen kivezető út. Amikor a véletlen séta során belépünk a körbe, akkor beragadunk, és körbe-körbe fogunk menni. Ez azt jelenti, hogy ezek a pontok folyamatosan egymásnak fogják adni a szavazataikat, amit attól a ponttól „szívnek el”, ami kapcsolódik a kör valamely pontjához, és sokkal nagyobb PageRank értékeket érnek el, mint amekkora a valódi fontosságuk.

Ennek a két problémának a feloldására vezették be a teleportálást. A séta során minden időpillanatban β valószínűséggel fogunk a következő pontra átmenni a ki-éleken keresztül, és

$1 - \beta$ valószínűséggel egy véletlenszerű pontra ugrunk a gráfban. A β értékét a gyakorlati alkalmazásokban jellemzően 0,8 és 0,9 közé szokás állítani.

A PageRank egyenletet módosítani kell, hogy a teleportálásokat is figyelembe vegyük. Így a végleges alakja:

$$r_j = \sum_{i \rightarrow j} \beta \frac{r_i}{d_i} + (1 - \beta) \frac{1}{N} \quad (2.5)$$

Ahol d_i jelenti az i pont ki-fokát, N a csúcsok száma a gráfban, és feltételezzük, hogy a gráfot olyan alakra hoztuk, hogy abban nincsenek olyan pontok, amelyeknek 0 a ki-fokuk, vagy ha mégis vannak, akkor ezekre a pontokra a teleportálás valószínűsége 1.

2.1.2. Perszonalizált PageRank

Láttuk, hogy a PageRank algoritmus a gráf minden pontjához egy fontossági értéket rendel, amelyet a pont PageRank értékének nevezünk. Ebben a dolgozatban ennek mi az egyik módosított változatát fogjuk használni, melyet Perszonalizált PageRank algoritmusnak nevezünk. A módosítás annyiban tér el az előző megvalósítástól, hogy definiáljuk a $\emptyset \neq T \subseteq V$ halmazt, melyet **topik**nak nevezünk. A megvalósítás annyiban tér el az előzőektől, hogy a teleportálás után nem egy véletlenszerű pontra ugrunk a gráfban $p = 1/|V|$ eséllyel, hanem valamely $v \in T$ pontra ugrunk véletlenszerűen $p = 1/|T|$ valószínűséggel. Ezáltal a pontokhoz rendelt PageRank értékekből meghatározhatjuk, hogy melyek a leghasonlóbbak a T topik elemeihez.

A módosított mátrix alakban felírt egyenlet a következőképpen néz ki:

$$A = \beta M + (1 - \beta) \left[\frac{1}{N} \right]_{n \times n} \quad (2.6)$$

Ez ekvivalens a fenti egyenlettel. Az $\mathbf{r} = A\mathbf{r}$ alapján az $\mathbf{r} = \beta M\mathbf{r} + \left[\frac{1-\beta}{N} \right]_N$ alakra hozható az egyenlet. Ahhoz, hogy ez kompatibilis legyen a Perszonalizált PageRank-kel, változtatnunk kell az A mátrixon. A perszonalizált esetben:

$$A_{ij} = \begin{cases} \beta M_{ij} + \frac{1-\beta}{|T|}, & \text{ha } i \in T \\ \beta M_{ij}, & \text{különben} \end{cases} \quad (2.7)$$

2.2. Maximum entrópia osztályozás

Az általunk definiált bináris osztályozási feladat megoldásához egy maximum entrópia alapú osztályozót használtunk, amelyre szokás logisztikus regresszióként is hivatkozni [2]. Munkánk során a Mallet [6] gépi tanuló programcsomag implementációját használtuk.

A maximum entrópia modellezés előnye, hogy az – szemben például a Naïve Bayes osztályozóval – nem tételezi fel a jellemzők egymástól vett, célváltozón kondicionált feltételes függetlenségét. Ez az osztályozó olyan módon képes az egyes jellemzőkhöz tartozó modellparamétereket (súlyokat) meghatározni, hogy azok használata mellett a tanító adatbázisban lévő mintákhoz tartozó osztálycímkéknek a (megfigyelhető jellemzőkön) kondicionált valószínűségét (vagy annak logaritmusát) maximalizálja.

A használt modell ebből adódóan a természetesnyelv-feldolgozásban és a gépi tanulásban egyaránt széleskörűen használt diszkriminatív megközelítés osztályozási problémák megoldására. Egy alternatív modellezési lehetőséget a generatív modellek családja biztosított volna, melyek a tanító adatbázisban lévő megfigyelés-, és a célváltozók együttes valószínűségének, $P(\mathbf{x}, y)$ -nek maximalizását tűzik ki céljukul a modell paramétereinek meghatározása során. Ez a megközelítés azonban az előbbieken említett, túlzó feltételezésekkel él a jellemzők értékeinek viselkedésére nézve, melyet mi a diszkriminatív modell használatával el tudtunk kerülni.

3. fejezet

A jellemzőtér bemutatása

A maximum entrópia osztályozáshoz az egyedeket jellemzőkkel kell leírnunk, amelyek valamilyen tulajdonság szerint értéket rendelnek hozzájuk, amit a tanulási folyamat során használunk fel arra, hogy kategóriákba soroljuk azokat.

Ebben a fejezetben a saját munkámat írom le, mely a jellemzőcsoportok részletes leírása. A fejezet végén lesz definiálva, hogy hogyan fogjuk ezeket felhasználni a konkrét tanulási folyamat során.

3.1. A jellemzőcsoportok leírása

Mivel a feladatot visszavezettük osztályozási problémára, ahol a és b felhasználókból álló (a, b) példányokról kell eldönteni a $barátság(a, b)$ reláció értékét, ezért a jellemzőteret úgy alakítjuk ki, hogy megállapítjuk többféle szemszögből az a és b felhasználó hasonlóságát. Ehhez definiáljuk, hogy mit értünk hasonlóság alatt:

3.1. Definíció. Legyen $G = (V, E)$ egy irányított, vagy irányítatlan gráf, ahol $v_1, v_2 \in V$. Azt mondjuk, hogy v_2 pont v_1 ponthoz képesti **Perszonalizált PageRank szerinti értéke** (PPR-értéke) s , és $s = PPR(G, v_1, v_2)$ -vel jelöljük, ahol s egy valós szám, ha a G gráfon a $T = \{v_1\}$ topikkal indított perszonalizált PageRank algoritmus a v_2 csúchoz az s valós számot rendeli.

A PageRank tulajdonságaiból az következik, hogy ha a $T = \{v\}, v \in V$ topikkal indítunk egy Perszonalizált PageRank algoritmust, akkor az algoritmus szerint azon $v_i \in V$ elem lesz a leghasonlóbb v -hez, amelynek a PPR-értéke a legkevésbé tér el tőle, azaz azon v_i , melyre $|PPR(G, v, v) - PPR(G, v, v_i)|$ érték minimális.

3.2. Definíció. A G a v_2 pont v_1 ponthoz képesti **relatív PPR-hasonlóságán** a v_1 ponthoz a perszonalizált PageRank szerint v_2 -nél kevésbé hasonló pontok számát, és az összes pont számosságának a hányadosát értjük, azaz

$$\frac{|\{i | abs(PPR(G, v_1, v_2) - PPR(G, v_1, v_i)) < abs(PPR(G, v_1, v_1) - PPR(G, v_1, v_2)) \wedge v_i \in V\}|}{|V|} \quad (3.1)$$

A hasonlóságon ezen túl mindig relatív hasonlóságot értünk. A hasonlóság nem kommutatív: ha veszünk két pontot: v_1 és v_2 -t, és kiszámítjuk a relatív hasonlóságukat, akkor a v_1 pont v_2 -höz képesti relatív hasonlósága általában nem egyezik meg a v_2 pont v_1 -hez képesti relatív hasonlóságával. Ez az állítás ugyanúgy igaz a PPR-hasonlóságra.

A barátok kereséséhez három fő jellemzőcsoportot állapítottunk meg. Az első a felhasználók nyelvezetének a hasonlósága, második a felhasználók étteremlátogatási preferenciái, a harmadik pedig egy kapcsolati gráfon alapuló hasonlóság megállapítása. A továbbiakban ezt a három jellemzőcsoportot fogjuk sorra venni. Mind a három jellemzőnél az értékek meghatározásához a felhasználók PPR-hasonlóságát fogjuk figyelembe venni különböző gráfokban.

3.1.1. A felhasználók nyelvezete

Ezt a jellemzőt azért hoztuk létre, mert úgy gondoljuk, hogy egy felhasználónak a nyelvhasználatból következtethetünk arra, hogy bizonyos egyéneknek mi az érdeklődési körük, ebből pedig azt szeretnénk megállapítani, hogy mennyiben befolyásolja a barátságokat a közös érdeklődési kör.

Ennek a jellemzőnek az alapja a szöveges vélemény halmaz: ahogy az 1-es fejezetben az adathalmazok leírásánál láttuk, a vélemény fájlokban található meg, hogy melyik felhasználó, melyik étteremről milyen véleményt ír. Így minden rekord összekapcsol egy éttermet, egy felhasználót, és egy véleményt. Itt most csak a vélemény, és a felhasználó kapcsolatot fogjuk vizsgálni. Mivel a vélemények sima szöveges formátumban vannak, így előfeldolgozást kell rajtuk végeznünk azért, hogy információt nyerhessünk ki belőlük.

Először az összes vélemény összes szavának megállapítjuk a szófaját, amelyhez a Stanford NLP POS tagger csomagját [11] használjuk. A szófaji kódok megállapítása után olyan fájlt kapunk, ahol a véleményekben a szavak után egy alsóvonallal összekapcsolva szerepel a szó szófajának betűkódja. Néhány gyakori szófaji kód, és a jelentésük:

Szófaji kód	Szófaj
NN, NNS, NNP, NNPS	Főnevek: egyes szám, többes szám, tulajdonnév egyes számban, tulajdonnév többes számban
JJ, JJR, JJS	Melléknévek: alap-, közép-, és felsőfok.
VB, VBD	Ige jelen idő, múlt idő

Egy szófaj annotációval ellátott példamondat:

```
I_PRP love_VBP this_DT place_NN !_. I_PRP have_VBP
worked_VBN in_IN the_DT area_NN for_IN 5_CD
years_NNS ,_, and_CC these_DT folks_NNS are_VBP
probably_RB the_DT best_JJS quality_NN lunch_NN you_PRP
```

can_MD find_VB around_RB here_RB ._. Fresh_JJ ingredients_NNS
 ,_, friendly_JJ staff_NN ,_, rather_RB quick_JJ service_NN
 ._. The_DT Greek_JJ salad_NN with_IN gyro_NN meat_NN
 -LRB-_-LRB- #_# 41_CD -RRB-_-RRB- is_VBZ so_RB delicious_JJ
 ,_, I_PRP have_VBP n't_RB swayed_VBN from_IN it_PRP in_IN
 three_CD years_NNS ._. The_DT dining_NN room_NN is_VBZ
 always_RB clean_JJ ._. I_PRP highly_RB suggest_VBP lunch_NN
 here_RB !_.

Ha szófajokat rendelünk a szavakhoz, akkor jóval több információ áll rendelkezésre róluk. Például egy szót kétféle képen is kezelhetünk: ha valaki leírta azt, hogy „vár”, értelmezhetjük igeiként is egy olyan mondatban, hogy „Egészen pontosan 20 perce vár az ételre.”, de akár abban a mondatban is lehet, hogy „Egy felettébb romantikus vacsorát költöttem el a vár egyik királyi lakosztályában berendezett középkori stílusú étteremben.” viszont itt már főnévként szerepel.

Minket ebben a feladatban viszont csak a főnevek érdekelnek, mert azt szeretnénk kideríteni, hogy milyen témákról írnak az egyes felhasználók. Mivel a szavakat külön-külön fogjuk vizsgálni, így az unigram modellünkben (ahol nem vizsgálunk szókapcsolatokat, csak a szavakat önmagukban vizsgáljuk) nem vesszük figyelembe, hogy valaki a kedvenc étterméhez milyen jelzőt kapcsol, csak azt, hogy milyen étteremről, vagy milyen ételekről ír. Önmagában abból, hogy két felhasználó leírta azt a szót, hogy „kedvenc”, „jól”, „rossz”, nem tudnánk növelni a tudásbázisunkat, így egyértelmű, hogy például a melléknevekkel ebben a formában nem tudunk mit kezdeni, csak zajosabb lenne tőle a modellünk. Így a példamondatunkból megmaradó szavak a következők lesznek:

lunch_NN quality_NN folks_NNS years_NNS area_NN place_NN

A harmadik lépésben az a feladatunk, hogy kiszűrjük azokat a szavakat, amik nem bírnak számunkra semmilyen hasznos jelentéssel. Ezen szavak listája a stopszógyűjteményben [1] található meg. A szűrést végrehajtjuk a szótövesítés előtt, és a szótövesítés után is. Ezek után egy irányítatlan páros gráfot építünk, ami azt reprezentálja, hogy melyik felhasználó melyik szót írta le:

A szófajok megállapítása után jön a szótövesítés. Ez azért hasznos, mert sok esetben ugyanaz a szó több szóalakban fordul elő. A szótövesítő feladata, hogy a ragokat levágja a szóról, és így a szótövek alapján több különböző formában, de ugyanarra vonatkozó szavakat össze tudjuk vonni. Ilyen lehet például az, ha van két mondatunk: „Nagyon ízlenek az ilyen olaszosan fűszerezett ételek. . .”, és a „Az olasz konyha teljes repertoárját véltem felfedezni. . .” akkor a szótövesítés nélkül két külön dolgot értenénk az „olasz”, és az „olaszosan” szó alatt, pedig „érezzük”, hogy ez a két mondat hasonló jelentéstartalommal bírhat. Viszont a szótövesítés után azt kapjuk, hogy az „olaszosan” -ból le lett vágva a toldalék, így „olasz” lett, a második

mondatban az „olasz”-t pedig érintetlenül lett hagyva, így a megérzésünknek helyt adva a két szót összevonjuk, és a két mondatot leíró felhasználóról esetleg azt a következtetést vonhatjuk le, hogy mind a ketten érdeklődnek az olasz konyha iránt.

3.3. Definíció (felhasználó-szó gráf). *Legyen $G = (\{V_U \cup V_W\}, E)$ egy irányítatlan páros gráf, ahol minden $u_i \in U$ felhasználóhoz hozzárendelünk egy $v_{u_i} \in V_U$, és minden w feldolgozott szóhoz egy $v_{w_j} \in V_W$ szögponthoz. A $(v_{u_i}, v_{w_j}) \in E$ akkor, és csak akkor teljesül, ha a v_{u_i} ponthoz tartozó felhasználó legalább egyszer leírta azt a szót, amelyet az előfeldolgozás során v_{w_j} szóra alakítottunk.*

Később látni fogjuk, hogy a felhasználók hasonlóságát nem befolyásolják azok a csúcsok, amik csak egy éllel rendelkeznek, így azokat el lehet távolítani a gráfból annak érdekében, hogy a jellemzőt kiszámító algoritmusunk sebességét növeljük.

3.1.2. Az étteremlátogatási preferenciák megállapítása

Az étteremlátogatási szokások megállapítása azért tűnhet jó ötletnek, mert azt gondolhatjuk, hogy az, hogy a felhasználók mennyire hasonló éttermeket értékelnek, hatással lehet arra, hogy kialakítanak-e barátságot a valóságban. Például ha a , és b felhasználó írt véleményt a Gundelről, a' , és b' pedig a helyi kifőzdét értékelte, akkor következtethetünk arra, hogy a felhasználónak közelebb áll az ízlése b -hez, mint a' -höz, hiszen a két étterem teljesen más célközönséget vonz. Ebből azt a következtetést vonhatjuk le, hogy más társadalmi csoporthoz tartoznak, mindenestre az étterem látogatási szokásaik merőben különböznek egymástól, így nem lenne jó ötlet felajánlani barátoknak őket, hiszen nem tudnának egymással hasznos információkat megosztani.

Ennek a jellemzőnek a kiszámolásához a vélemények halmazából indulunk ki. A véleményekben ugyanis a fent említett módon van összekapcsolva egy felhasználó egy étteremmel. Ha egy felhasználó írt egy véleményt egy étteremről, akkor nagy valószínűséggel járt is ott, de legalábbis érdeklődik iránta. Ez az információ elég ahhoz, hogy egy újabb páros gráfot építsünk fel a felhasználókból és az éttermekből a következők szerint:

3.4. Definíció (Felhasználó-szavak gráf). *Legyen $G = (\{V_U \cup V_R\}, E)$ egy irányítatlan páros gráf, $u_i \in U$ egy felhasználó, $r_j \in R$ egy étterem, ahol U a felhasználók, R pedig az éttermek halmaza; $v_{u_i} \in V_U$ az u_i felhasználót, a v_{r_j} pedig az r_j éttermet reprezentáló pont a gráfban. Ekkor a $(v_{u_i}, v_{r_j}) \in E$ akkor és csak akkor teljesül, ha az u_i felhasználó írt egy véleményt az r_j étteremről.*

3.1.3. A kapcsolati háló felhasználása

A kapcsolati háló alatt egy olyan hálózatot értünk, ahol két felhasználót akkor kapcsolunk össze, ha az egyik barátja a másiknak. Ebből következtetni tudunk arra, hogy milyen kapcsolatokkal rendelkeznek az egyes felhasználók.

A kapcsolati hálót azért gondoljuk fontosnak megvizsgálni, mert okunk van feltételezni, – ahogy a valóságban is megfigyelhető – hogy az ismerősünk ismerősével előbb fogunk barátkozni, mint teljesen idegen emberekkel.

3.5. Definíció (kapcsolati vagy felhasználó-felhasználó gráf). Legyen $G = (V, E)$ egy irányított gráf, ahol minden $v_i \in V$ szögponthoz egy $u_i \in U$ felhasználót rendelünk (azaz a gráf egy pontja egy felhasználót reprezentál), és $(v_1, v_2) \in E$ irányított él akkor és csak akkor teljesül, ha a v_2 ponthoz tartozó u_2 felhasználó barátja a v_1 szögponthoz tartozó u_1 felhasználónak.

A kapcsolati gráf felépítéséhez a felhasználók adataiból álló adathalmazt használjuk. A *user* fájlban minden felhasználóhoz megkapjuk az összes barátjának a felhasználói azonosítóját, így a gráfot könnyen előállíthatjuk. A kapcsolati gráfból a tanításkor kivettük azokat az éleket, amik azokat a barátságokat reprezentálják, amik a példák között vannak, ugyanis ebben az esetben olyan információhoz jutna az algoritmus, amit éppen meg kell határoznia.

3.2. A jellemzők előállítása a tanuláshoz

Eddig leírtuk, hogy mi alapján számoljuk ki az egyes jellemzőket. Itt pontosan azt fogjuk definiálni, hogy milyen jellemzővektorokat fogunk a tanítási folyamat során felhasználni egy (a, b) példányhoz.

A fejezetben előzőleg definiált három jellemzőcsoportokhoz konkrét jellemzőket állapítunk meg, mindegyikhez pontosan hatot. Az alábbi folyamat érvényes mind a három jellemzőre; így összesen 18 jellemzővektort generálunk a három jellemzőcsoportból.

Az első két jellemző során kiszámoljuk az a felhasználóhoz rendelt v_a pont, és a b felhasználóhoz rendelt v_b ponttal a Perszonalizált PageRank vektort, úgy, hogy v_a -t választjuk topiknak, majd ezt megcsináljuk visszafelé is: a v_b topikkal számoljuk ki a vektort. Legyen ez a két vektor $\mathbf{pr}_a$, és $\mathbf{pr}_b$, értelemszerűen. A PageRank vektorok a gráf pontjai felett egy eloszlást definiálnak. A jellemzők során a felhasználókhoz rendelt pontokon kívül más pontok is szerepelhetnek a gráfban (például a felhasználó-szavak gráfban szavak, és felhasználók is vannak, és mindegyik pontnak van PageRank értéke), ezért először normalizálni kell a két vektort, hogy eloszlást kapjunk. Ezek után kiszámoljuk a $D_{KL}(\mathbf{pr}_a || \mathbf{pr}_b)$ és a $D_{KL}(\mathbf{pr}_b || \mathbf{pr}_a)$ értékét. Az elsőből lesz az első jellemző érték, míg utóbbiból a második:

$$\text{jellemző}_1 = D_{KL}(\mathbf{pr}_a || \mathbf{pr}_b) \quad (3.2)$$

$$\text{jellemző}_2 = D_{KL}(\mathbf{pr}_b || \mathbf{pr}_a) \quad (3.3)$$

$$D_{KL}(\mathbf{pr}_a || \mathbf{pr}_b) = \sum_i \mathbf{pr}_a(i) \log \frac{\mathbf{pr}_a(i)}{\mathbf{pr}_b(i)} \quad (3.4)$$

A harmadik, és az ötödik jellemző érték a b felhasználóhoz rendelt v_b pont, és az a felhasználóhoz rendelt v_a pont relatív PPR-hasonlósága valamelyik gráfon. Ezt a fenti képlettől eltérően úgy számoljuk ki, hogy sorba rendezzük az elemet a PageRank értékeik szerint, és megállapítjuk a sorrendben v_a , és v_b helyét, majd kiszámoljuk a következő értéket, ami a jellemző értéke lesz:

$$\text{jellemző}_{\{3,5\}} = 1 - \frac{\text{rang}(v_b) - \text{rang}(v_a)}{|U|} \quad (3.5)$$

Ha a a , és b egyre közelebb kerül egymáshoz a rangsorban, akkor a tört számlálója tart az 1-hez. A nevezőben az összes felhasználó van, így a tört egy ϵ számhoz tart. Így az egész kifejezés értéke (így a jellemző értéke) $1 - \epsilon \approx 1$ -hez tart. Ellenkező esetben, minél távolabb vannak egymástól, a tört egy $1 - \epsilon \approx 1$ számhoz tart, így a jellemző értéke ≈ 0 lesz.

A harmadik, és ötödik jellemzőt ugyanúgy számoljuk ki, a különbség, hogy az 5. esetben felcseréljük a -t, és b -t mindkét oldalról megvizsgálva a hasonlóságot.

A negyedik, és hatodik jellemző azt veszi figyelembe, ha a v_a pontnak minél kisebb a PageRank értéke. Ugyanis alacsonyabb PageRank esetén a hasonlósági reláció kevésbé megbízható. Vegyük megint az elemek sorba rendezését PageRank érték szerint, majd számoljuk ki a jellemző értékét:

$$\text{jellemző}_{\{4,6\}} \text{ val} = 1 - \frac{\text{rang}(v_a)}{|U|} \quad (3.6)$$

4. fejezet

Eredmények

A következőkben a modellünk kiértékelésére használt metrikákat, és az azok által elért eredményeinket mutatjuk be.

4.1. Metrikák

A kiértékelést úgy végezzük el, hogy veszünk egy (a, b) példányt, aminek tudjuk az osztálycímkejét: ez a **tényleges** osztálycímke. Ezek után a gépi tanuló algoritmussal megállapítjuk a példányra a címkét úgy, hogy a tanítóhalmazba nem vesszük fel (a, b) -t (így az osztályozó számára ismeretlen lesz a címke) ez a **predikált** osztálycímke. A cél nyilván az, hogy a predikált osztálycímke megegyezzen a tényleges osztálycímkével. Ekkor **igaz** (true, jól kategorizált) példányról beszélünk; ha nem egyezik meg, akkor pedig **hamis** (false, rosszul kategorizált) példányról. Mind a kettőből létezik pozitív, és negatív.¹

Az összefüggéseket a következő táblázat foglalja össze:

Értékeljük ki az algoritmust néhány példányon, majd képezzük a következő példányokat tartalmazó halmazokat a fenti táblázat alapján:

- TP - true positive, azaz a pozitív példák, amelyet helyesen pozitívnak kategorizált az algoritmus
- FP - false positive, a pozitívnak kategorizált, de valójában negatív példák
- TN - true negative, a helyesen negatív címkéjűnek predikált negatív példák
- FN - false negative, negatívnak predikált, de valójában pozitív példák

Ebből állíthatjuk elő a tévesztési mátrixot, mely a 4.1 táblázatban van összefoglalva.

¹ Nem tévesztendő össze a nem példány a barátság reláció értékével: ha van egy tanítópéldánk (a, b) , melyet a *nem* címkével látunk el (tehát *bartsg*(a, b) nem teljesül), és az algoritmus helyesen *nem* címkével látja el, akkor ez egy TN , azaz true negative példa

		Predikált		Összesen
		Pozitív	Negatív	
Tényleges	Pozitív	$ TP $	$ FN $	$ TP + FN $
	Negatív	$ FP $	$ TN $	$ FP + TN $

4.1. táblázat: Tévesztési mátrix

A **pontosság** (precision) jellemezzük, hogy a tanulás futtatása után milyen arányban vannak a ténylegesen pozitívnak kategorizált példák az összes igaznak predikált példák számosságához képest. A kiszámítása a következőképpen történik:

$$precision = \frac{|TP|}{|TP| + |FP|} \quad (4.1)$$

A **fedés** (recall) azt jelenti, hogy mekkora a jól kategorizált pozitív példányok aránya a pozitív példákhoz, és a rosszul kategorizált negatív példányokhoz képest, melyet a következőképpen kapunk meg:

$$recall = \frac{|TP|}{|TP| + |FN|} \quad (4.2)$$

A pontosság, és fedés harmonikus közepéből képezzük az **F₁ pontszámot**, amely az algoritmusunk hatékonyságának a számszerűsítése ezen két összetevő alapján:

$$F_1 = 2 * \frac{precision * recall}{precision + recall} \quad (4.3)$$

Ezen kívül még használjuk a **helyességet** (accuracy) is, amely a jól kategorizált példányok aránya a jól kategorizált, és a rosszul kategorizált példák számosságának összegéhez:

$$Accuracy = \frac{|TP| + |TN|}{|TP| + |TN| + |FP| + |FN|} \quad (4.4)$$

Mivel itt egy olyan feladattal állunk szemben, ahol csak a pozitív, illetve címkézetlen példányok állnak rendelkezésre az adathalmazban, így csak az igaz pozitív példákra tudunk „biztosat” mondani. A negatív példányoknál a tényleges osztálycímét önkényesen választjuk ki véletlenszerűen.

A tanítást 1000 db „igen”, és 1000 db „nem” címkével ellátott (a, b) példán hajtottuk végre. A kiértékelés során a példahalmazt 10 részre osztottuk, majd keresztvalidációt használva vettük a kiértékelések számtani közepét. A tanítást 14 különböző jellemző kombináció esetén végeztük el, hogy mérni tudjuk az egyes jellemzők hatását, és az eredmények a következőképpen alakultak a jellemzőcsoportok kombinációk függvényében:

	Pozitív osztály			
	Pontosság	Precízió	Fedés	F_1
B_1	0,500	0,500	1,0	0,667
B_2	0,500	0,500	0,500	0,500

4.2. táblázat: A különböző baseline rendszerek eredményei

4.2. A baseline

A kiértékeléshez megállapítottunk két **baseline**-t. Baseline alatt egy olyan képzeletbeli algoritmus által elért eredményt értünk, amit mindenképpen túl szeretnénk szárnyalni.

Mivel csak a pozitív példákról tudunk biztosat mondani, hiszen csak azok vannak definiálva a kiindulási adathalmazban, így a baseline-okat is ehhez igazítjuk.

A B_1 baseline egy olyan algoritmus által elért eredmény, ami egyszerűen azt csinálja, hogy minden ismeretlen példát az *igaz* címkével lát el.

Ekkor az algoritmus minden olyan példát, amelynek valójában igaz a címkéje, jól fog kategorizálni, és az összes negatív példát is eltéveszti.

A B_2 baseline legyen egy olyan algoritmus, ami véletlenszerűen kategorizálja a példányokat. Ebben az esetben $1/2$ valószínűséggel kategorizálja helyesen a pozitív, és a negatív példákat is. Tehát a TP , és TN példák aránya is $1/2$. Ebből triviálisan következik, hogy a rosszul kategorizált negatív, és pozitív példányok aránya is $1/2$ lesz.

Megjegyzendő, hogy ezeknek a baseline-oknak nem vagy csupán csekély mértékű gyakorlati hasznosságuk van, hiszen ha ezeket használnánk fel, akkor algoritmusunk az összes lehetséges felhasználót felkínálná barátnak (B_1) vagy pedig minden információt nélkülözve, teljesen uniform módon választana a további felhasználók közül (B_2). A B_1 , és B_2 baseline által elért eredményeket a 4.2 táblázat foglalja össze.

4.3. Az algoritmus eredményei

Az eredményének méréséhez először teszteljük külön-külön az egyes jellemzőcsoportokat úgy, hogy használjuk a KL-divergenciát (4.5 táblázat), illetve úgy is, hogy nem (4.4 táblázat).

Rev	Rest	Soc	Pozitív osztály			
			Pontosság	Precízió	Fedés	F_1
•	•		0,793	0,782	0,809	0,795
•		•	0,929	0,910	0,952	0,930
	•	•	0,933	0,919	0,952	0,935
•	•	•	0,933	0,917	0,951	0,933

4.3. táblázat: KL divergenciát nem használó eredmények, az egyes jellemzőcsoportok összevont vizsgálatával

	Pozitív osztály			
	Pontosság	Precízió	Fedés	F_1
Review	0,727	0,720	0,745	0,731
Restaurant	0,772	0,747	0,821	0,781
Social	0,929	0,907	0,953	0,929

4.4. táblázat: KL divergenciát nem használó eredmények

	Pozitív osztály			
	Pontosság	Precízió	Fedés	F_1
Review	0,737	0,732	0,750	0,739
Restaurant	0,780	0,755	0,830	0,790
Social	0,935	0,919	0,952	0,935

4.5. táblázat: KL divergenciát használó eredmények

A táblázatokból leolvasható, hogy a KL divergencia megállapítása majdnem minden esetben hozzásegít ahhoz, hogy jobb eredményt érjen el az algoritmus. Főleg akkor javít az eredményeken, ha az F_1 pontszám alacsonyabb. Mikor mind a három jellemzőt bekapcsoljuk, akkor már némi visszaesés figyelhető meg.

A jellemzőcsoportok külön-külön vizsgálatából megállapítható, hogy a várakozásoknak megfelelően a kapcsolati háló járul hozzá leginkább ahhoz, hogy jó eredményeket érjünk el, viszont a cél az lenne, hogy ezen jellemző kihagyásával is meg tudjuk közelíteni az elért eredményeket. A további munkáknál ezeket a lehetőségeket vesszük számba.

4.4. A gráfok statisztikái

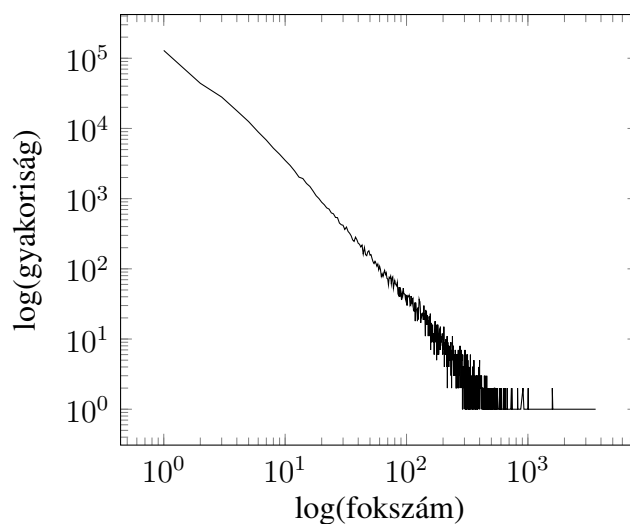
A gráfokon statisztikai elemzést végeztünk. Ebből világosan látszik, hogy a felhasználó-szavak gráf éleinek száma egy nagyságrenddel nagyobb, mint a másik két gráfé. A szociális háló igen ritka, megfigyelhető, hogy érvényesül bennük a skála invariáns hálózatokra jellemző hatvány-eloszlásos fokszámeloszlás, azaz sok csúcsonak van kevés éle, a legtöbb él kialakításában a csúcsoknak egy szűk részhalmaza vesz részt. Ez az, ami indokolja azt, hogy a grafikonoknál log-log skálát alkalmazunk.

Rev	Rest	Soc	Pozitív osztály			
			Pontosság	Precízió	Fedés	F_1
•	•		0,807	0,802	0,817	0,809
•		•	0,932	0,918	0,947	0,932
	•	•	0,934	0,924	0,945	0,934
•	•	•	0,932	0,921	0,945	0,933

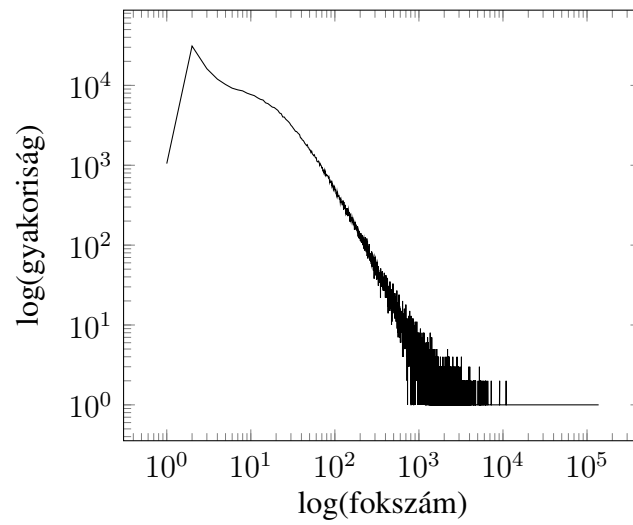
4.6. táblázat: KL divergenciára támaszkodó eredmények, az egyes jellemzőcsoportok összevont vizsgálatával

	Fokszám				csúcsok	összél (millió)
	min	max	átlag	szórás		
Felhasználó-szavak	0	137469	98,839	936,34	350603	34,653
Felhasználó-éttermek	1	3615	7,387	34,334	294615	2,176
Szociális háló	0	2917	7,560	40,687	252898	1,912

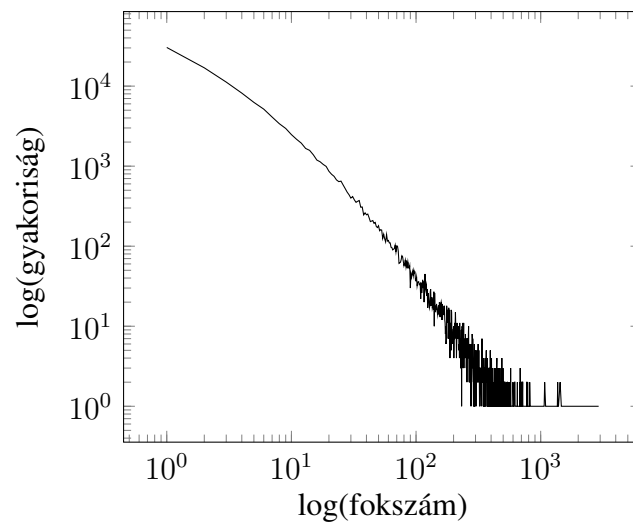
4.7. táblázat: A különféle gráfok deskriptív jellemzése



4.1. ábra. A felhasználó-étterem gráf fokszámeloszlása



4.2. ábra. A felhasználó-szó gráf fokszámeloszlása



4.3. ábra. A felhasználó-felhasználó gráf fokszámeloszlása

5. fejezet

Kapcsolódó munkák

A szociális kapcsolatok mibenlétét többen is vizsgálták előttünk.

A [5] cikkben a szerzők egy gráfot definiálnak, ami lényegében megegyezik a mi kapcsolat gráfunkkal, viszont az élek nem csak a barátsági kapcsolatokból származhatnak, hanem akármi-lyen más interakció nyomán, ezen kívül pedig két pont között több él is mehet, amivel többféle kapcsolatot reprezentálhatunk. Az a célkitűzésük, hogy ezeket az interakciókat jelentő éleket tudják előrejelezni az szociális gráf időbeli megváltozása során, és ebben a témakörben vizsgálódnak. Ehhez azt az egyébként kevésbé megkérdőjelezhető tényt veszik alapul, hogy a szociális gráf időről időre változik, így a t időpillanatban a szociális gráf részgráfja a t' időpillanathoz tartozó gráfnak, ahol $t < t'$. A t időpillanathoz tartozó gráfot $G[t]$ -vel jelölik. Ezt **kapcsolat predikciós problémának** nevezik el a szerzők. A tanítást, és kiértékelést úgy végzik el, hogy választanak négy időpillanatot, amelyet két intervallumra osztanak fel: $t_0 < t'_0 < t_1 < t'_1$. Egy olyan algoritmus, mely képes ezt a problémát megoldani a $G[t_0, t'_0]$ gráfon, azaz a G gráfon a t_0 -tól a t'_0 időpillanatig történt változásokat tanulja meg, majd a t_1 -től a t'_1 időpillanatig kialakuló éleket (és csomópontokat) próbálja meg minél pontosabban előrejelezni. Ezt részletesebben úgy teszik meg, hogy (x, y) párokhoz definiálnak egy értéket, ami a két pont hasonlóságát méri a gráfban, és úgy jelölik, hogy $\text{score}(x, y)$. A $\text{score}(x, y)$ értékét többféle módon számolják ki, és összevetik az egyes módszerek által elért eredményeket.

A [3] cikkben olyan a szerzők a kapcsolat predikciót oldják meg, és ehhez három külön módszert mutatnak be: az elsőben jellemzőket nyernek ki a csomópontokról, majd egy bináris osztályozó segítségével végeznek predikciót két csomóponton. Ehhez megvizsgálják különböző metrikák hatékonyságát, ahogy az előzőleg idézett cikkben is, mint: legrövidebb út hossza két pont között, Katz (a legrövidebb út egyik változata, amely hatékonyabb szociális gráf esetén), Perszonalizált PageRank, SimRank. A második módszer szerint a csomópontok halmazát vizsgálják valószínűségi modellekkel, majd az éleknek az együttes eloszlását predikálják a Bayes statisztika segítségével. A harmadik megközelítés, amit bemutatnak, a lineáris algebrán, és az optimalizáláson alapul, mégpedig úgy, hogy közvetlenül a szociális kapcsolatokat leíró gráf

szomszédsági mátrixán dolgoznak. Legyen A az a mátrix, amely a tanulási folyamathoz tartozó gráf szomszédsági mátrixra, a B mátrix pedig a kiértékeléshez használt gráf mátrixa. Ekkor a feladat, hogy valamilyen optimalizálási módszer segítségével egy olyan f függvényt tanuljunk meg, amely az A mátrixot a B -re transzformálja.

A [9] cikkben a szerzők a Last.fm, és a Fickr portálról kinyert adathalmazon dolgoznak. A Last.fm a lejátszási listák alapján ajánl fel a felhasználóknak barátokat. A szerzők rámutatnak, hogy jobb eredményt lehet elérni a felhasználók által a zenékhez rendelt címkék vizsgálatával, illetve a felhasználók csoportokba tartozásának meghatározásával.

A [4] cikkben a szerzők ugyanúgy a Yelp-en dolgoznak, a véleményeket pedig a nyelvészeti oldalukról közelítik meg. Az a hipotézisük, hogy egy vélemények nem csak az a szerepe, hogy egy éttermet jó, illetve rossz színben tüntessen fel. Ehelyett a véleményt író személy pszichológiai aspektusait, és szociális helyzetét is figyelembe veszik. Azt teszik fel, hogy különböző típusú elbeszélések az éttermekről mást, és mást jelentenek különböző szociális helyzetben lévő embereknek, és ez befolyásolja azt is, hogy milyen stílusban írnak jó, vagy rossz véleményt egy drága, vagy éppen egy olcsó étteremről. A hipotézisüket automatikus módszerekkel tesztelik a portál véleményeiből felépülő korpuszon.

6. fejezet

Összegzés

Ugyan az elkészült algoritmus az eredmények tekintetében hatékonynak bizonyul abban, hogy barátságokat tudjunk vele keresni, viszont számos lehetőségünk van arra, hogy a felhasznált technikák továbbfejlesztésével még jobb eredményeket érhessünk el, illetve a szociális hálót kihagyjuk a jellemzők közül, így lehetővé tesszük, hogy az algoritmusunkat szélesebb körben is fel lehessen használni.

Az egyik irány, amerre elindulhatunk, hogy a problémákat átfogalmazzuk, és egy rangsorolási feladatként tekintünk rá a továbbiakban. A későbbiekben nem csak annyit mondhatnánk egy felhasználó párról, hogy őket barátoknak gondoljuk-e vagy sem, de választhatunk több felhasználót, és rangsoroljuk őket, és a felhasználónak a legvalószínűbbeket kínálunk fel.

A felhasználók nyelvezetének hasonlóságának megállapításához további eszközöket is bevetethetünk annak érdekében, hogy jobb eredményeket érjünk el: használhatunk taxonómia, illetve ontológiai adatbázist. Például ha az egyik véleményben megtaláljuk a *spagettit*, a másikban pedig a *pizzát*, akkor a taxonómia alapján ez a két étel az olaszos ételek alatt lenne a hierarchiában, míg ha másik két felhasználónak a véleményében megtaláljuk a *chilit*, illetve a *tacost*, ezeket össze tudnánk foglalni úgy, hogy mexikói ételek. Ebből arra következtethetünk, hogy – bár két külön szót írtak le – mégis hasonló ízlésük lehet az egyéneknek.

Itt még további finomítási lehetőség lenne, ha új szófajcsoportokat is elemeznénk, és nem csak szavakkal dolgoznánk a feldolgozás során, hanem kifejezéseket vizsgálnánk. Ha ezen kívül szemantikai vizsgálatot is végzünk a szavakon, akkor több esélyünk lehet a megfelelő kategorizálásra. Például az egyik felhasználó leírja, hogy „hot soup”, a másik pedig, hogy „hot chili”, akkor a „hot” jelentésének megállapítása után azt találjuk, hogy az egyik felhasználó azt értette rajta, hogy „forró”, a másik esetben pedig azt, hogy „erős”. Így ezt a két esetet külön kezelhetnénk az azonos szóalak ellenére. További lehetőség lenne új jellemzők hozzáadása.

Az adathalmazban lehetőségünk van kinyerni további adatokat, például azt, hogy a felhasználók milyenek értékelték egy bizonyos éttermet. Ezt felhasználhatjuk úgy, hogy klaszterezzük az éttermeket: ha két felhasználó ugyanabba a klaszterbe tartozó éttermekre hasonló pontszá-

mot ad, akkor gondolhatjuk azt, hogy ennek a felhasználó párnak hasonló az ízlése.

Érdekes lehet a felhasználókat klaszterezni, majd az egy klaszterbe tartozó felhasználók közül indulhatnánk ki a barátok kereséséhez. A jelenleg elkészített jellemzőinket ennek függvényében értékelnénk ki: felhasználhatjuk azt az információt, hogy hogyan aránylik a klaszter mérete a felhasználók közötti hasonlósághoz. Például egy kis klaszterben ugyanaz az eltérés egészen más jelentőséggel bír, mint egy nagy klaszterben.

Köszönetnyilvánítás

6.1. Köszönetnyilvánítás

Köszönetet mondok a témavezetőimnek, Berend Gábornak, és Dr. Farkas Richárdnak, hogy bizalmukkal, útmutatásukkal, és segítőkészségükkel hozzájárultak ahhoz, hogy ez a dolgozat létrejöhessen.

A kutatást a FuturICT.hu nevű, TÁMOP-4.2.2.C-11/1/KONV-2012-0013 azonosítószerű projekt támogatta az Európai Unió és az Európai Szociális Alap társfinanszírozása mellett.

Irodalomjegyzék

- [1] Gábor Berend. Machine learning-based extraction of keyphrases and its applications in multiple domains.
- [2] Adam L. Berger, Vincent J. Della Pietra, and Stephen A. Della Pietra. A maximum entropy approach to natural language processing. *Comput. Linguist.*, 22(1):39–71, March 1996.
- [3] MohammadAl Hasan and MohammedJ. Zaki. A survey of link prediction in social networks. In Charu C. Aggarwal, editor, *Social Network Data Analytics*, pages 243–275. Springer US, 2011.
- [4] Dan Jurafsky, Victor Chahuneau, Bryan Routledge, and Noah Smith. Narrative framing of consumer sentiment in online restaurant reviews. *First Monday*, 19(4), 2014.
- [5] David Liben-Nowell and Jon Kleinberg. The link prediction problem for social networks. In *Proceedings of the Twelfth International Conference on Information and Knowledge Management, CIKM '03*, pages 556–559, New York, NY, USA, 2003. ACM.
- [6] Andrew Kachites McCallum. MALLET: A machine learning for language toolkit. <http://mallet.cs.umass.edu>, 2002.
- [7] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank citation ranking: Bringing order to the web., November 1999. Previous number = SIDL-WP-1999-0120.
- [8] Anand Rajaraman and Jeffrey David Ullman. *Mining of Massive Datasets*. Cambridge University Press, New York, NY, USA, 2011.
- [9] Rossano Schifanella, Alain Barrat, Ciro Cattuto, Benjamin Markines, and Filippo Menczer. Folks in folksonomies: Social link prediction from shared metadata. *CoRR*, abs/1003.2281, 2010.
- [10] Csentes Tibor. Közelítő és szimbolikus számítások. Typotex, 2007.

- [11] Kristina Toutanova and Christopher D. Manning. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Proceedings of the 2000 Joint SIGDAT conference on Empirical methods in natural language processing and very large corpora*, EMNLP '00, pages 63–70, Stroudsburg, PA, USA, 2000. ACL.