UNIVERZITA PAVLA JOZEFA ŠAFÁRIKA V KOŠICIACH

PRÍRODOVEDECKÁ FAKULTA

RELAČNÁ KLASIFIKÁCIA

DIPLOMOVÁ PRÁCA

Študijný program: Informatika

Katedra: Ústav informatiky

Školiteľ: RNDr. Tomáš Horváth PhD.

Košice, 2014

Bc. Tomáš Jakab

Abstrakt

Objekty vo svete neexistujú osamotené len tak kdesi vo vzduchoprázdne – napríklad užívatelia sociálnych sietí vytvárajú medzi sebou prepojenia, vzťahy a relácie tým, že zdieľajú spoločné odkazy, páčia sa im rovnaké stránky, prispievajú k rovnakým článkom a pod. Práve existenciu podobných prepojení využíva relačná klasifikácia. Úlohou relačnej klasifikácie je predikcia triedy (bude sa danému užívateľovi páčiť odkaz XY?) neohodnotených objektov v grafe s prihliadnutím na triedy (alebo iné vlastnosti) uzlov v jeho okolí a váh hrán, ktorými sú vzájomne prepojení. V našej práci sa sčasti venujeme štúdiu existujúcich riešení – ako je iteratívna klasifikácia, ktorá postupne, vo vlnách, ohodnocuje uzly, až kým neohodnotí všetky, alebo metóda využívajúca logistickú regresiu na naučenie závislostí medzi okolím uzla a jeho ohodnotením. Druhou, hlavnou časťou je potom návrh a implementácia vlastného algoritmu pre relačnú klasifikáciu. Jeho základnou myšlienkou je nájdenie takých centier v grafe (najlepších klebetníc), ktoré majú na svoje okolie najväčší vplyv. Tie potom propagujú do siete svoje „gossipy“ (klebety), ktoré ovplyvnia výslednú klasifikáciu neohodnotených uzlov. V práci sa tak venujeme skúmaniu rôznych možností ako tieto klebety propagovať a aké informácie by mali so sebou niesť, aby sme dosiahli čo najúspešnejšiu klasifikáciu.

Kľúčové slová: relačná klasifikácia, collective inference, ghost edge

Abstract

Objects in the world doesn´t exist separated somewhere in a vacuum – e.g. social network users create connections and relations between them by sharing links, liking same pages, contributing to the same article, etc. Exactly these relations uses relational classification. The aim of relational classification is class prediction (will user like reference XY?) of unknown nodes in the graph by the class (but also other properties) of nodes in its neighborhood and the weights of edges, by which they are linked to each other. First of all we studied existing solutions of relational classification – iterative classification, which gradually in waves (iterations), classify nodes until all of them are classified. Another method using a logistic regression to learn dependencies between node´s neighborhood and its classifications. The second, main part of our work is design and implementation of new algorithm for relational classification. Its basic idea is to find such centers in the graph, which have the greatest impact to their neighborhood. Then they propagate into nodes in graph their gossips, which will affect the final classification of unknown nodes. So we explore different ways how to find these centers, how to propagate gossips in graph and what information they should carry with them to achieve the most successful classification.

Keywords: relational classification, collective inference, ghost edge

OBSAH

1 ÚVOD 7

2 ATRIBÚTOVO ORIENTOVANÁ KLASIFIKÁCIA 8

2.1 Atribútové klasifikátory 8

3 RELAČNE ORIENTOVANÁ KLASIFIKÁCIA 10

3.1 Priame relačné klasifikátory 12

3.1.1 Weighted-Vote Relational Neighbor Classifier 12

3.1.2 Class-Distribution Relational Neighbor Classifier 13

3.1.3 Network-only Bayes Classifier 14

3.2 Collective Inference 14

3.2.1 Gibbs Sampling 15

3.2.2 Relaxation Labeling 16

3.2.3 Iterative Classification 18

3.3 Klasifikácia riedko ohodnotených grafov 19

3.3.1 Ghost Edges 19

3.3.1.1 Relačné klasifikátory Ghost Edges 21

3.3.1.2 Ghost EdgeNL 21

3.3.1.3 Ghost EdgeL 22

4 NÁVRH VLASTNEJ METÓDY RELAČNEJ KLASIFIKÁCIE 23

4.1 Motivácia 24

4.2 Prvotný návrh 25

4.3 Analýza algoritmu 27

4.3.1 Určenie váhy ohodnotených uzlov 27

4.3.1.1 Funkcie pre určenie váhy ohodnotených uzlov 32

4.3.2 Určenie váhy gossipu 38

4.3.2.1 Riešenia využívajúce váhu zdrojového uzla gossipu 39

4.3.2.2 Riešenia využívajúce pravdepodobnosť 41

4.3.3 Odoslanie a agregácia gossipov 43

4.3.4 Pseudokód Gossip Algoritmu 44

5 VYHODNOTENIE 45

5.1 Datasety 45

5.1.1 Dataset IMDB 45

5.1.2 Dataset Industry 46

5.1.3 Dataset CORA 46

5.1.4 Dataset Wisconsin 47

5.2 Metodológia 48

5.3 Výsledky 49

6 ZÁVER 54

7 LITERATÚRA 56

8 PRÍLOHY 57

A 57

1 Úvod

Klasifikácia je proces priraďovania tried jednotlivým objektom z reálneho prostredia. Takýchto klasifikácií, alebo tiež triedení, je okolo nás nespočetne veľa. Klasifikácia sa využíva v mnohých vedeckých disciplínach a už dávno pred postavením prvého počítača vznikli veľké klasifikačné systémy (napríklad medzinárodná klasifikácia chorôb). Tieto klasifikačné systémy vytváral človek manuálne – podrobným štúdiom každej jednej inštancie, jej atribútov a vlastností a následným vytvorením príslušnej hierarchie. Vytvorenie takejto klasifikácie vyžadovalo nemalú časovú investíciu mnoho expertov z danej vedeckej disciplíny. Rozmachom informačných technológii od polovice 20. storočia vzniká i nový odbor – dolovanie dát. Dolovanie dát resp. dolovanie v dátach prináša nové metódy ako strojovo a automatizovane hľadať v dátach nové zákonitosti, pravidlá. Vznikajú metódy špecializované na rôzne druhy dát od jednoduchých, triviálnych, mechanických problémov, ako napríklad triedenie materiálu a automat na lístky, až po najzložitejšie, vyžadujúce rozsiahly expertný systém ako lekárske expertné systémy na určenie diagnózy pacienta, ktoré v sebe dokážu skĺbiť encyklopedické vedomosti a poznatky z praxe lekára, prípadne i viacerých lekárov – expertov na danú oblasť.

Pre naše účely môžeme klasifikáciu triediť na dve základné skupiny – atribútová klasifikácia a relačná klasifikácia.

Atribútovo riadená klasifikácia je proces zatrieďovania objektov do tried na základe ich atribútov. Pod atribútmi objektov rozumieme ich vlastnosti, ktoré sú pre zatrieďovanie v danej problematike určujúce. Napríklad pri zatrieďovaní elektronických dokumentov sú určujúce atribúty názov, autor, citácie a samotný text dokumentu. Pri zatrieďovaní je možné niektorým atribútom prideliť váhy. Napríklad pri dokumentoch môže mať nadpis väčšiu váhu ako samotný text dokumentu. Podstatou atribútovo riadených klasifikácií je naučenie klasifikátora zatrieďovať inštancie do tried na základe hodnôt ich atribútov.

Existujú však klasifikačné úlohy pri ktorých je okolie klasifikovanej inštancie dôležitejšie ako samotné hodnoty jej atribútov. V takomto prípade hovoríme o relačne orientovanej klasifikácii. Medzi relačne orientované klasifikácie patrí napríklad klasifikácia webových stránok (klasifikácia webovej stránky je presnejšia, pokiaľ je obohatená o informácie o stránkach, na ktoré sa na skúmanej stránke nachádzajú odkazy) a klasifikácia pripojených osôb do telefónnej siete (snaha o odhalenie zločincov na základe predpokladu, že zločinec uskutočňuje telefónne hovory s iným zločincom), kde navyše ide o typický príklad riedko

ohodnotenej siete. Platí tu totižto, že o veľa ľuďoch na začiatku nevieme povedať nič. Nemáme žiadny dôkaz o ich nezákonnej činnosti, ale taktiež nevieme s istotou povedať, že ju nepáchajú. Napriek tomu sme schopní sa zo začiatočného stavu, keď o niekoľkých ľuďoch vieme s určitosťou povedať, že sú zločinci a máme vedomosť o uskutočnených hovoroch, dopracovať k odhaleniu ďalších zločincov.

Automatizované zatrieďovanie v relačnej klasifikácii vyžaduje teda odlišný prístup ako v atribútovo orientovanej klasifikácii. Napríklad v epidémiou postihnutej oblasti by bolo možné určiť nakazené osoby len na základe ich atribútov, ktoré popisujú zdravotný stav daného človeka. Ak by sme ale do klasifikátora dokázali zahrnúť i ďalšie fakty, ktoré popisujú danú situáciu (napríklad kontakt nakazeného s inými osobami), získali by sme ešte presnejšiu predikciu a odhad šírenia epidémie do budúcnosti.

Treba tiež povedať, že relácie možno transformovať i na atribúty inštancie. Napríklad už spomínaný účastník v telefónnej sieti by mohol mať *n* atribútov, ktoré predstavujú *n* iných účastníkov, s ktorými boli uskutočnené hovory. Takýto prístup však spôsobí, že inštancie nadobudnú rozličný počet atribútov. To ale znemožňuje atribútovo orientovanú klasifikáciu, pretože prístupy v atribútovo orientovanej klasifikácii vyžadujú rovnaký počet atribútov. Transformáciou nedosiahneme teda žiadne zjednodušenie, práve naopak.

2 Atribútovo orientovaná klasifikácia

Ako už bolo spomenuté, atribútovo orientovaná klasifikácia je proces zatrieďovania objektov do tried na základe ich atribútov. Toto sa docieli naučením klasifikátora zatrieďovať objekty do tried, pričom ide zväčša o takzvané učenie s učiteľom. Pri učení s učiteľom vytvárame klasifikačný model tak, že ho na tréningovej množine naučíme rozoznávať jednotlivé triedy na základe hodnôt atribútov objektov. Snažíme sa teda o zachytenie atribútov tréningovej množiny tak, aby čo najlepšie reprezentovali a determinovali triedy objektov.

2.1 Atribútové klasifikátory

Tieto sú najznámejšie atribútovo orientované prístupy :

- metóda na základe Bayesovho teorému – vypočítava podmienené pravdepodobnosti vyjadrujúce závislosť jednotlivých atribútov od tried.
- Bayesove siete – sú založené na princípe Bayesovho teorému, umožňujú modelovať podmienené závislosti medzi atribútmi inštancie.
- neurónové siete – sú založené na princípe biologických neurónov ľudského mozgu. Neurónová sieť sa skladá z viacerých vrstiev a neuróny každej vrstvy sú prepojené s neurónmi nasledujúcej vrstvy. Každé spojenie má určenú váhu. Klasifikátor sa učí upravovaním týchto váh až kým sa dosiahne akceptovateľný výsledok.
- Support Vector Machines – spočíva vo vytvorení stavového priestoru objektov, v ktorom sa vytvoria roviny, ktoré oddeľujú objekty rôznych tried. Trieda nového, nezatriedeného objektu sa potom určí na základe príslušnosti k jednej z rovín.
- rozhodovacie stromy – predstavujú spôsob usudzovania založený na princípe „ak platí X, potom platí Y“. V každom uzle rozhodovacieho stromu sa vyhodnotí podmienka vzťahujúca sa k niektorému z atribútov objektu. Vyhodnotenie následne determinuje smer k ďalšiemu uzlu a takto vzniká cesta, ktorou sa algoritmus dopracuje od koreňa stromu až k jednému z listov stromu. Listy stromu určujú výslednú klasifikáciu objektu.
- *k*-najbližších susedov – tento prístup na začiatok vytvorí *n*-rozmerný priestor, pričom *n* je počet atribútov objektov. Do tohto priestoru vloží všetky objekty z tréningovej množiny. Následne pri klasifikácii neznámeho objektu sa tento vloží do priestoru, nájde sa *k*-najbližších susedov ktorých klasifikácia je známa a na základe ich tried sa určí trieda i pre nepoznaný objekt. Môže sa použiť rôzna metrika, no najčastejšie sa používa euklidovská vzdialenosť. Táto metóda je zaujímavá tým, že tu neprebehne klasická fáza s trénovaním klasifikátora.

Pre vymenované prístupy je charakteristická nezávislosť jednotlivých atribútov, teda neexistuje žiadny determinujúci vzťah medzi atribútmi objektov. Rovnako tak existencia jedného objektu nevypovedá nič o existencii či neexistencii iných objektov. Dôležitá je tiež podmienka rovnakého počtu atribútov pre každý objekt.

Ešte spomenieme, že existujú aj prístupy, ktoré medzi atribúty jedného objektu priradia i atribúty iných objektov. V takomto prípade hovoríme o virtuálnych dokumentoch a daný spôsob sa využil napríklad pri klasifikácii webových stránok, kde k textu pôvodnej stránky bol pridaný i text stránky, na ktorú existoval hypertextový odkaz. Existuje viacero spôsobov

ako začleniť cudzí atribút medzi pôvodné - cudzí atribút môže mať rovnakú, alebo inú váhu ako pôvodné atribúty.

U virtuálnych dokumentoch už teda badať akúsi snahu o zachytenie tej skutočnosti, že objekty neexistujú osamotené, ale práve naopak, s inými objektmi interagujú a existujú medzi nimi vzťahy, ktoré majú vplyv na ich klasifikáciu. Virtuálne dokumenty túto aditívnu informáciu využívajú, no samotná klasifikácia ostáva stále atribútovo orientovaná.

3 Relačne orientovaná klasifikácia

Kým v atribútovo orientovanej klasifikácii sa na objekty hľadí ako na samostatne existujúce entity, ktoré nemajú žiadnu vedomosť o existencii iných objektov a tobôž o možných vzťahoch, reláciách medzi nimi, tak v relačne orientovanej klasifikácii je toto presne naopak. Objekty v rámci relačne orientovanej klasifikácie si treba predstaviť ako graf, kde je každý objekt reprezentovaný jedným uzlom a hrana medzi dvoma uzlami hovorí o existencii vzťahu medzi nimi. Dôležité je, že príslušnosť k triede sa v takomto grafe šíri pozdĺž existujúcich hrán a teda fakt, že objekt *A* patrí do triedy *1*, má vplyv na klasifikáciu všetkých jeho susedov. Dá sa predpokladať, že takýmto spôsobom ovplyvní každý objekt v najväčšej miere svojich priamych susedov a čím je vzdialenosť medzi dvoma objektmi väčšia, tým menší bude ich vzájomný vplyv.

Pre relačnú klasifikáciu sú charakteristické nasledovné črty:

- prepojenosť dát – medzi objektmi sú prítomné väzby, vzťahy, relácie
- within-network classication – klasifikácia neohodnotených objektov v grafe je založená na distribúcii informácie o triede známych objektov v rámci existujúcich prepojení – hrán
- node-centric classification – v danom čase pracujú metódy s jedným objektom, jeho atribútmi a okolím a takto postupne prechádzajú celým grafom

V ďalšom pokračovaní budeme používať nasledovné označenia:

Na vstupe je daný graf $G = (V, E, X)$, kde V je množina vrcholov, E je množina hrán, pričom $E \subseteq [V]^2$, a X_i je atribút vrcholu $v_i \in V$.

Ďalej x^K označíme vektor tried pre vrcholy $v \in V^K$, kde V^K je množina takých vrcholov, ktorých klasifikácia je na začiatku známa. Obdobne x^U označíme vektor tried pre vrcholy $v \in V^U$, kde V^U je množina takých vrcholov, ktorých klasifikácia je na začiatku neznáma. Potom $G^K = (V, E, x^K)$, predstavuje všetky informácie, ktoré sú nám dopredu známe – poznáme všetky vrcholy a hrany a tiež ohodnotenie niektorých z vrcholov. Hrana $e_{ij} \in E$ predstavuje hranu medzi vrcholmi v_i a v_j pričom uvažujeme neorientované hrany a w_{ij} predstavuje váhu danej hrany. Uvažujeme neorientované hrany, preto $e_{ij} = e_{ji}$ a tiež $w_{ij} = w_{ji}$.

$P(x_i = c|\mathcal{N}_i)$ značí pravdepodobnosť pridelenia triedy c pre i-ty objekt (uzol v grafe) na základe klasifikácie susedov i-teho objektu.

Ukázalo sa, že kompletný relačne orientovaný klasifikačný systém je vhodné rozdeliť na 3 základné časti:

- nerelačný („lokálny“) model – pozostáva z modelu, ktorý pri učení využíva len lokálne informácie objektov, t.j. len hodnoty atribútov týchto objektov. Takto sa vytvorí akési prvotné, základné ohodnotenie pre objekty, ktorých príslušnosť do tried nepoznáme a máme ju predikovať. Táto prvotná klasifikácia nemusí byť ešte presná, presnosť sa optimalizuje až v ďalších krokoch. V prvom kroku je teda podstatné dosiahnuť aspoň približnú klasifikáciu a nenechať žiadny objekt neklasifikovaný.
- relačný model – cieľom druhého kroku je tiež klasifikácia objektov, avšak tu sa už využíva znalosť o existencii relácií medzi objektmi. Skrz tieto prepojenia sa tak šíria jednak samotné príslušnosti objektov k triede, ale aj hodnoty atribútov objektov. Informácie od jedného objektu ovplyvnia klasifikáciu nie len bezprostredných susedov, ale potenciálne aj nepriamych susedov cez niekoľko hrán.
- collective inferencing – posledná tretia časť je zameraná práve na šírenie informácie od jedného objektu cez niekoľko hrán aj k vzdialenejším objektom. V skutočnosti to nie je nová klasifikačná metóda, nevytvára sa tu už model ako v predchádzajúcich dvoch krokoch. Collective inferencing berie výsledky z druhého kroku len ako čiastkové a tieto rôznymi spôsobmi iteratívne distribuuje do ďalších susedných uzlov, až kým sa situácia v grafe neustáli a dosiahne sa čo najviac vyvážený stav priradenia tried k objektom.

Jednotlivé prístupy v každom z troch bodov môžu byť vďaka takémuto rozdeleniu osobitne skúmané a navzájom porovnávané. Rozdelenie ďalej otvára priestor na experimentovanie – do každej z častí môže byť nezávisle od ostatných dosadená nová metóda, nový prístup, ktorý môže priniesť lepšie výsledky. Dokonca sa objavili i pokusy o systematické generovanie a skúmanie všetkých kombinácií.

O budovaní nerelačných modelov už bolo povedané v rámci atribútovo orientovanej klasifikácii, teraz priblížime druhú a tretiu časť uvedenej schémy.

3.1 Priame relačné klasifikátory

Úlohou relačných klasifikátorov je určiť pravdepodobnosť príslušnosti daného objektu k niektorej z tried. Ako už bolo niekoľkokrát spomenuté, relačné klasifikátory na určenie príslušnosti k triede využívajú znalosť o triedach okolitých susedov v grafe. Všetky nasledovné relačné klasifikátory majú spoločné, že vyžadujú znalosť len o triedach najbližších susedov. Tu treba spomenúť mechanizmus nazývaný homofília.

Homofília hovorí, že objekty, ktoré sú navzájom prepojené, sú si podobné s vyššou pravdepodobnosťou, ako objekty, ktoré prepojené nie sú. Tento úkaz sa vyskytuje najmä v grafoch zachytávajúcich sociálne väzby v spoločenstve ľudí a práve vďaka nemu má význam hovoriť o tom, že príslušnosť objektu k niektorej z tried ovplyvňuje príslušnosť k triedam aj jeho priamych a skrz nich aj nepriamych. Typickými väzbami v spoločenstve ľudí, ktoré nesú v sebe črty homofílie, sú napríklad vek, rasa, národnosť, náboženstvo a geografická poloha.

Ešte spomenieme, že objekty, ktorých trieda nie je známa, sa buď ignorujú, alebo im je určená jedna z tried v závislosti od výberu nerelačného klasifikátora v prvom kroku relačne orientovaného klasifikačného systému.

3.1.1 Weighted-Vote Relational Neighbor Classifier

Metóda WVRN predikuje $P(x_i|\mathcal{N}_i)$pre objekt $v_i \in V^U$ako vážený priemer klasifikovaných tried objektov z $\mathcal{N}_i$nasledovne:

$$P(x_i = c|\mathcal{N}_i) = \frac{1}{Z} \sum_{v_j \in \mathcal{N}_i} w_{i,j}.P\left(x_j = c\middle|\mathcal{N}_j\right)$$

kde Z je normalizačná konštanta. Je to jeden z najjednoduchších relačných klasifikátorov. Teda pravdepodobnosť toho, že i-ty uzol bude klasifikovaný do triedy c je rovná súčtu súčinov príslušných váh s pravdepodobnosťou toho, že aj susedný uzol je klasifikovaný do tej istej triedy c. Tento výsledok je ešte normovaný konštantou Z.

3.1.2 Class-Distribution Relational Neighbor Classifier

Metóda CDRN spresňuje metódu WVRN o zohľadnenie lokálnej distribúcie tried v susedstve vrcholu. Za týmto účelom sa zavádza vektor tried vrcholu $CV\ (v_i)$ pre každý vrchol v_i a referenčný vektor triedy $RV\ (c)$ pre každú triedu c. Zadané sú nasledovne:

$$CV\ (v_i)_k = \sum_{v_j \in \mathcal{N}_i, x_j = c_k} w_{i,j}$$

kde $CV\ (v_i)_k$ predstavuje k-tu pozíciu v danom vektore tried a c_k je k-ta trieda. Teda na k-tej pozícii vektora tried pre objekt v_i je súčet váh spojení daného objektu s takými susedmi, ktorí sú klasifikovaní do triedy k.

$$RV\ (c) = \frac{1}{|V_c^K|} \sum_{v_j \in V_c^K} CV\ (v_i)$$

kde $|V_c^K|$je počet objektov, ktorých klasifikácia je od začiatku známa a sú klasifikované do triedy c. Referenčný vektor pre triedu c je teda súčet vektorov tried takých vrcholov, ktoré sú klasifikované do triedy c a ich klasifikácia je od začiatku známa, normovaný počtom objektov klasifikovaných do triedy c.

Pre daný objekt $v_i \in V^U$ CDRN metóda určí teda pravdepodobnosť $P(x_i = c|\mathcal{N}_i)$ za pomoci vektora tried vrcholu a referenčného vektora triedy nasledovne:

$$P(x_i = c|\mathcal{N}_i) = sim(CV\ (v_i), RV\ (c))$$

kde $sim(a, b)$ je ľubovoľná funkcia podobnosti vektorov (L_1, L_2, *cosine similarity, atď.*), normovaná do intervalu [0,1]. Táto metóda teda porovnáva pre daný vrchol a triedu distribúciu triedy v jeho okolí voči globálnej distribúcii triedy v celom grafe. Víťazná trieda je pre vrchol tá, ktorej lokálna distribúcia sa najviac zhoduje s globálnou.

3.1.3 Network-only Bayes Classifier

NBC metóda využíva na výpočet pravdepodobnosti príslušnosti objektu v_i do triedy c Bayesov teorém:

$$P(x_i = c|\mathcal{N}_i) = \frac{P(\mathcal{N}_i|c).P(c)}{P(\mathcal{N}_i)}$$

kde$P(c)$ je pomer počtu vrcholov, ktoré patria do triedy c voči všetkým vrcholom, $P(\mathcal{N}_i) = \frac{1}{|V|}$ a podmienená pravdepodobnosť $P(\mathcal{N}_i|c)$ je určená nasledovne:

$$P(\mathcal{N}_i|c) = \frac{1}{Z}\prod_{v_j \in \mathcal{N}_i} P\left(x_j = \tilde{x_j}|x_i = c\right)^{w_{ij}}$$

kde Z je normalizačná konštanta, $\tilde{x_j}$ je trieda j-teho objektu.

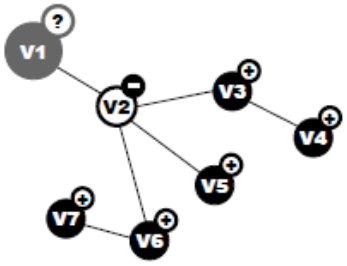
Uvedené priame relačné klasifikátory spája ich jednoduchosť a priamočiarosť. Hodnoty atribútov objektov v nich nehrajú žiadnu rolu a pravdepodobnosť príslušnosti objektu k niektorej z tried je určená výhradne podľa klasifikácie okolitých susedov v grafe. Môžeme si tiež všimnúť, že nejde o klasické učenie s učiteľom, pretože dané metódy nemajú tréningovú a testovaciu fázu.

Ako vidíme, priame relačné klasifikátory dokážu distribuovať informáciu o klasifikácii objektu k jeho susedom. Vrchol však vidí iba na svojich priamych susedov (hovoríme, že zdieľanie zodpovedá Markovskému reťazcu rádu 1). V nasledujúcej časti uvedieme niekoľko metód, ktoré túto informáciu dokážu distribuovať i do vzdialených uzlov cez niekoľko hrán.

3.2 Collective Inference

Už bolo spomenuté, že CI nepredstavuje ďalší klasifikátor tak, ako tomu bolo v prípade atribútových a priamych relačných klasifikátorov. Táto posledná, tretia časť relačne orientovaného klasifikačného systému sa stará o prepojenie výsledkov z použitých atribútových a relačných klasifikátorov.

Spomenuli sme tiež, že CI dokáže distribuovať informácie v grafe aj cez viac hrán medzi uzlami, ktoré nie sú priamymi susedmi. Predstavme si napríklad nasledujúcu situáciu:



OBR. 1

V grafe sú vrcholy klasifikované v dvoch triedach {+,-}. Aplikovaním samotnej metódy priamej relačnej klasifikácie nadobudne vrchol V1 vždy klasifikáciu -. Správnosť takejto klasifikácie je ale otázna a vynárajú sa dve otázky. Prvá, či je vrchol V2 správne ohodnotený a druhá, ak aj je vrchol V2 správne ohodnotený, vidíme veľký nepomer medzi počtom vrcholov v jednotlivých triedach a preto sa môžeme domnievať, že nový, neklasifikovaný uzol by mal byť ohodnotený s prihliadnutím na tento nepomer. Problémom tu je homofília (tá vystupuje v rôznej miere a v rôznej forme v závislosti od charakteru dát) a nízky stupeň vlastného susedstva.

Spomeňme, že existuje i iný spôsob ako sa s problémom prenesenia informácie na vzdialené uzly vysporiadať – riešením by bolo, ak by už relačný klasifikátor nazeral na triedy vzdialenejších susedov.

Podstatou metód CI je iteratívne aplikovanie niektorého z relačných klasifikátorov, až kým sa šírenie informácií o príslušnosti k triedam neustáli. Jednotlivé metódy sa líšia v spôsobe, akým sa dosiahnuté čiastkové výsledky z jednotlivých iterácií spájajú dokopy a tiež v iteračných podmienkach.

V nasledujúcej časti predstavíme 3 metódy patriace do CI a to Gibbs Sampling, Relaxation Labeling, Iterative Classification.

3.2.1 Gibbs Sampling

GS je často používanou metódou v CI. Výhodou je jeho priamočiarosť, jasnosť, nevýhodou vopred dané, relatívne vysoké konštanty počtu iterácií, pričom k ustálenému stavu v grafe dôjde oveľa skôr.

Algoritmus Gibbs Sampling

1. Inicializácia $\forall v_i \in V^U$ použitím atribútového modelu $\mathcal{M}_L$ nasledovne:

a. $\hat{c}_i \leftarrow \mathcal{M}_L(v_i)$, kde $\hat{c}_i$ je vektor pravdepodobností $P(x_i|\mathcal{N}_i)$ vypočítaných modelom $\mathcal{M}_L$. Zápisom $\hat{c}_i(k)$ sa rozumie k-ta hodnota vektora $\hat{c}_i$, ktorá predstavuje $P(x_i = c_k|\mathcal{N}_i)$.
b. výber hodnoty c_s z vektora $\hat{c}_i$ tak, že $P(c_s = c_k|\hat{c}_i) = \hat{c}_i(k)$.
c. dosadenie $x_i \leftarrow c_s$.

2. Vygenerovanie náhodného usporiadania O z uzlov patriacich do V^U
3. Pre každý uzol $v_i \in O$ v poradí:
   a. využitím relačného klasifikátora: $\hat{c}_i \leftarrow \mathcal{M}_R(v_i)$
   b. výber hodnoty c_s z vektora $\hat{c}_i$tak, že $P(c_s = c_k|\hat{c}_i) = \hat{c}_i(k)$
   c. dosadenie $x_i \leftarrow c_s$.

   pričom pri dosadení $\hat{c}_i \leftarrow \mathcal{M}_R(v_i)$v čase t uzlyv_i pre$i \in \{1, \dots, i-1\}$ obsahujú už nové ohodnotenie z iterácie v čase t a uzly v_i pre $i \in \{i+1, \dots, n\}$ obsahujú ohodnotenie z iterácie v čase $t-1$.
4. Opakovanie kroku 3. po dobu 200 iterácií, pričom sa nezaznamenáva žiadna štatistika.
5. Opakovanie kroku 3. po dobu 2000 iterácií s vedením štatistiky pre každý uzol o tom, koľkokrát mu je ktorá trieda priradená. Znormalizovaním dostaneme výsledné pravdepodobnosti klasifikácie do jednotlivých tried.

Pri metóde GS platí, že v každej jednej chvíli behu algoritmu sú všetky uzly ohodnotené a priradené do niektorej z tried. Relačný klasifikátor tak vždy vidí kompletne ohodnotené okolie.

3.2.2 Relaxation Labeling

Na rozdiel od GS, metóda RL neponúka pre uzly daného grafu G klasifikáciu v každom jednom bode algoritmu, namiesto toho zachováva nekompletnosť v klasifikácii uzlov a vedie sa záznam o triedach, ktoré sú pridelené uzlom v každej jednej iterácii. Zmenené je i dosadenie novej klasifikácie pre uzly. V RL sa nové ohodnotenia dosadzujú až po vykonaní klasifikácie pre všetky uzly v danej iterácii, na rozdiel od GS, kde po výpočte novej klasifikácie sa okamžite zmenilo i ohodnotenie uzla.

Algoritmus Relaxation Labeling

1. Inicializácia $\forall v_i \in V^U$, použitím atribútového modelu $\mathcal{M}_L$ nasledovne:

a. $\widehat{c_i}^{(0)} \leftarrow \mathcal{M}_L(v_i)$, kde $\widehat{c_i}^{(0)}$ je vektor pravdepodobností $P(x_i|\mathcal{N}_i)$ vypočítaných modelom $\mathcal{M}_L$ v 0-tej iterácii. Zápisom $\widehat{c_i}(k)$ sa rozumie k-ta hodnota vektora $\widehat{c_i}$, ktorá predstavuje $P(x_i = c_k|\mathcal{N}_i)$.

b. výber hodnoty c_s z vektora $\widehat{c_i}$ tak, že $P(c_s = c_k|\widehat{c_i}) = \widehat{c_i}(k)$.

c. dosadenie $x_i \leftarrow c_s$.

2. Pre $\forall v_i \in V^U$:

   a. výpočet hodnoty x_i aplikovaním relačného modelu:

$$\widehat{c_i}^{(t+1)} \leftarrow \mathcal{M}_R({v_i}^t)$$

   kde$\mathcal{M}_R({v_i}^t)$ vychádza pri výpočte z vektora $\widehat{c_j}^{(t)}$ pre $v_j \in \mathcal{N}_i$ a t je poradie iterácie. Tým je dosiahnutý efekt pseudosimultánneho nastavenia nových ohodnotení relačného klasifikátora.

3. Opakovanie kroku 2. po dobu T. Vektor $\widehat{c_i}^{(T)}$ v sebe obsahuje výsledné pravdepodobnosti klasifikácie tried pre uzol v_i.

Občas sa vyskytne situácia, kedy RL nekonverguje ku konečnému riešeniu a začne oscilovať medzi niekoľkými ohodnoteniami grafu. Tomuto javu je možné predísť postupným znižovaním vplyvu susedných uzlov pri určení triedy uzla nasledovne:

V bode 2. nahradíme existujúce priradenie $\widehat{c_i}^{(t+1)} \leftarrow \mathcal{M}_R({v_i}^t)$priradením:

$$\widehat{c_i}^{(t+1)} \leftarrow \beta^{(t+1)}.\mathcal{M}_R\left({v_i}^{(t)}\right) + (1 - \beta^{(t+1)}).\widehat{c_i}^{(t)}$$

kde

$$\beta^0 = k$$

$$\beta^{(t+1)} = \beta^{(t)}.\alpha$$

kde k je konštanta z intervalu $[0,1]$ a α je konštanta predstavujúca rýchlosť rastu útlmu, ideálne z rozsahu $0{,}9 < \alpha < 1$. Tým pádom hodnota β s pribúdajúcim počtom iterácií klesá. V takto upravenom algoritme je volením konštánt k a α možné určiť váhy, ktorými zložky $\mathcal{M}_R({v_i}^t)$ a $\widehat{c_i}^{(t)}$ prispievajú k výpočtu vektora $\widehat{c_i}^{(t+1)}$. Nevhodné nastavenie konštánt môže spôsobiť, že susedné uzly stratia svoj vplyv skrz relačný klasifikátor príliš skoro, ešte pred dokonvergovaním k ideálnemu riešeniu. Opačný extrém je už spomínaná oscilácia medzi niekoľkými stavmi grafu.

3.2.3 Iterative Classification

Z algoritmu IC vyplýva, že môže nastať situácia, kedy po skončení algoritmu niektoré uzly $v_i \in V^U$ zostanú bez priradenej triedy. To môže byť výhodné, ak je pre nás dôležitá presnosť klasifikácie a uprednostníme situáciu, kedy klasifikátor radšej žiadnu triedu nepriradí, akoby ju priradil so zvýšenou chybou.

Algoritmus Iterative Classification

1. Inicializácia $\forall v_i \in V^U$ použitím atribútového modelu $\mathcal{M}_L$ nasledovne:
   a. $\hat{c}_i \leftarrow \mathcal{M}_L(v_i)$, kde $\hat{c}_i$ je vektor pravdepodobností $P(x_i|\mathcal{N}_i)$ vypočítaných modelom $\mathcal{M}_L$. Zápisom $\hat{c}_i(k)$ sa rozumie k-ta hodnota vektora $\hat{c}_i$, ktorá predstavuje $P(x_i = c_k|\mathcal{N}_i)$.
   b. výber hodnoty c_s z vektora $\hat{c}_i$ tak, že $P(c_s = c_k|\hat{c}_i) = \hat{c}_i(k)$.
   c. dosadenie $x_i \leftarrow c_s$.
2. Vygenerovanie náhodného usporiadania O z uzlov patriacich do V^U
3. Pre každý uzol $v_i \in O$:
   a. využitím relačného klasifikátora: $\hat{c}_i \leftarrow \mathcal{M}_R(\mathcal{N}_i)$, pričom uzly, ktorých klasifikácia doposiaľ nie je známa, sa ignorujú. Ak sú všetky susedné uzly ignorované, do vektora $\hat{c}_i$ sa dosadí hodnota $null$
   b. klasifikácia v_i:

$$x_i = c_k$$
$$k = argmax\ \hat{c}_i(j)$$

   kde $\hat{c}_i(j)$ je j-ta hodnota vektora $\hat{c}_i$. Uzol v_i teda klasifikujeme triedou s najvyšším likelihood-om.
4. Opakovanie krokov po dobu $T = 1000$ iterácií, alebo pokiaľ žiadny z vrcholov nenadobudne novú klasifikáciu. Výstupom algoritmu je klasifikácia uzlov podľa poslednej iterácie.

V tejto kapitole sme podrobne opísali všetky súčasti fungujúceho relačne orientovaného klasifikačného systému – s dôrazom na druhú a tretiu časť – priame relačné klasifikátory a collective inference. Vidíme, že pre každú z častí existuje mnoho metód. Ich správny výber je nevyhnutný predpoklad pre dosiahnutie úspešnej klasifikácie. Otázka voľby preto nie je jednoduchá a veľakrát ani jednoznačná. Pred výberom samotných klasifikačných metód je preto potrebné v prvom rade pochopiť dáta, na ktoré budú klasifikátory aplikované. Platí

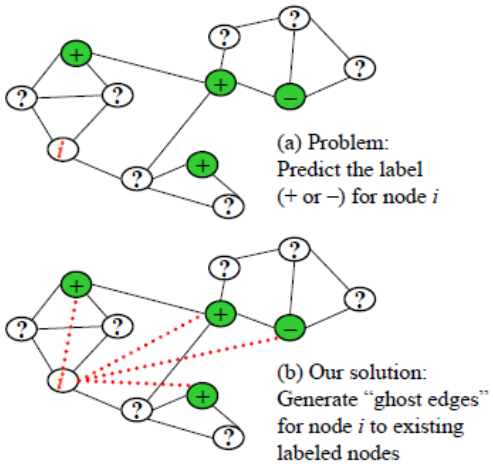
tiež, že rôzne kombinácie metód dávajú dobré výsledky pre istý druh problémov a teda žiadna metóda nie je univerzálna.

3.3 Klasifikácia riedko ohodnotených grafov

U doteraz spomínaných metódach sme zatiaľ vždy predpokladali, že okolo uzla sa nachádza dostatok ohodnotených susedov a tak pomocou relačného klasifikátora je možné získať zmysluplnú informáciu, ktorá zvýši presnosť klasifikácie. Predstavme si ale teraz situáciu, keď na vstupe je graf s menej ako 10%, alebo dokonca menej ako 1% ohodnotenými uzlami. Princíp fungovania relačných klasifikátorov je šírenie informácie o príslušnosti k jednej z tried skrz existujúce hrany, spojenia. U riedko označkovaných sieťach je to ale problém, pretože túto informáciu nepoznáme. A keďže nepoznáme klasifikáciu susedných uzlov nejakého vrcholu v_i, tradičné metódy relačnej klasifikácie nám nevedia povedať nič ani o triede samotného uzlu v_i.

3.3.1 Ghost Edges

Jedným riešením je využitie metódy Ghost Edges, ktorá do grafu pridá nové, „tieňové“ ghost hrany spájajúce vrchol, ktorého triedu chceme predikovať s vrcholmi, ktorých klasifikáciu poznáme.



OBR. 2

Prístupy popísané v predchádzajúcej kapitole fungujú, za predpokladu splnenia istých podmienok, na princípe toho, že ohodnotené vrcholy šíria svoju príslušnosť k triede k svojím susedom. V ideálnom prípade pri klasifikácii neznámeho uzla poznáme príslušnosť k triedam jeho susedov a taktiež poznáme charakteristiku relácie, ktorá dva uzly spája. V takomto ideálnom prípade vieme úspešne klasifikovať neohodnotené vrcholy s vysokou úspešnosťou. Avšak poklesom počtu ohodnotených vrcholov v susedstve neohodnoteného uzla priamo úmerne klesá i šanca na jeho správnu klasifikáciu. Na daný problém je pritom možné nahliadať z dvoch perspektív:

- neohodnotený uzol je prepojený s dostatočným množstvom okolitých uzlov, avšak len málo z nich je ohodnotených
- v grafe sa nachádza dostatočné množstvo ohodnotených uzlov, no náš jeden konkrétny uzol, ktorý skúmame, je prepojený len s malým počtom z nich

Práve prvý pohľad predstavuje techniky spadajúce pod collective inference, ktoré akoby vo vlnách, postupne ohodnocujú uzly a v nejakej *i*-tej iterácii ich určite ohodnotia všetky, pričom pribúdajúcimi iteráciami sa sieť postupne ustáli do finálnej podoby, ktorá predstavuje konečné riešenie. Jednotlivé techniky sa líšia tým, ako sa výsledky každej z vĺn navzájom ovplyvňujú, prípadne koľko iterácií je potrebné vykonať, aby graf dokonvergoval k riešeniu. Dané riešenie, ktoré poskytujú techniky collective inference, teda postupne ohodnotí uzly v grafe, až nakoniec ohodnotí aj priamych susedov nášho neohodnoteného uzla. Samozrejme, ohodnotenie týchto susedov nie je stopercentne bezchybné, a tým sa zníži aj pravdepodobnosť úspešného ohodnotenia nášho konkrétneho uzla. Vo všeobecnosti preto úspešnosť klasifikácie týchto techník s pribúdajúcou riedkosťou v grafe klesá, pretože šíria čoraz väčšie množstvo šumu a chýb v grafe.

Druhý pohľad stojí za ideou techniky Ghost Edge. Inak povedané, riešenie pre tento pohľad tkvie v zistení, že v grafe je málo hrán a preto spravíme to najjednoduchšie – pridáme do neho nové hrany. Keďže neohodnotený uzol je priamo prepojený s ohodnotenými uzlami, grafom sa nešíri šum a chyby a, ako ukázali výsledky, ak charakter dát spĺňa isté predpoklady, je technika Ghost Edge úspešná i pri riedko ohodnotených grafoch.

Otázkou však ostáva výber vhodných uzlov, ktoré budú spojené s klasifikovaným uzlom v_i. Intuícia nám našepkáva, že by to mali byť uzly, ktoré sú s ním v čo najbližšom vzťahu. Ako ale tento vzťah kvantifikovať? V metóde Ghost Edges je to vyriešené nasledovne:

Ghost hrany sú pridané medzi každý jeden pár uzlov, kde jeden z nich je ohodnotený a druhý nie. Následne je týmto hranám pridelená váha, ktorá je odvodená od vzdialenosti uzlov, ktoré spája. Na určenie vzdialenosti sa využíva Random Walk with Restart (RWR) algoritmus, ktorý pridelí vysoké skóre takej hrane $r_{i,j}$ medzi uzlami v_ia v_j, ak medzi nimi existuje veľa vysoko ohodnotených krátkych ciest.

3.3.1.1 Relačné klasifikátory Ghost Edges

Metóda Ghost Edges v sebe skrýva dva relačné klasifikátory:

- GhostEdgeNL – klasifikátor bez učenia
- GhostEdgeL – klasifikátor s učením

Oba klasifikátory distribuujú ohodnotenie vrcholov v grafe skrz ghost hrany, avšak líšia sa v týchto dvoch bodoch

- využitie ohodnotení vrcholov – GhostEdgeNL jednoducho len predpokladá, že vrcholy spojené ghost hranami majú tendenciu mať rovnaké ohodnotenie a táto tendencia rastie úmerne s hodnotou váhy hrany medzi nimi. GhostEdgeL využíva ohodnotené vrcholy v grafe ako tréningovú množinu pre učenie logistickej regresie, pričom učenie prebieha zvlášť na pôvodných susedoch a zvlášť na susedoch prepojených ghost hranami.
- využitie RWR vzdialeností – GhostEdgeNL využíva všetky ghost hrany, pričom najbližším pridelí najväčšie váhy. GhostEdgeL zadelí ghost hrany do niekoľkých tried podľa ich vzdialeností a následne z ohodnotených dát sa naučí váhy pre každú triedu hrán.

Predstavme si prezentované klasifikátory bližšie.

3.3.1.2 Ghost EdgeNL

Klasifikátor Ghost EdgeNL je klasifikátor bez učenia, veľmi jednoducho aplikovateľný a kombinovateľný so všetkými relačnými klasifikátormi. Jeho práca spočíva vo vytvorení nového grafu nasledujúcim algoritmom.

Algoritmus Ghost EdgeNL

1. Vytvor hrany medzi klasifikovaným uzlom *u* a všetkými ohodnotenými uzlami.
2. Pomocou RWR ohodnoť vytvorené hrany. Pôvodné hrany v grafe sa v ďalšom behu algoritmu už nebudú brať do úvahy.
3. Na takto vzniknutý ohodnotený graf aplikuj niektorý z prístupov relačných klasifikátorov a ohodnoť uzol *u*.

3.3.1.3 Ghost EdgeL

Prístup Ghost EdgeL využíva *logForest*, čo je súbor niekoľkých logistických regresií a vďaka tomu dokáže odhaliť rôzne závislosti medzi susedmi klasifikovaného uzla a jeho ohodnotením (respektíve medzi charakterom (vzdialenosť, typ hrany, trieda) susedov a klasifikáciou neohodnoteného uzla). Logistická regresia je dobre známa data-miningová metóda, spadajúca medzi metódy učenia s učiteľom. Na jej vstup je potrebné dodať presne určený počet atribútov a pred začatím samotnej klasifikácie je potrebné ju natrénovať. Popíšme teda kompletný algoritmus.

Algoritmus Ghost EdgeL

1. vyrobíme súbor 500 logistických regresií tvoriacich *logForest*
2. trénovacia fáza
    a. pre každý ohodnotený uzol sa vyrobí nový graf, v ktorom figurujú pôvodné, ale aj ghost hrany. Ghost hrany sú ohodnotené pomocou RWR.
    b. ghost hrany rozdelíme do šiestich skupín na základe ich váh nasledovne:

        skupina A – 3% najlepšie ohodnotených ghost hrán

        skupina B – obsahuje ghost hrany z intervalu 3% - 6%

        skupina C – obsahuje ghost hrany z intervalu 6% - 12%

        skupina D – obsahuje ghost hrany z intervalu 12% - 25%

        skupina E – obsahuje ghost hrany z intervalu 25% - 40%

        skupina F – obsahuje ghost hrany z intervalu 40% - 80%

    c. vytvoríme vstupné atribúty regresií nasledovne:

- počty takých susedov z každej triedy, ktorí sú spojení hranami pôvodného grafu
- počty takých susedov z každej triedy a z každej skupiny vytvorenej v kroku b., ktorí sú spojení ghost hranami

d. natrénovanie *logForest*-u, pričom každá regresia dostane na vstup *log (M) + 1* atribútov, kde *M* je celkový počet atribútov

3. klasifikačná fáza
   a. opäť pre každý neohodnotený uzol vytvoríme nový graf obsahujúci všetky pôvodné, ale aj ghost hrany. Ghost hrany sú ohodnotené pomocou RWR.
   b. podobne ako v tréningovej fáze, určíme početnosť skupín a hodnoty jednotlivých atribútov
   c. zosumarizovaním výsledkov regresií z *logForest*-u dostaneme výslednú klasifikáciu daného uzla

Ako je vidno, Ghost EdgeL algoritmus využíva na klasifikáciu ako pôvodné hrany grafu, tak aj ghost hrany a navyše sa dokáže naučiť, kedy a ktorým má prikladať väčšiu váhu.

Podľa publikovaných výsledkov v 0 prezentované Ghost Edges relačné klasifikátory sú vo všeobecnosti minimálne tak úspešné, ako predchádzajúce metódy a v prípade riedko ohodnotených grafov často i lepšie. V prípade Ghost EdgeL je badať typickú vlastnosť metód s učením a to že pri veľmi malom množstve tréningových dát algoritmus nedokáže vytvoriť dostatočne presný model, čoho následkom je potom menej presná klasifikácia. Dôležité je tiež spomenúť menšiu, alebo až žiadnu závislosť týchto klasifikátorov na prítomnosti homofílie v grafe.

4 Návrh vlastnej metódy relačnej klasifikácie

Doposiaľ predstavené algoritmy majú spoločný jeden menovateľ – pri klasifikácii konkrétneho uzla sa vždy pozrieme na jeho susedov a na základe ich príslušnosti do niektorej z tried ohodnotíme náš neohodnotený uzol.

Pri použití priamych relačných klasifikátorov to je zrejmé – pravdepodobnosť príslušnosti uzla do niektorej z tried je o to vyššia, o čo vyšší je počet susedov klasifikovaných do tejto triedy a tiež o čo vyššie je ohodnotenie hrán prepojených uzlov. Následným použitím

kolektívneho usudzovania docielime to, že ohodnotený uzol ovplyvní nie len svojich priamych susedov, ale aj susedov svojich susedov a podobným rekurzívnym princípom i vzdialené neohodnotené uzly. Tento vplyv s narastajúcou vzdialenosťou výrazne klesá, prípadne je úplne eliminovaný, keďže spomínané techniky využívajú aj náhodné prvky, čím sa môže dosť výrazne ovplyvniť následný chod algoritmu. Výhodou náhodného prvku je však to, že algoritmus sa nezasekne v lokálnom extréme a naopak, dokáže nájsť i globálne extrémy. V relačnej klasifikácii si to môžeme predstaviť nasledovne: lokálnym extrémom je ohodnotenie jedného uzla. Ak je tento uzol ohodnotený správne, dosiahneme lokálne extrém. Klasifikovaný uzol však môže mať veľký počet susedov, navyše spojených hranami s vysokou váhou. To potom znamená, že ohodnotenie daného uzla do značnej miery ovplyvní aj jeho okolie a to buď negatívne, teda zhorší ich klasifikáciu, alebo pozitívne, teda zlepší klasifikáciu. Občas je preto vhodné obetovať správnu klasifikáciu jedného uzla a docieliť tým správnu klasifikáciu jeho susedov, keďže konečným cieľom je korektná klasifikácia čo najväčšieho počtu uzlov a nie korektná klasifikácia niektorých vybraných uzlov.

Rovnako i GhostEdge prístupy klasifikujú uzol na základe jeho okolia – i keď tu je už okolie rozšírené pridaním a ohodnotením ghost hrán do grafu. Tým pádom je uzol priamo spojený s väčším počtom uzlov, spravidla práve tých, ku ktorým sa dá dostať tými najrýchlejšími, najkratšími cestami. Predikcia ohodnotenia neznámeho uzla je potom vypočítaná práve na základe tried týchto okolitých susedov a váh hrán, ktorými sú prepojení. Ešte efektívnejší je ale prístup GhostEdge s učením, ktorý sa pomocou logistickej regresie dokáže naučiť klasifikáciu neohodnotených uzlov podľa počtu susedov v jednotlivých triedach, podľa hodnoty váh ich hrán a podľa typu hrán. Tým pádom je táto technika schopná klasifikovať uzly v takých grafoch, ktoré nevykazujú homofíliu. GhostEdge s učením dokáže odhaliť a naučiť sa akékoľvek pravidlo, nech už vyplýva z akejkoľvek vlastnosti konkrétneho grafu.

4.1 Motivácia

Skúsme sa ale na problém pozrieť z iného uhlu pohľadu. Nie je ojedinelý jav, že sa v grafe nachádzajú uzly s rôznym počtom susedov – sú uzly, ktoré sú hranami napojené na veľký počet okolitých uzlov a teda majú veľa susedov, a naopak, sú uzly, do ktorých smeruje len niekoľko málo hrán a teda majú len málo susedov. Predstavme si graf klebetníc – klebetnica, ktorá má pod palcom celú obec, s každým sa pozná a za jeden deň je schopná nadobudnúť

veľké množstvo nových informácií bude práve taký uzol v grafe, ktorý je napojený na veľký počet iných uzlov. Nazvime ju klebetnica *Janka.* Klebetnica *Janka* je navyše veľmi zdatná v rétorike, preto dokáže primäť svojich susedov, aby ju počúvli a priklonili sa k jej názorom. V obci však býva veľa iných klebetníc. Napríklad aj klebetnica *Danka,* ktorá je taktiež prepojená s veľa obyvateľmi, ale v rétorike za *Jankou* výrazne zaostáva, preto sa darmo snaží, nik ju v dedine nechce počúvať a jej názory nenachádzajú pochopenie. Čo to ale znamená pre graf? Popísaná situácia hovorí o tom, že máme dve silné centrá, ktoré sú prepojené s veľkým množstvom uzlov, pričom jedno z týchto centier ovplyvňuje svoje okolie viac a druhé menej. Ak sa do obce prisťahuje niekto nový, je veľký predpoklad, že klebetnice *Janka* i *Danka* ich navštívia a vznikne prepojenie medzi novým uzlom a klebetnicami. Keďže ale *Janka* je oveľa lepšia rečníčka, noví obyvatelia (nové uzly) sa priklonia k jej názorom, nechajú sa ňou ovplyvniť a je veľká pravdepodobnosť, že v dôležitých otázkach týkajúcich sa obce budú držať zajedno, teda sú akoby klasifikované v jednej triede.

4.2 Prvotný návrh

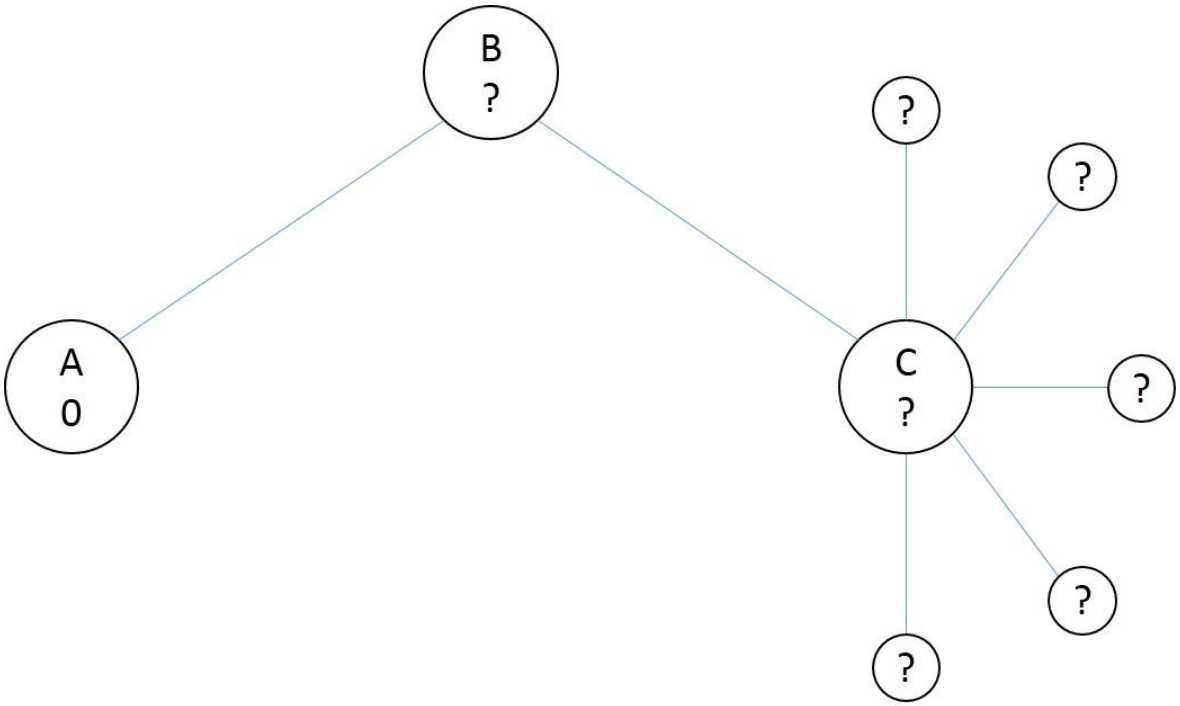
A tu sa dostávame k informatickej podstate veci. Hneď na úvod uvedieme, že v našom návrhu počítame s prítomnosťou homofílie v dátach. Je to základný predpoklad pre fungovanie našej metódy, i keď, ako neskôr uvidíme, algoritmus do istej miery funguje i s datasetmi nevykazujúcimi homofíliu.

Každý ohodnotený uzol v grafe má teda svojich susedov a istý počet týchto susedov je klasifikovaný do rovnakej triedy. Ak je podiel rovnako klasifikovaných susedov v okolí klasifikovaného uzlu vysoký, znamená to, že tento uzol má silný vplyv na svoje okolie. Inými slovami možno povedať, že okolie tohto uzla vykazuje homofíliu. Uzol preto vyšle gossip (správu) k jeho neohodnoteným susedom, aby si svoju triedu nastavili práve na tú jeho. A keďže homofília je v tomto prípade prítomná, gossipu je priradená vysoká váha (priamo úmerná podielu susedných uzlov s rovnakou triedou). Naopak, ak rovnako klasifikovaní susedia majú len malý podiel, znamená to, že okolie uzla nevykazuje homofíliu a teda vyšle k svojim neohodnoteným susedom gossip s veľmi nízkou váhou o tom, že ich trieda sa zhoduje.

Zopakovaním tohto procesu pre každý ohodnotený uzol sa dopracujeme ku konečnému stavu a to že každý jeden neohodnotený uzol disponuje súborom gossipov o tom, aby si nastavili

tú-ktorú triedu. Finálna klasifikácia neohodnoteného uzlu tak spočíva vo vhodnej agregácii doručených gossipov s prihliadnutím na ich kvalitu (váhu) a kvantitu (početnosť správ pre jednotlivé triedy).

Čo však v prípade, ak všetky susedné uzly neohodnoteného uzla sú tiež neohodnotené? Situáciu ilustruje nasledovný obrázok. Techniky kolektívneho usudzovania riešia túto situáciu zopakovaním algoritmu v niekoľkých iteráciách, čoho výsledkom je, že postupne sa dopracujeme k ohodnoteniu susedov a následne i k ohodnoteniu neohodnoteného uzlu. Teda ak chceme ohodnotiť uzol C, musíme ohodnotiť najprv jeho susedov . Takýto prístup by bol možný i pre náš algoritmus, avšak znamenalo by to, že informácia od uzlu A by v prvej iterácii dorazila len k uzlu B. Uzol B by bol ohodnotený a až ten by následne vyslal gossip k uzlu C. Uzol B však môže mať mnoho iných susedov a tak jeho ohodnotenie sa nemusí zhodovať s triedou uzlu A. Tým pádom vyslaný gossip z uzlu B sa už netýka tej istej triedy, ako pôvodný gossip z uzlu A. Gossip z uzlu A sa tak stratí, nemá možnosť ovplyvniť klasifikáciu uzla C.



OBR. 3

Naša metóda sa s daným problémom vysporiada tak, že gossip z uzlu je vyslaný nie len do najbližších – priamych susedov, ale i do susedov jeho susedov – teda nepriamych susedov spojených skrz jeden, dva, až *n* uzlov. Gossipy odoslané do vzdialenejších uzlov je možné penalizovať, napríklad jednoduchým predelením váhy vzdialenosťou, vďaka čomu gossipy

vyslané k blízkym uzlom budú mať väčšiu výslednú váhu ako gossipy odoslané k vzdialeným uzlom. Tomu zodpovedá i jednoduchá logická úvaha – uzol vyšle do okolia svoj gossip, avšak aj keby to bol veľmi „silný“ uzol (teda má veľa susedov klasifikovaných do rovnakej triedy), len s veľmi malým vplyvom môže ovplyvniť uzly nachádzajúce sa v grafe niekde na opačnom konci (teda najkratšia cesta medzi nimi vedie cez niekoľko desiatok, stoviek, tisícov uzlov). Diaľka, do ktorej sa gossip distribuuje, je jeden zo vstupných parametrov našej metódy a o vplyve jeho hodnoty na dosiahnuté výsledky pojednáme ešte neskôr v kapitole o vyhodnotení našich experimentov.

Toto bola základná prvotná idea, myšlienkový postup demonštrujúci ako a prečo sme sa rozhodli problém relačnej klasifikácie riešiť pomocou gossipov. V nasledujúcich riadkoch opíšeme priebeh analyzovania a výsledky rôznych metód pre jednotlivé časti algoritmu ako výpočet váhy gossipu, jeho odoslanie a následné spracovanie, agregovanie gossipov u neohodnotených uzlov. Ešte predtým uvádzame pseudokód prvotného návrhu nášho prístupu:

Prvotný Gossip Pseudokód

1. určenie váhy ohodnotených uzlov (zdrojov gossipov), inými slovami kvantitatívne ohodnotiť, či je uzol centrum, ktoré vyšle do okolia gossipy s vysokou, alebo naopak s nízkou váhou
2. určiť váhu konkrétneho gossipu od ohodnoteného uzlu *A* k neohodnotenému uzlu *B* vyplývajúc z váhy uzlu *A* a vzdialenosti medzi uzlami *A* a *B*
3. odoslanie gossipov a ich agregácia v každom neohodnotenom uzle

4.3 Analýza algoritmu

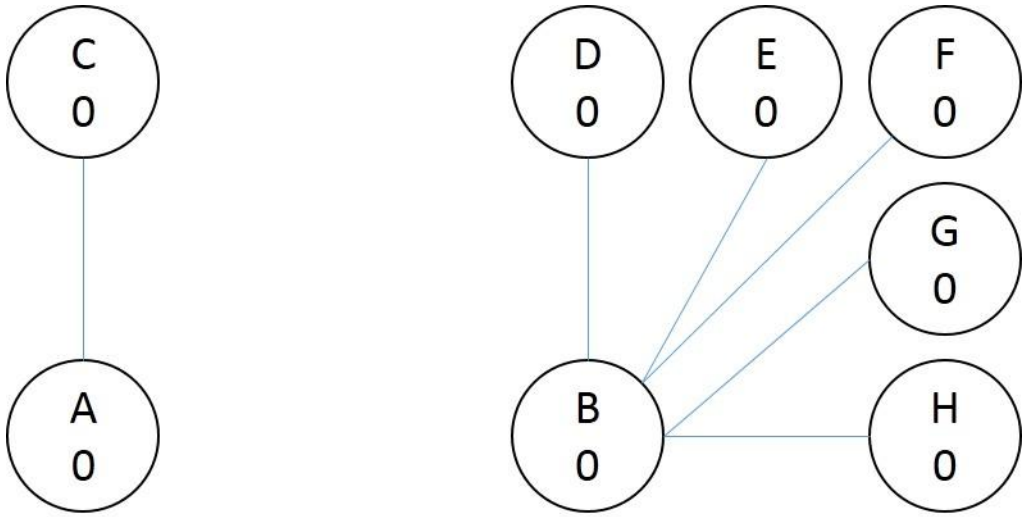
Náš algoritmus sa teda skladá z troch hlavných častí. Predstavíme teraz úvahy, ktorými sme sa dopracovali k rôznym metódam v každej z nich.

4.3.1 Určenie váhy ohodnotených uzlov

V prvom rade je potrebné priradiť každému ohodnotenému uzlu váhu hovoriacu o tom, či je silným centrom „klebiet“, teda gossipov, alebo nie. Keďže vychádzame z prítomnosti homofílie v dátach, prvotnou myšlienkou bolo priradenie váhy uzlu *A* priamo úmerne počtu

susedov uzla *A* v pomere k celkovému počtu susedov uzlu *A*. Teda váhu uzla *A* sme vyjadrili ako podiel počtu rovnako ohodnotených susedov a celkového počtu susedov.

Už i takéto priradenie váhy uzlu spĺňalo svoj účel, algoritmus dokázal klasifikovať neohodnotené uzly, avšak mal jednu nevýhodu. Predstavme si situáciu znázornenú na nasledujúcom obrázku.



OBR. 4

Vidíme dva uzly – uzol *A* a uzol *B*. Oba uzly majú ohodnotených susedov – uzol *A* má jedeného suseda, ktorý je v rovnakej triede ako uzol *A*, teda v triede *0*. Uzol *B* má až piatich susedov a opäť sú všetci klasifikovaní do rovnakej triedy, bez ujmy na všeobecnosti nech je to znovu trieda *0*. Akú váhu by sme priradili uzlom *A* a *B*? Uzlu *A* by to bola váha $w_A = \frac{1}{1} = 1$. Uzlu *B* by sme priradili váhu $w_B = \frac{5}{5} = 1$. Uzlom *A* a *B* by sme priradili rovnakú váhu. To však nezodpovedá intuícii, ktorú nadobudneme pri pohľade na obrázok. Oba uzly majú rovnaký pomer počtu susedov klasifikovaných do rovnakej triedy k celkovému počtu susedov. Nedáva nám ale fakt, že uzol *B* má vyšší počet susedov, vyššiu pravdepodobnosť, že i prípadný ďalší uzol spojený s uzlom *B* bude klasifikovaný do rovnakej triedy? Zrejme áno. Preto sme sa rozhodli určiť váhu uzlu podielom počtu susedov klasifikovaných do rovnakej triedy a maximálnym počtom susedov aký sa v grafe nachádza. Ak sa vrátime k obrázku, znamená to, že ak uzol s maximálnym počtom susedov má 5 susedov, tak váhy uzlov *A* a *B* nastavíme nasledovne: $w_A = \frac{1}{5}$; $w_B = \frac{5}{5} = 1$. Podobným princípom ak by v grafe existoval ešte uzol, ktorý je prepojený s 10-imi uzlami, znamenalo by to, že váhy uzlov *A* a *B* nastavíme na hodnoty $w_A = \frac{1}{10}$; $w_B = \frac{5}{10} = \frac{1}{2}$. Výsledný efekt takejto úpravy je

zrejmý – na to, aby bol uzol považovaný za centrum, ktoré do veľkej miery ovplyvní svoje okolie (teda je to veľká „klebetnica“), už nestačí mať len vysoký podiel svojich susedov v rovnakej triede, musí to byť aj uzol, ktorý je prepojený s veľkým počtom uzlov. Takto sa nám podarilo nájsť v grafe silné centrá a za predpokladu prítomnosti homofílie v dátach sme dosiahli i vylepšenie úspešnosti klasifikácie.

Čo však v prípade, ak dáta homofíliu nevykazujú? A na dôvažok je graf nevyvážený v tom zmysle, že v ňom existujú celky(podgrafy), ktoré homofíliu vykazujú viac a niektoré menej?

Problémom je už samotná absencia homofílie – spôsobuje, že len veľmi ťažko možno nájsť silné centrá a aj tie nájdené sú len nízko ohodnotené. To v konečnom dôsledku zapríčiní malý počet prichádzajúcich gossipov pre neohodnotené uzly, čím sa znižuje úspešnosť predikcie.

Tieto úvahy nás k viedli k pokusu nájsť funkciu, ktorá by zohľadnila dva faktory – homofíliu v celom grafe a homofíliu lokálnu, v okolí jedného uzla. Ich vzájomným prepojením by sme dostali jednu hodnotu – váhu pre daný uzol. Otázka je, aký priebeh by takáto funkcia mala mať. Skúsme si preto rozobrať situáciu v okolí štyroch hraničných bodoch, najprv ale ukážeme, ako homofíliu meriame.

Za hodnotu lokálnej homofílie v skutočnosti dosadíme práve podiel, ktorý sme odvodili. Nech $max =$ počet susedov toho uzla, ktorý má najvyšší počet susedov a $n =$ počet susedov uzla *u*. Lokálnu homofíliu uzla *u* potom určíme podielom $\frac{n}{max}$.

Hodnotu globálnej homofílie vypočítame nasledovne: najprv vyrobíme maticu priradení medzi triedami, nazvime ju *M*, s rozmermi $n * n$, kde *n* je počet rôznych tried v grafe. Prvok $e_{i,j}$ v matici *M* predstavuje podiel takých hrán z celého grafu, ktoré smerujú z uzlov s triedou c_i do uzlov s triedou c_j. V prípade, že pracujeme s neorientovanými grafmi, môžeme jednoducho predpokladať že hrana je obojsmerná. Z toho vyplýva že $\sum_{i,j} e_{i,j} = 1$.

Následne:

$$a_i = \sum_j e_{i,j}$$

$$b_j = \sum_i e_{i,j}$$

$$homofília = \frac{\sum_i e_{i,j} - \sum_i a_i * b_i}{1 - \sum_i a_i * b_i}$$

Obe homofílie nadobúdajú hodnoty z intervalu $< \mathbf{0}, \mathbf{1} >$**.** Hraničnými bodmi sú teda body [0,0] ; [0,1] ; [1,0] ; [1,1] (body sú uvedené v tvare [*globálna homofília, lokálna homofília*]).

V bode [1,1] dataset vykazuje homofíliu na globálnej aj lokálnej úrovni. V tomto prípade je situácia vždy jasná. Znamená to totiž, že globálne v celom grafe platí, že hrany sú vedené zväčša medzi uzlami v rovnakej triede a aj v rámci susedstva konkrétneho uzla *u* sa nachádza veľké množstvo uzlov s rovnakou triedou, akú má uzol *u*. To znamená, že takémuto uzlu chceme priradiť čo najvyššiu váhu, preto mu udelíme váhu rovnú 1. Od našej funkcie teda očakávame, že v bode [1,1] vráti hodnotu 1.

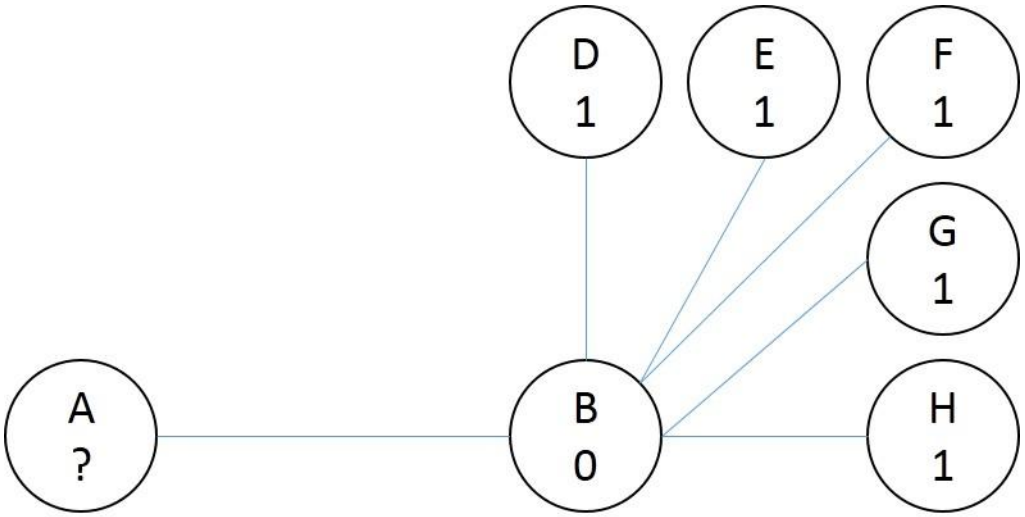
V prípade kombinácie [1,0] z globálneho hľadiska homofília platí, ale uzol *u* nemá žiadnych susedov v rovnakej triede. V tomto prípade je otázne akú váhu chceme uzlu *u* prideliť. Vychádzajúc z globálnej homofílie by sme mu mali udeliť najvyššiu možnú váhu. Pri pohľade na okolie vidíme ale opak, žiadnemu zo susedov nie je pridelená rovnaká trieda, preto vychádzajúc z lokálnej homofílie by sme mali uzlu prideliť najmenšiu možnú, teda nulovú váhu. Pravda bude zrejme niekde uprostred, núka sa nám možnosť prideliť uzlu váhu s hodnotou 0,5. Ako neskôr uvidíme, vyskúšali sme rôzne funkcie, s rôznym prístupom k prideleniu váhy v takýchto sporných bodoch.

V prípade kombinácie [0,1] je situácia obdobná ale s opačným garde. Globálne graf nevykazuje žiadnu homofíliu, ale práve tento jeden konkrétny uzol *u* je taký, že má veľké množstvo rovnako ohodnotených susedov. Znovu je to situácia, kedy s prihliadnutím na jeden parameter by sme udelili uzlu najvyššiu váhu, ale s prihliadnutím na druhý parameter práve naopak. Testy rôznych funkcií v ďalšej časti ukážu, pri akých hodnotách sme dostali najúspešnejšiu predikciu.

Treba povedať, že extrémne kombinácie [0,1] a [1,0] v reálnych dátach možné nie sú, nakoľko ide o dva navzájom sa vylučujúce prípady. Predpokladajme, že každý uzol má aspoň jedného suseda (inak by aplikovať relačnú klasifikáciu na dáta ani zmysel nemalo – uzol, ktorý nie je v žiadnej relácii s inými uzlami nemožno klasifikovať na základe relácii). Potom ak je v dátach homofília prítomná stopercentne, znamená to, že každý uzol je spojený výhradne s uzlami z rovnakej triedy a teda nemôže nastať situácia, že uzol nemá suseda z rovnakej triedy. Teda extrémna situácia [1,0] nie je možná. Rovnaká je situácia ohľadom kombinácie [0,1] keď globálne graf vykazuje heterofíliu, čo znamená, že všetky uzly sú

prepojené len s uzlami inej triedy. Potom ale nie je možné aby existoval uzol, ktorý je prepojený s čo i len jedným uzlom v rovnakej triede. Navyše si môžeme položiť otázku – je vôbec relačná klasifikácia potrebná ak vieme, že dáta vykazujú stopercentnú homofíliu? Zrejme potom každý nový neohodnotený uzol bude klasifikovaný do tej triedy, v ktorej je klasifikovaný uzol, s ktorým je prepojený. Samozrejme, neohodnotený uzol môže byť prepojený i s viac ako len jedným uzlom, a tieto uzly nemusia byť v jednej triede – v tom prípade je už ale homofília narušená. V prípade heterofílie to nie je také jednoznačné. Heterofília hovorí len toľko, že všetky uzly v susedstve sú klasifikované do inej triedy, nehovorí však nič o tom, do ktorej z nich. Iste, ak sú dáta klasifikované do dvoch tried, potom pri platiacej heterofílii je celé susedstvo klasifikované do „tej druhej“ triedy. Tried ale môžeme mať desiatky, stovky. Preto samotná znalosť heterofílie pre klasifikáciu nových uzlov, aj keby bola stopercentná, nie je dostačujúca, pokiaľ klasifikujeme do viac ako dvoch tried. Samotné dve hraničné situácie teda nie sú v praxi reálne, avšak situácie v ich okolí už áno.

Ešte sme sa ale nevenovali situácii, ktorú popisuje dvojica [0,0]. Teda situácii, kde z globálneho hľadiska dáta vykazujú heterofíliu a rovnaká je aj situácia v okolí konkrétneho uzlu – žiadny z jeho susedov nemá rovnakú triedu. Zrejme sa tu, celkom intuitívne a logicky, núka riešenie priradiť takémuto uzlu nulovú váhu. Zamyslime sa však nad otázkou – bude potom v takomto grafe aspoň jeden uzol, ktorý by mal nenulovú váhu? V celom grafe platí heterofília a jej dôsledkom je, že každý uzol je prepojený len s uzlami z iných tried. Tým pádom by sme boli nútení každému uzlu priradiť nulovú váhu. Tak by sme ale rozposielali iba gossipy s nulovou váhou – výsledkom by bolo, že síce by k neohodnoteným uzlom gossipy dorazili, ich váhy by ale boli nulové a nebol by žiaden gossip, ktorý by niesol hodnotnú informáciu. Prakticky je to stav rovný s tým, že žiadne gossipy k neohodnoteným uzlom nedorazili a teda nemáme ich na základe čoho klasifikovať. Na obrázku vidíme, ako by takýto prípad mohol vyzerať v prípade klasifikácie do dvoch tried.



OBR. 5

Na rozdiel od situácií [0,1] a [1,0], ktoré reálne nastať nemohli, je toto situácia, ktorá nastať môže. Čo by sa teda stalo, keby sme situácii v bode [0,0] priradili váhu jednotkovú? Áno, zrejme by k neohodnotenému uzlu A dorazil gossip z uzlu B o tom, aby si svoju váhu nastavil na triedu *0*. Ale k uzlu A dorazia i gossipy z uzlov D až H, ktorých jednotlivá váha bude síce nižšia, pretože už to nie sú priami susedia, ale ich počet je výrazne vyšší a preto ich celkový súčet prevýši súčet gossipov z triedy *0*. Uzol A by si teda v konečnom súčte vybral triedu *1,* čo je správna klasifikácia ak má v grafe platiť heterofília.

Charakter dát, čo sa týka homofílie a heterofílie, môže byť rôzny. Stopercentná homofília alebo heterofília sú extrémne stavy, no nie nemožné. I pre situáciu [0,0] sme preto vyskúšali priradiť uzlom rôzne váhy. Ktorý prístup bol najúspešnejší, uvidíme o chvíľu.

4.3.1.1 Funkcie pre určenie váhy ohodnotených uzlov

Celkovo sme vyskúšali päť rôznych funkcií s rôznym priebehom s ohľadom na vyššie popísané podmienky. Konkrétne sú to tieto funkcie:

[1] $váha\ uzla = (globálnaHomofília * lokálnaHomofília)^2$

[2] $váha\ uzla = globálnaHomofília * lokálnaHomofília$

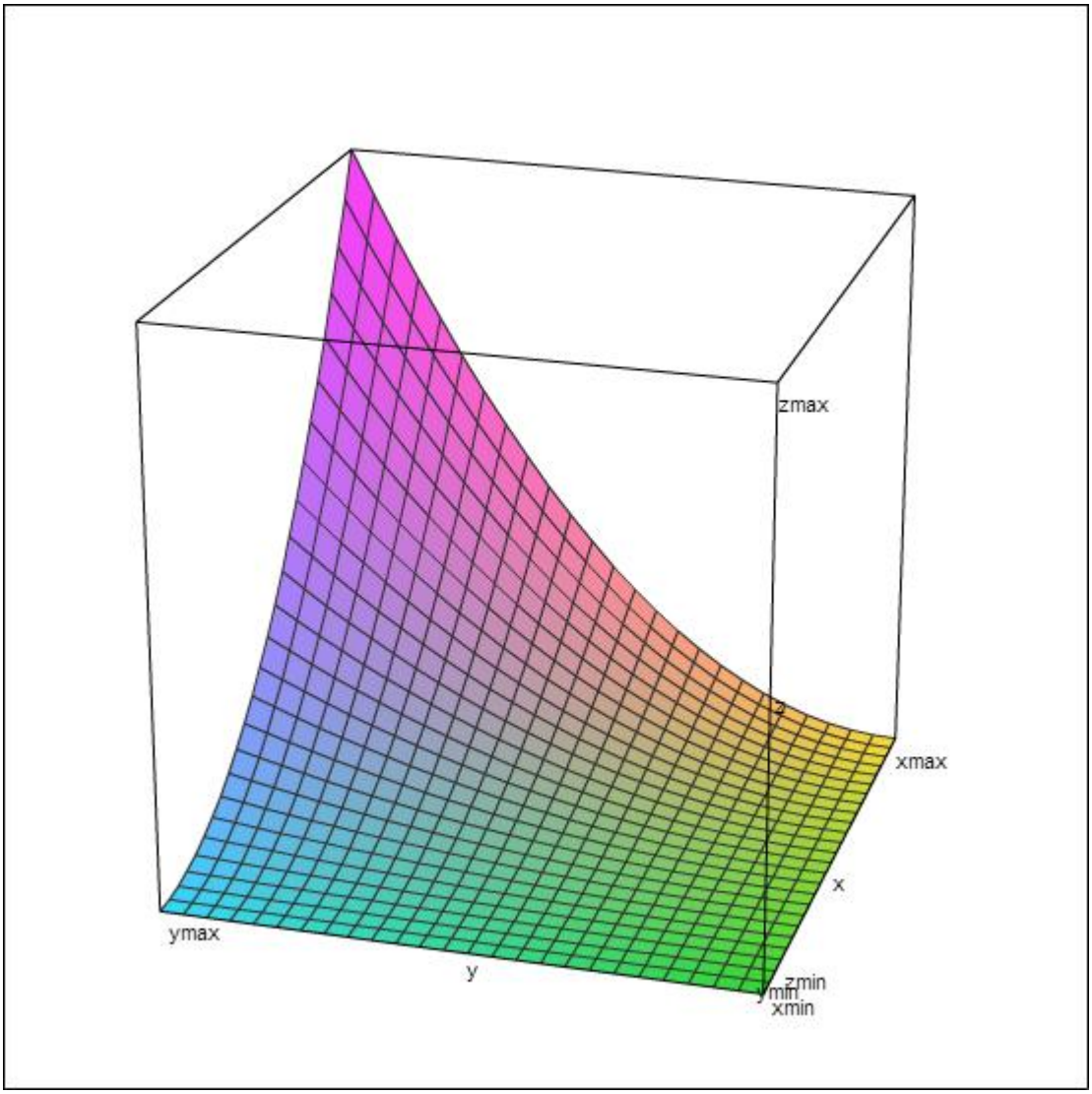
[3] $váha\ uzla = \max[globálnaHomofília * lokálnaHomofília; globálnaHomofília^2 - 0{,}5; lokálnaHomofília^2 - 0{,}5]$

[4] $váha\ uzla = globálnaHomofília^{0.6} * lokálnaHomofília^{0.6}$

[5] $váha\ uzla = 1 - (globálnaHomofília - lokálnaHomofília)^2$

Predstavme si teraz jednotlivé funkcie bližšie.

4.3.1.1.1 Funkcia [1]:



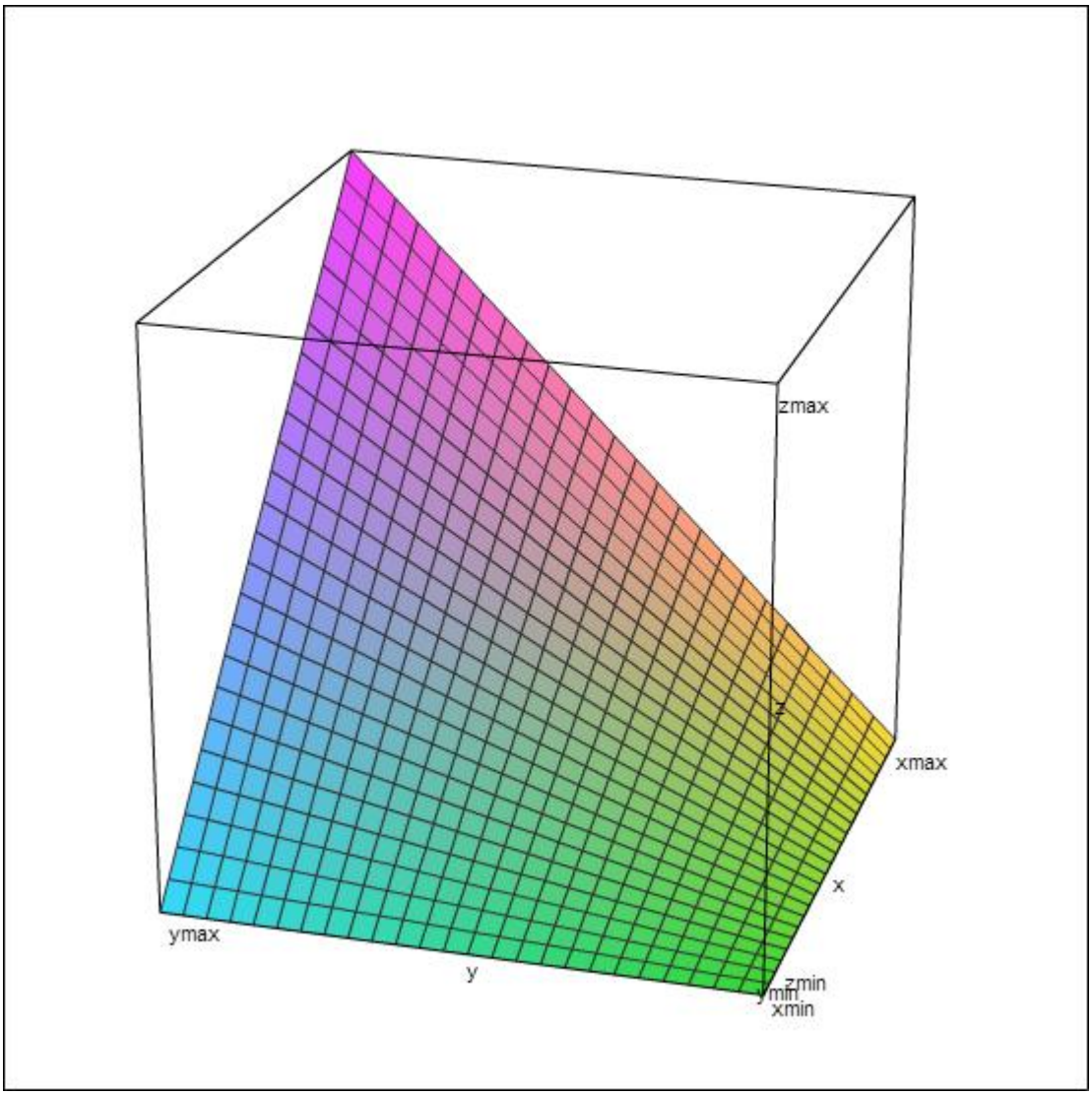
OBR. 6

Na úvod uvádzame popis jednotlivých ôs na grafoch funkcií: x-ová os predstavuje globálnu homofíliu, y-ová os predstavuje lokálnu homofíliu a z-ová os predstavuje výslednú váhu uzla. Hodnoty na všetkých troch osiach sú z intervalu [0,1].

Prvá funkcia udeľuje vysoké váhy uzlom len v prípade vysokej prítomnosti homofílie z globálneho i lokálneho hľadiska. Smerom k ostatným trom hraničným bodom hodnota pridelenej váhy výrazne klesá. Podľa očakávaní takáto funkcia dokáže úspešne klasifikovať len dáta vykazujúce homofíliu, v iných prípadoch úspešnosť výrazne klesá, nakoľko sa zníži počet gossipov s nenulovou váhou. Neohodnotené uzly tak majú k dispozícii len veľmi

obmedzený, prípadne až nulový počet použiteľných gossipov, čo negatívne vplýva na úspešnosť predikcie.

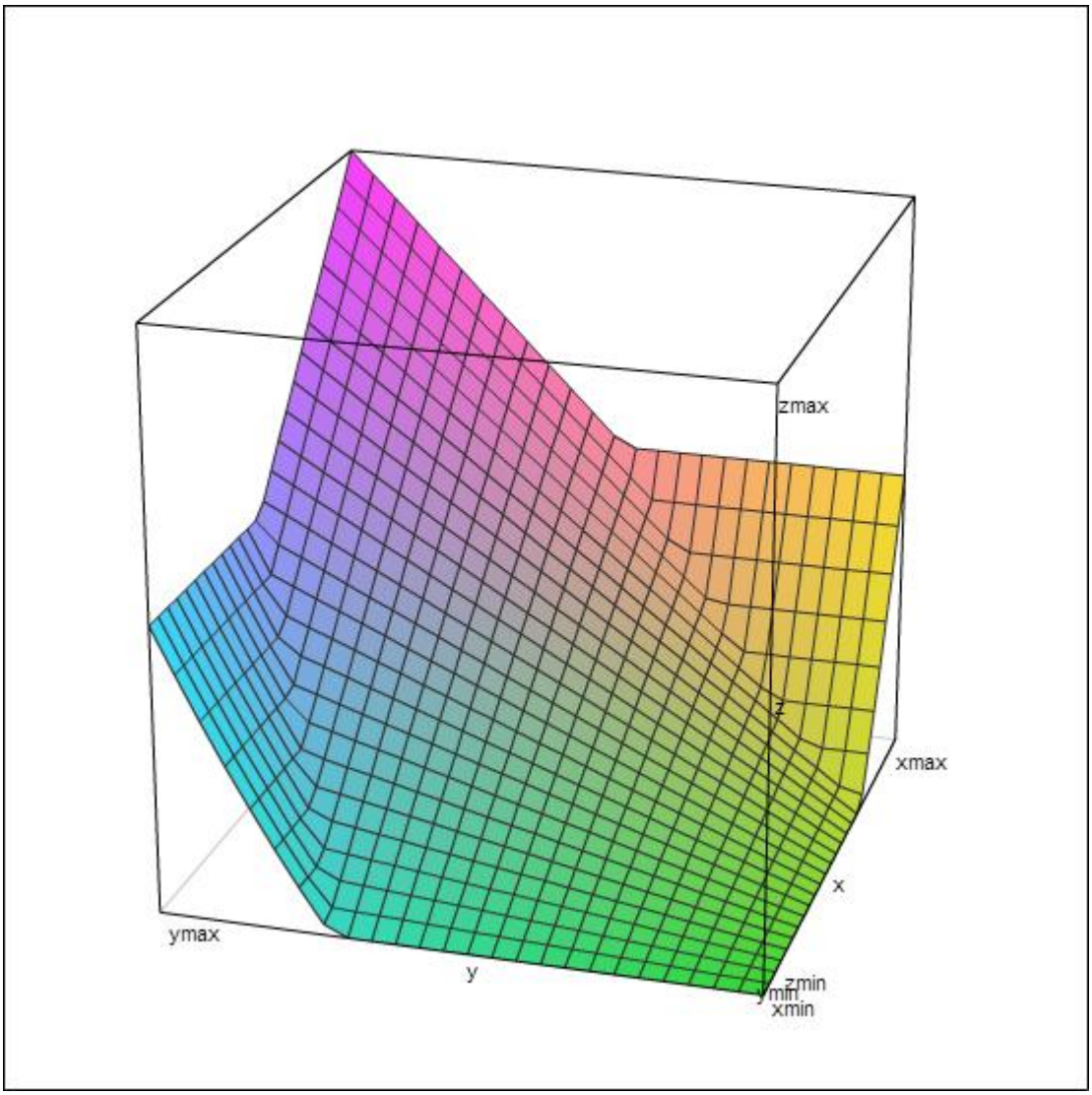
4.3.1.1.2 Funkcia [2]:



OBR. 7

Druhá funkcia prideľuje v hraničných bodoch rovnaké váhy ako prvá funkcia. Na rozdiel od prvej funkcie, ktorá rastie exponenciálne, rastie druhá funkcia lineárne a preto v okolí bodov [0,0]; [1,0]; [0,1] sú hodnoty pridelených váh vyššie. Pomocou druhej funkcie sme dosiahli len o málo lepšie výsledky predikcie, nakoľko sa mierne zvýšila váha prideľovaná uzlom a tým pádom sa nepriamo zvýšila i váha prideľovaná gossipom.

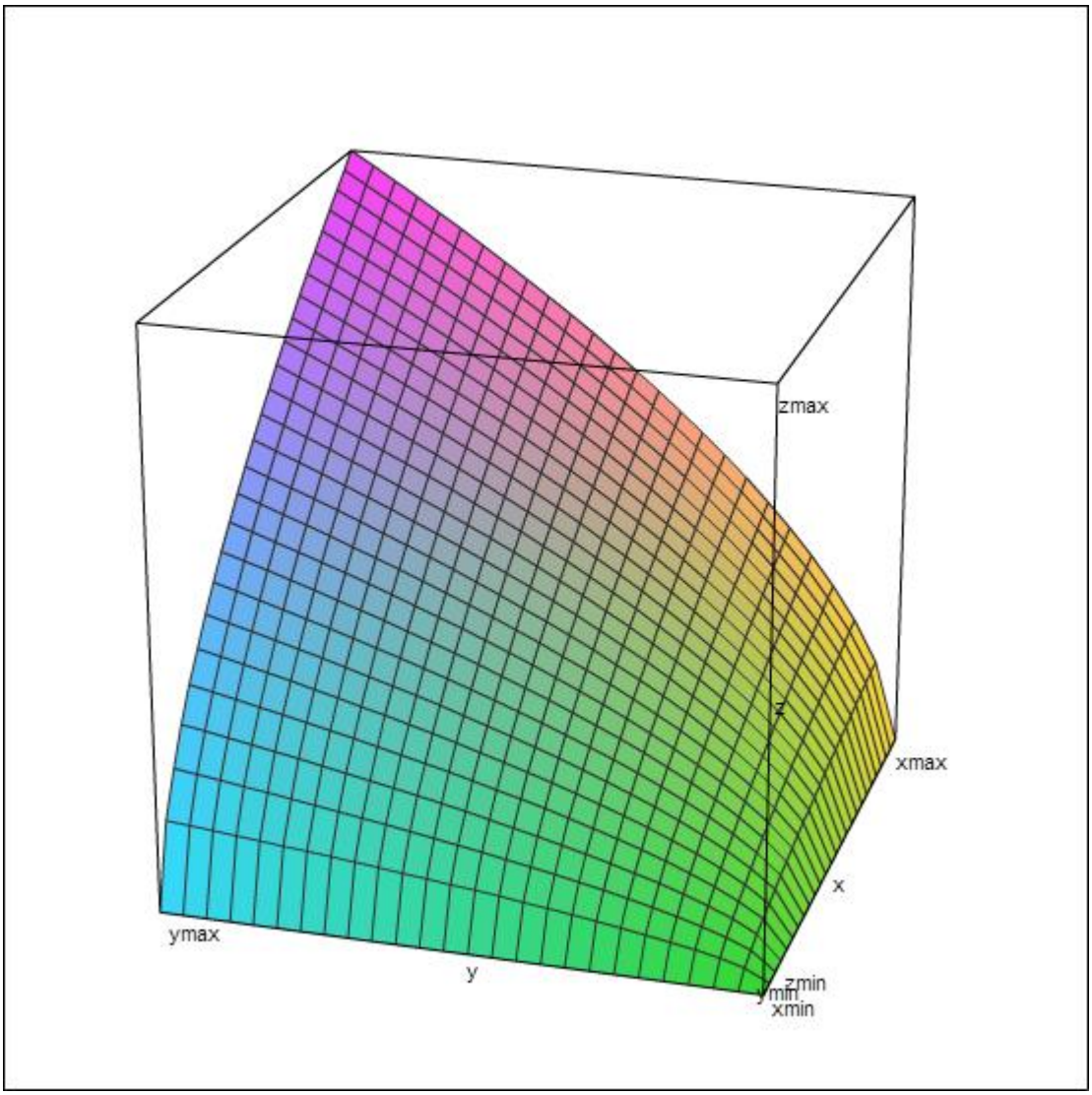
4.3.1.1.3 Funkcia [3]:



OBR. 8

Keďže z výsledkov bolo zrejmé, že pre dosiahnutie lepších výsledkov predikcie je potrebné zvýšiť váhy gossipov, vyskúšali sme funkciu, ktorej priebeh je popísaný maximom ďalších troch. Týmto krokom sme zvýšili váhy uzlom nachádzajúcim sa v blízkosti bodov [0,1] a [1,0] čím sa nám znovu podarilo zvýšiť úspešnosť predikcie až tak, že to bola jedna z dvoch funkcií, ktoré dosiahli najlepšie výsledky. Rozdiel oproti predchádzajúcim funkciám sa ukázal hlavne pri veľmi riedko ohodnotených grafoch, kde bol rozdiel v úspešnosti klasifikovania neohodnotených uzlov približne 15 percent.

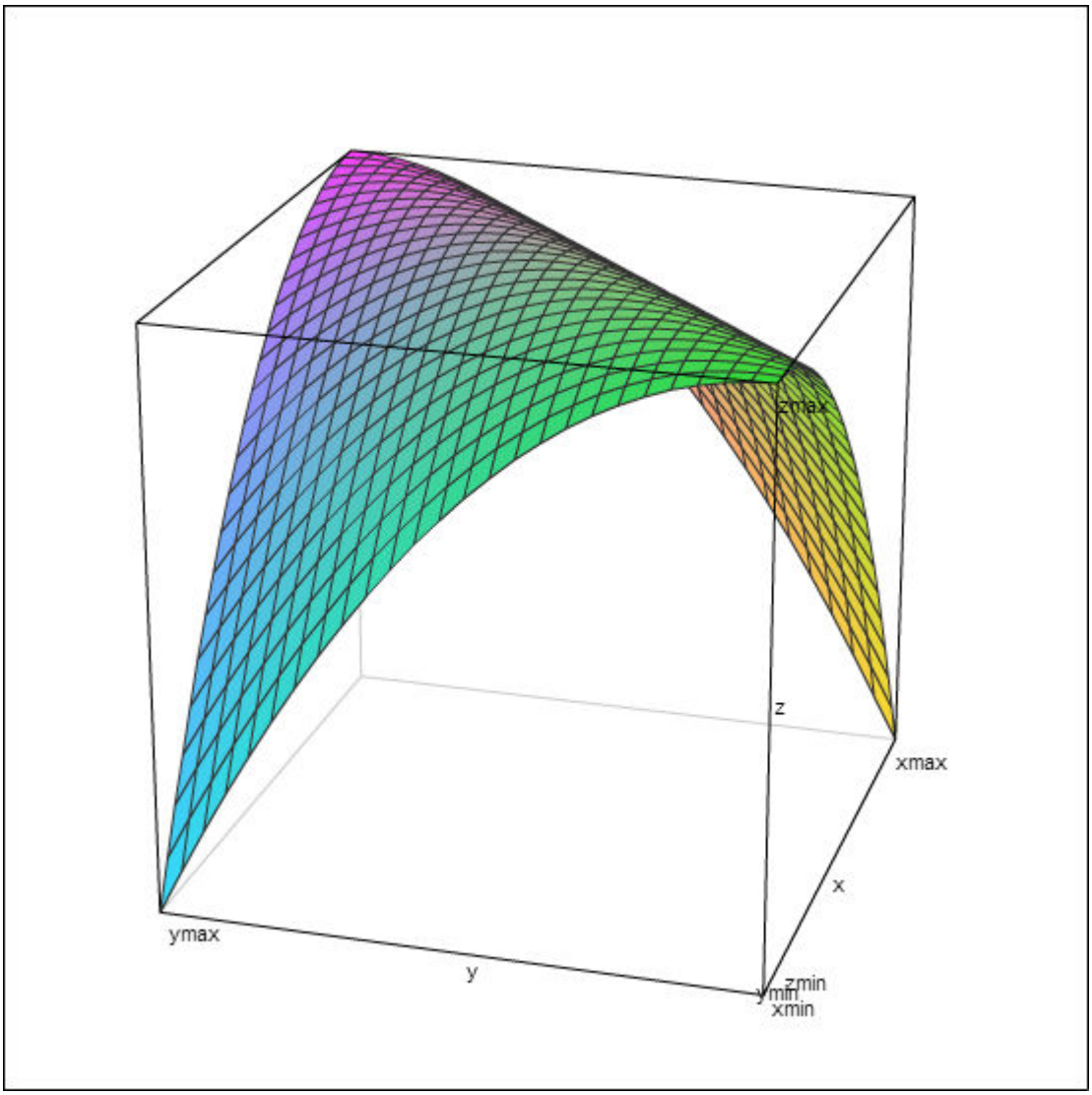
4.3.1.1.4 Funkcia [4]:



OBR. 9

Pri štvrtej funkcii sme sa na znovu vrátili k idei nulových váh uzlov v bodoch [0,1] a [1,0]. Avšak funkcia rastie smerom k [1,1] rýchlejšie. Takýmto riešením sme dosiahli lepšie výsledky ako v prípade funkcií 1 a 2, ale len v prípade malého počtu neohodnotených uzlov. Akonáhle sa počet neohodnotených uzlov zvýšil, funkcie 1, 2 a 4 klasifikovali približne s rovnakou úspešnosťou.

4.3.1.1.5 Funkcia [5]:

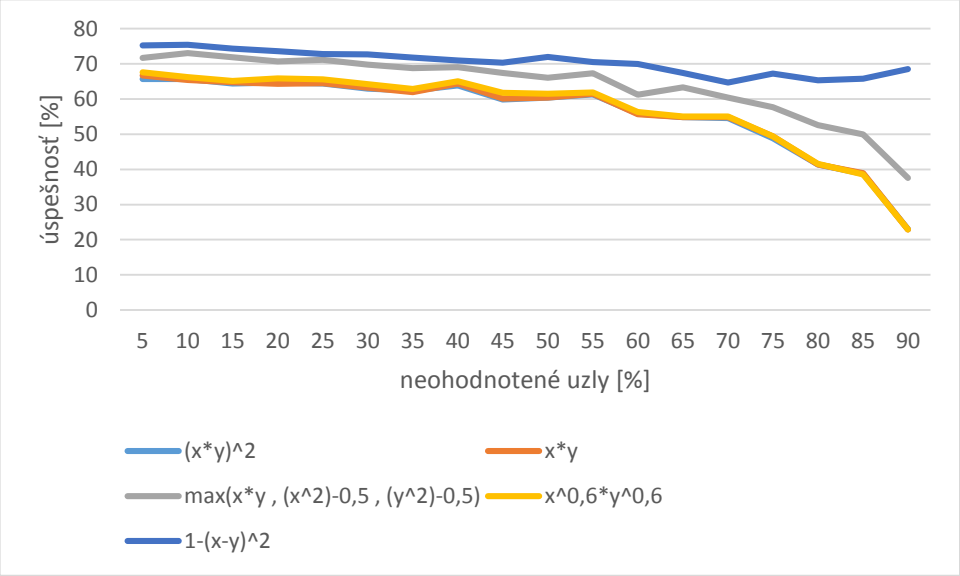


OBR. 10

Najviac sa nám osvedčila funkcia posledná, kde sme vyskúšali priradiť najvyššie váhy uzlom nachádzajúcim sa na priamke ([0,0],[1,1]). Uzlom v bodoch [1,0] a [0,1] je síce pridelená nulová váha, ale ako sme hovorili, reálne sa tu žiaden uzol nachádzať nemôže. Od týchto dvoch bodov rastie funkcia veľmi strmo až k priamke ([0,0],[1,1]), kde je uzlom priradená jednotková váha. Táto funkcia prideľuje nízke váhy len uzlom, ktoré sú v rozpore medzi globálnym a lokálnym pohľadom na graf. Teda ak v grafe z globálneho hľadiska je homofília prítomná (prípadne neprítomná) a zároveň z lokálneho hľadiska je homofília neprítomná (prípadne prítomná). Uzol je postavený pred dve protichodné rozhodnutia – tým pádom je pravdepodobnosť, že uzol vyšle gossip, ktorý bude správne klasifikovať svoje okolie nízka. A preto takémuto uzlu priradíme len malú váhu, pričom sa spoliehame, že v neďalekom okolí sa nachádzajú uzly s vysokými váhami a tie sa postarajú o silné,

spoľahlivé gossipy, ktoré správne klasifikujú neohodnotené uzly. Keďže táto funkcia dávala na testovaných dátach najlepšie výsledky, práve s touto sme neskôr pokračovali v ďalších testoch a porovnaniach s inými technikami relačnej klasifikácie.

Namerané výsledky testovaní s datasetom CORA (o použitých datasetoch širšie pojednáme neskôr v kapitole o vyhodnotení) v grafe ukázali, že najúspešnejšia je funkcia 5, hlavne v prípadoch vyššieho podielu neohodnotených uzlov v datasetoch. V grafe sú priemerné hodnoty z testovania na sto datasetoch, kompletné výsledky uvádzame v prílohe A.



GRAF 1

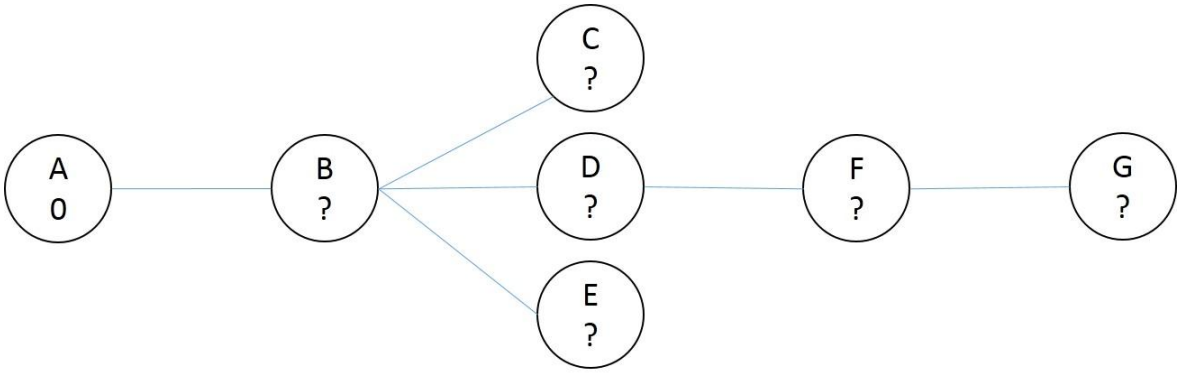
4.3.2 Určenie váhy gossipu

V predošlej kapitole sme sa venovali určeniu váhy ohodnotených uzlov, čím sme vyjadrili do akej miery je ten-ktorý uzol silným centrom. Čím je uzol silnejším centrom, tým vyššia váha mu bola pridelená. Skúmaním sme sa dopracovali k funkcii, ktorá prideľuje uzlom váhu na základe prítomnosti homofílie v grafe a to z pohľadu globálneho (prítomnosť homofílie v celom grafe) a lokálneho (prítomnosť homofílie v okolí uzla).

Ďalším krokom algoritmu je rozoslanie gossipov od ohodnotených k neohodnoteným uzlom. Nato ale potrebujeme každý gossip odoslaný z uzlu A do uzlu B ohodnotiť. Znovu sme vyskúšali rôzne riešenia, ktoré budeme v tejto kapitole prezentovať. Rozdeliť ich môžeme na dve základné kategórie – prvou je kategória riešení, ktoré na určenie váhy gossipu vychádzajú, okrem iného, z váhy pridelenej zdrojovému uzlu gossipu a druhou kategóriu sú riešenia, ktoré priraďujú váhu gossipom na základe počtu susedov a pravdepodobnosti výberu cieľového uzla gossipu zo všetkých okolitých uzlov.

4.3.2.1 Riešenia využívajúce váhu zdrojového uzla gossipu

Sme teda v situácii, že každý ohodnotený uzol má priradenú svoju váhu a chceme, aby tieto uzly vyslali do svojho okolia gossipy. Naskytá sa ale otázka, čo to vlastne okolie je? Mohli by sme zaň považovať len okolitých, priamych susedov uzla. Takéto riešenie by fungovalo, avšak len v prípade, že je neohodnotených uzlov málo a ďalšou podmienkou je, aby v grafe bolo dostatok hrán. V opačnom prípade nedokážeme klasifikovať neohodnotené uzly z toho dôvodu, že k nim jednoducho nedorazia žiadne gossipy. Ak totiž v okolí neohodnoteného uzla sú len ďalšie neohodnotené uzly, nemajú sa k nemu ako gossipy dostať. Ako sme už spomínali v prvotnom návrhu našej metódy, jedným z riešení je zopakovanie celého algoritmu až nakoniec minimálne jeden zo susedov bude v nejakej *n*-tej iterácii ohodnotený. V nasledujúcej *n+1* iterácii tento ohodnotený sused vyšle gossip k neohodnotenému uzlu a tak ho vieme ohodnotiť. Písali sme tiež aké sú negatíva takéhoto riešenia a preto sme zvolili prístup taký, že gossipy budú vyslané nie len do svojho najbližšieho okolia, ale i do vzdialenejších uzlov, až po nejakú hraničnú vzdialenosť, ktorú predom určíme. Takto sa dopracujeme k výsledku, že od ohodnoteného uzla je vyslaný gossip do každého neohodnoteného uzla ktorého vzdialenosť je menšia ako nejaká konštanta. Túto situáciu popisuje nasledovný obrázok:



OBR. 11

Uzly B až G sú neohodnotené a všetky prijmú gossip od uzla A. Položme si ale otázku, má mať gossip odoslaný z uzlu A do uzlu B rovnakú váhu ako gossip odoslaný od uzlu A k uzlu G? Áno, zdrojový uzol je stále ten istý a teda aj váha zdroja stále rovnaká. Avšak gossip k uzlu G prechádza cez veľa hrán a iných uzlov. Nezabudnime, že pri určení váhy uzlu A sme brali do úvahy aj lokálnu homofíliu. Čím ďalej budeme od uzla A, tým viac sa lokálna homofília môže líšiť a tým nepresnejší bude gossip vyslaný do tejto vzdialenosti. Preto váhy

gossipov vyslaných z jedného uzla by sa mali líšiť, pokiaľ ich vzdialenosť od zdrojového gossipu je rôzna.

Váhy gossipov sme sa preto rozhodli penalizovať vzdialenosťou zdrojového a cieľového uzla. Otázkou ešte ostáva, ako určíme vzdialenosť medzi dvoma uzlami.

V prvom rade si spomeňme, aké dáta máme k dispozícii. Sú to uzly a hrany medzi nimi, pričom hrany sú ohodnotené, čím sa vyjadruje kvalita spojenia, relácie (napríklad ak by hrana predstavovala spoločnú publikáciu dvoch vydavateľov, tak váha hrany by bola rovná počtu spoločných publikácií). Platí, že čím je váha hrany vyššia, tým je vzťah dvoch uzlov silnejší. My túto skutočnosť využijeme na vybudovanie grafu s rovnakými uzlami i hranami, ale váhu hrany pretransformujeme na vzdialenosť a to tak, že vzdialenosť bodov, ktoré spája hrana h bude rovná podielu $\frac{1}{h_w}$ kde h_w je váha hrany h. Váhy v grafe sú z intervalu [1,n], kde n je nejaké prirodzené číslo. Tým pádom takto vypočítané vzdialenosti sú z intervalu $[\frac{1}{n}, 1]$ a platí, že čím bola váha hrany vyššia, tým je jej vzdialenosť nižšia. Ak sú dva uzly spojené s hranou s vysokou váhou, znamená to, že ich vzdialenosť je nízka.

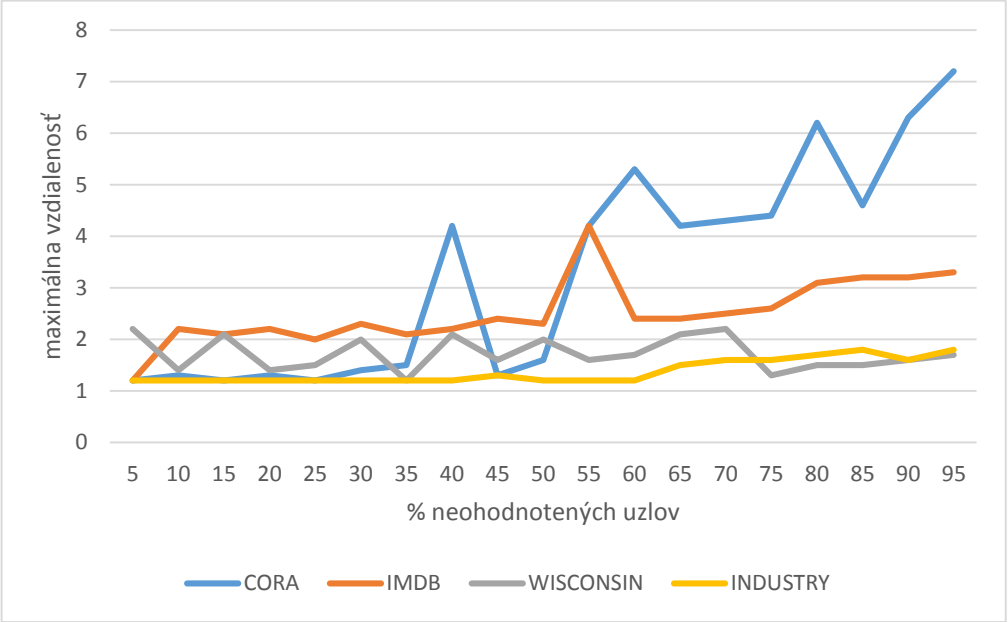
Máme teda graf, poznáme ohodnotené aj neohodnotené uzly a vypočítali sme vzdialenosti medzi každou dvojicou spojených uzlov. Výslednú váhu gossipu smerujúceho z uzlu A do uzlu B určíme podielom $\frac{A_w}{d_{A,B}}$ kde A_w je váha uzla A a $d_{A,B}$ je minimálna vzdialenosť medzi uzlami A a B. Minimálna vzdialenosť susedov, ktorí sú priamymi susedmi zdrojového uzlu gossipu, je daná priamo vzdialenosťou hrany medzi nimi. Minimálna vzdialenosť uzlov z ostatného okolia (teda takých uzlov, ktoré nie sú priamym susedom) je vypočítaná pomocou Dijkstrovho algoritmu.

Čím je cieľový uzol vzdialenejší, tým je váha gossipu menšia, preto po presiahnutí istej vzdialenosti už nemá zmysel gossipy ďalej rozosielať. Ich vplyv na klasifikáciu uzla bude zanedbateľne malý. Vďaka tomuto poznatku vieme, že nemá zmysel počítať vzdialenosti medzi každou dvojicou uzlov, ale len medzi takými, kde gossip dorazí s dostatočne veľkou váhou.

Skúmali sme jednak vzdialenosť, do akej má zmysel gossipy rozosielať a tiež aj funkciu, ktorou penalizujeme váhu gossipu smerujúceho k viac vzdialeným uzlom.

Čo sa týka optimálnej maximálnej vzdialenosti, do ktorej sa gossip odosielal, sa výsledky na štyroch datasetoch značne líšili. Jedným faktorom, ktorý ju ovplyvňuje, je počet hrán. Čím

je vyšší, tým menšia maximálna vzdialenosť je dostačujúca na to, aby k neohodnotenému uzlu dorazil dostatočný počet gossipov na korektnú klasifikáciu. Ďalším faktorom sú váhy hrán. Ak sú váhy hrán vysoké, tak sú vzdialenosti, naopak, nízke. Potom ale stačí aj malá maximálna vzdialenosť na to, aby bol gossip vyslaný do dostatočného počtu uzlov. Posledným faktorom je pomer ohodnotených a neohodnotených uzlov v datasete. Čím je neohodnotených uzlov viac, tým ďalej je potrebné odosielať gossipy.

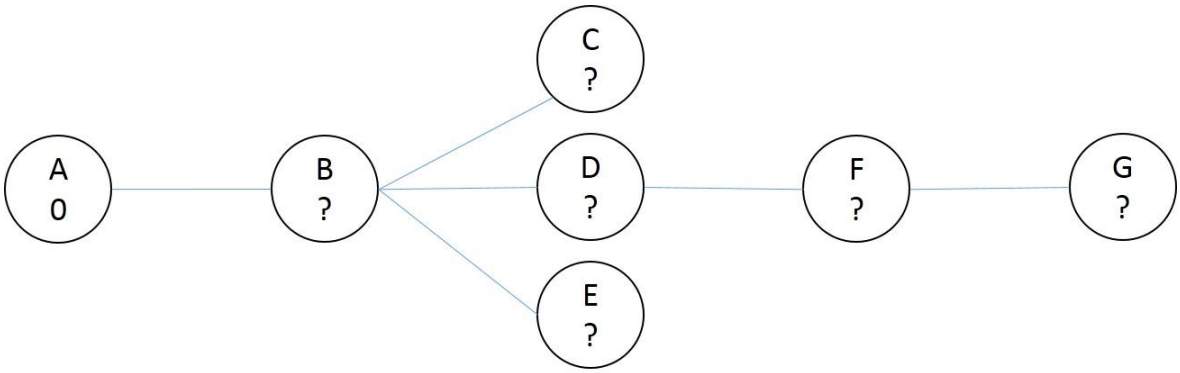


GRAF 2

V grafe vidíme, pri akej maximálnej vzdialenosti sme dosiahli najlepšiu klasifikáciu neohodnotených uzlov. Pri klasifikácii uzlov v datasete CORA sa optimálna maximálna vzdialenosť veľmi výrazne začala zvyšovať pokiaľ sme mali klasifikovať viac ako 50% uzlov z datasetu a naopak pri malých datasetoch ako INDUSTRY a WISCONSIN algoritmus najlepšie klasifikoval neohodnotené uzly vždy s veľmi podobnou maximálnou vzdialenosťou.

4.3.2.2 Riešenia využívajúce pravdepodobnosť

Pod týmto nejasným názvom sa skrývajú také metódy, ktoré určujú váhu gossipu podľa toho, aká je pravdepodobnosť, že sa náhodným výberom jednou z možných hrán dostane gossip od zdrojového uzlu až po cieľový. Demonštrujme takéto riešenie na nasledujúcom obrázku:



OBR. 12

Keďže má uzol *A* len jednu hranu, pravdepodobnosť, že sa náhodným výberom hrany dostaneme z uzlu *A* do uzlu *B* je rovná jednej. Formálne zapísané $G_{A,B} = P(A,B)$, kde $G_{A,B}$ je váha gossipu z uzlu *A* do uzlu *B* a $P(A,B)$ je pravdepodobnosť, s akou sa dostaneme z uzlu *A* do uzlu *B* pri náhodnom výbere jednej z hrán uzlu *A*. Gossip z uzlu *A* do uzlu *D* by sme mohli vypočítať takto:

$$\boldsymbol{G_{A,D} = P(A,B) * P(B,D)}$$

(1)

Ak ešte chceme k tomu využiť i znalosť o váhach hrán, môžeme túto pravdepodobnosť vynásobiť váhou hrany medzi uzlami *A* a *B*. Takto by nám ale váhy gossipov rástli s pribúdajúcu vzdialenosťou, preto takto vypočítanú váhu ešte predelíme minimálnou vzdialenosťou medzi nimi. Formálne zapísané tento gossip vypočítame nasledovne: $G_{A,B} = \frac{P(A,B)*w_{A,B}}{d_{A,B}}$ kde $w_{A,B}$ je váha hrany medzi týmito dvoma uzlami a $d_{A,B}$ je ich minimálna vzdialenosť. Gossip z uzlu *A* do uzlu *C* by sme následne mohli počítať dvomi spôsobmi:

$$G_{A,C} = \frac{P(A,B) * w_{A,B} + P(B,C) * w_{B,C}}{d_{A,C}}$$

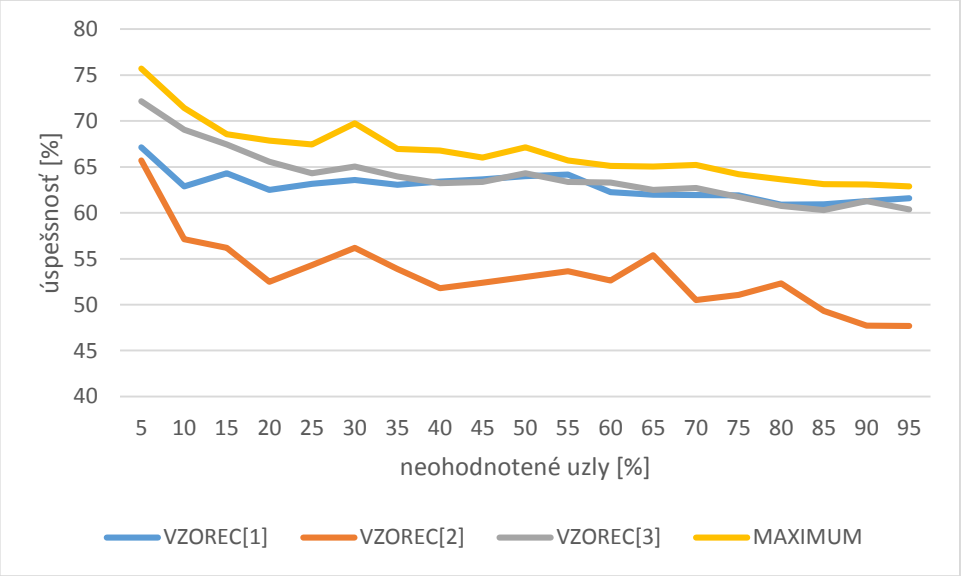
(2)

alebo

$$G_{A,C} = \frac{P(A,B) * w_{A,B} * P(B,C) * w_{B,C}}{d_{A,C}}$$

(3)

Opäť sme sa skúmaním maximálnej vzdialenosť, do akej má zmysel gossipy vysielať snažili nájsť čo najlepšie nastavenie, ale ani jednou z týchto troch metód sme nedosiahli lepšie výsledky ako pri použití prideľovania váh gossipom popísanom v predchádzajúcej kapitole. Porovnanie jednotlivých metód je v nasledujúcom grafe. Zaznamenané sú priemerné hodnoty z testovania na sto rôznych datasetoch IMDB. Jednotlivé vzorce predstavujú pravdepodobnostné prístupy očíslované tak, ako sú vyššie uvádzané. Krivka maximum predstavuje najúspešnejšie riešenie využívajúce váhu zdrojového uzla gossipu. Kompletné výsledky uvádzame v prílohe B.



GRAF 3

4.3.3 Odoslanie a agregácia gossipov

Ukázali sme akým spôsobom ohodnocujeme gossipy a nakoniec sme sa dostali do záverečnej fázy – odoslanie a agregácia gossipov na strane príjemcu. Každý gossip v sebe nesie dve informácie a to triedu, o ktorej gossip hovorí a jeho váhu. Na konci algoritmu sme teda v stave, že každý neohodnotený uzol má zoznam gossipov, ktoré mu boli doručené. Otázkou je, akým spôsobom doručené gossipy vyhodnotiť a ako z nich odvodiť konečný ortieľ – pridelenie triedy neohodnotenému uzlu.

Prvým a najjednoduchším riešením je sčítanie váh všetkých doručených gossipov pre jednotlivé triedy a klasifikovať uzol do tej triedy, ktorej súčet je najvyšší.

Uvažovali sme i nad tým, že z celého množstva doručených gossipov budeme brať do úvahy len určité percento tých najsilnejších. Tento krok ale neviedol k zvýšeniu úspešnosti nášho algoritmu, nakoľko už pri ohodnocovaní gossipov sme sa snažili o rozlíšenie prospešných gossipov, ktoré zvýšia úspešnosť klasifikácie a im sme prideľovali vyššie váhy. Preto tým, že sme agregovali len tie najúspešnejšie doručené gossipy, sme neurobili nič zásadné – odstránili sme len tie gossipy, ktoré aj tak nemali takmer žiaden vplyv na klasifikáciu.

Osvedčilo sa však súčet váh gossipov z jednej triedy predeliť počtom gossipov, ktoré prislúchali danej triede. Týmto dostaneme váhu priemerného gossipu pre každú triedu a neohodnotený uzol napokon klasifikujeme do tej triedy, ktorej prislúcha najvyššia priemerná váha.

Týmto sme ukončili predstavenie Gossip algoritmu, pre skompletizovanie uvádzame výsledný pseudokód obsahujúci tie postupy a metódy, s ktorými sme dosiahli najlepšie výsledky.

4.3.4 Pseudokód Gossip Algoritmu

1. vytvoríme maticu *D,* ktorá bude maticou vzdialeností medzi dvojicami susedných uzlov. Prvok tejto matice $d_{i,j}$ je definovaný vzťahom $d_{i,j} = \frac{1}{w_{i,j}}$ kde $w_{i,j}$ je váha hrany medzi uzlami *i* a *j*.
2. pomocou Dijkstrovho algoritmu vytvoríme maticu *M,* ktorá bude maticou minimálnych vzdialeností medzi uzlami. Prvok $m_{i,j}$ je potom minimálna vzdialenosť medzi uzlami i a j.
3. všetkým ohodnoteným uzlom $v \in V^K$ je pridelená váha $v(w)$ vzťahom $v(w) = 1 - (locH - globH)^2$ kde $locH = \frac{\sum_{x \in N(v)} c(x) = c(v)}{max}$, $max = maximum\big(n: n_i = \sum_{x \in N(v_i)} c(x) = c(v_i)\big)$ a $globH$ je rovné globálnej homofílii v grafe
4. určíme váhu všetkých gossipov $g \in G$ a odošleme, pričom $g_{v,u} =< label, weight >$ kde $g_{v,u}(weight) = \frac{v(w)}{m_{v,u}}$, $g_{v,u}(label) = label(v)$, v je zdrojový uzol gossipu a *u* je cieľový uzol gossipu

5. pre každý neohodnotený uzol $u \in V^U$ vypočítame vektor $T_u = < t_1^u, t_2^u, \dots t_n^u >$ kde n je počet tried a $t_i^u = \sum_{v \in V^k} g_{v,u}(weight)$ pričom $g_{v,u}(label) = i$. Tieto súčty ešte predelíme počtom gossipov pre každú z tried nasledovne $t_i^u = \frac{t_i^u}{|t_i^u|}$
6. každý neohodnotený uzol $u \in V^U$ klasifikujeme do takej triedy *i,* pre ktorú platí $max\{t_i^u | t_i^u \in T_u\}$

5 Vyhodnotenie

V tejto kapitole predstavíme najprv datasety, na ktorých sme jednotlivé algoritmy testovali, porovnáme úspešnosť existujúcich algoritmov a nakoniec predstavíme výsledky nášho algortimu.

5.1 Datasety

Celkovo sme pre naše účely využili štyri datasety, ktoré teraz v krátkosti predstavíme a popíšeme ich základné vlastnosti.

5.1.1 Dataset IMDB

Dataset IMDB je dataset obsahujúci 1440 filmov, ktoré sú klasifikované do dvoch tried. Rozlišujeme nimi či daný film je trhák, alebo nie, pričom film je trhákom, pokiaľ jeho príjem za prvý týždeň premietania presiahne 2 milióny $. Hrán obsahuje dataset celkovo 20317, pričom hrana medzi dvomi filmami existuje, pokiaľ majú spoločného vydavateľa. Váha hrany predstavuje počet spoločných vydavateľov. Počet kladne a záporne klasifikovaných filmov je približne rovnaký. Cieľom klasifikácie je teda odhadnúť, či daný film bude trhákom.

id triedy	interpretácia	počet uzlov
0	nie je trhákom	826
1	je trhákom	615

TAB. 1

5.1.2 Dataset Industry

Dataset Industry je dataset firiem pracujúcich v jednom z priemyselných odvetví. Dataset rozlišuje celkovo 12 odvetví rozdelených podľa spoločnosti Yahoo! a každá firma je klasifikovaná do jedného z nich. Obsahuje celkovo 2189 uzlov a 12357 hrán. Relácia medzi dvomi firmami existuje, ak spoločne vydajú nejaký druh tlačovej správy. Váha hrany hovorí o počte takýchto spoločných tlačových správ. Cieľom klasifikácie je odhadnúť odvetvie pre neohodnotenú publikáciu – neohodnotený uzol.

id triedy	odvetvie	počet uzlov
0	technology	609
1	basic materials	83
2	financial	268
3	trasportation	47
4	healthcare	279
5	services	478
6	consumer NonCyclical	59
7	utilities	69
8	capital goods	78
9	consumer Cyclical	94
10	energy	112
11	conglomerates	13

TAB. 2

5.1.3 Dataset CORA

Dataset CORA je dataset s publikáciami z oblasti informatiky obsahujúci celkovo 4240 uzlov a 35912 hrán. Uzol predstavuje jednu publikáciu a trieda uzla hovorí o oblasti informatiky, ktorej sa publikácia týka. Dataset rozlišuje celkovo sedem tried a dve publikácie sú prepojené pokiaľ majú spoločného autora alebo pokiaľ jedna publikácia cituje druhú. Váha hrany predstavuje počet spoločných autorov a počet vzájomných citácií. Cieľom klasifikácie je odhadnúť oblasť, ktorej sa publikácia týka.

id triedy	oblasť	počet uzlov
0	rule learning	609
1	reinforcement learning	83
2	theory	268
3	neural networks	47
4	probabilistics methods	279
5	genetic algorithms	478
6	case based	59

TAB. 3

5.1.4 Dataset Wisconsin

V datasete Wisconsin uzly predstavujú webové stránky a hrany medzi nimi hovoria o prepojení týchto stránok hyperlinkami. Keďže ide o školské stránky zamestnancov, študentov a stránky samotnej školy, uzly sú klasifikované do tried, ktoré zodpovedajú účelu stránky v štruktúrach školy. Váhy hrán zodpovedajú počtu prepojení medzi dvomi stránkami. Cieľom klasifikácie je, podobne ako pri ostatných datasetoch, zatriediť neohodnotenú stránku do jednej zo stránok.

id triedy	účel stránky	počet uzlov
0	student	155
1	course	85
2	faculty	38
3	department	39
4	project	25
5	staff	12

TAB. 4

V nasledujúcej tabuľke uvádzame niekoľko základných parametrov datasetov.

	CORA	IMDB	Wisconsin	Industry
počet tried	7	2	6	12
počet uzlov	4240	1441	354	2189
počet hrán	35912	20317	16625	12357
minimálny počet susedov jedného uzla	0	0	0	1
maximálny počet susedov jedného uzla	341	181	229	141
priemerný počet susedov jedného uzla	16,94	28,20	93,93	11,29
počet uzlov bez susedov	215	272	6	0
homofília	0,812	0,739	0,712	0,656
minimálna váha hrany	1	1	1	1
maximálna váha hrany	20	7	85	113
priemerná váha hrán	1,21	1,08	1,75	1,84

TAB. 5

Z uvedeného vyplýva, že charakteristika datasetov je rôznorodá – datasety s vyrovnaným počtom uzlov v každej triede ale aj datasety s nerovnomerným rozložením počtov uzlov v jednotlivých triedach. Máme datasety husté, s veľkým počtom hrán ale i riedke, kde sú uzly prepojené len s malým počtom susedov. Rozdiely sú i v miere homofílie a v ohodnoteniach váh hrán.

5.2 Metodológia

V rámci testovania sme náš algoritmus porovnali celkovo s ôsmymi predchádzajúcimi prístupmi, konkrétne: tri techniky collective inferencing (Gibbs Sampling, Relaxation Labeling, Iterative Classification) spárované s relačnými klasifikátormi (Weighted Vote, Class Distribution) nám spolu dávajú šesť rôznych použiteľných kombinácií a k tomu dve techniky využívajúce GhostEdge (GhostEdge with Learning, GhostEdge NonLearning).

Keďže v drvivej väčšine bol relačný klasifikátor Weighted Vote lepší ako Class Distribution klasifikátor, pre prehľadnosť výsledkov ho nebudeme uvádzať. Toto zistenie zodpovedá i výsledkom iných publikácií, kde bol tiež klasifikátor Weighted Vote úspešnejší.

Z každého datasetu sme vytvorili sto kolekcií, každá kolekcia v sebe obsahuje verziu pre rôzny pomer ohodnotených a neohodnotených uzlov v nej. Počet verzií v jednej kolekcii je

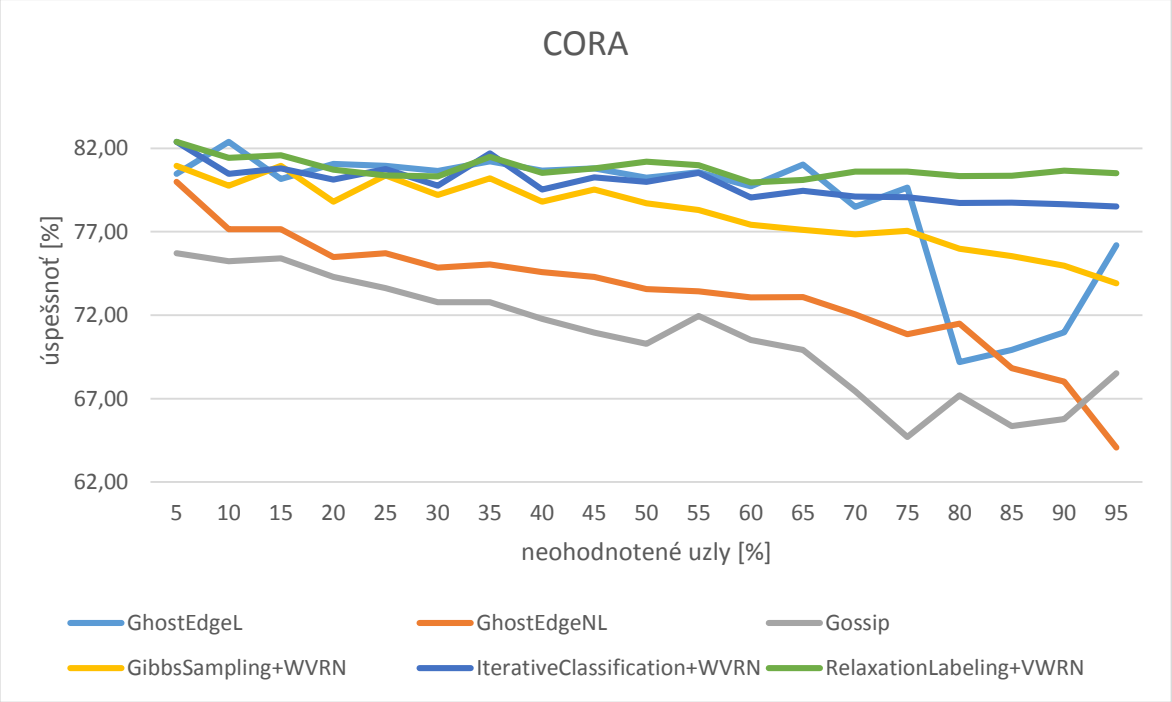
19 čo predstavuje zastúpenie neohodnotených uzlov od 5% až po 95% s rozdielom 5% medzi verziami. Celkovo bol teda jeden algoritmus na jednom datasete spustený celkovo 1900 krát. Vo výsledkoch uvádzame priemer zo všetkých dosiahnutých hodnôt.

Podotýkame tiež, že počas testovania sme používali vlastné implementácie uvedených algoritmov.

5.3 Výsledky

Výsledky vyhodnotíme pre každý dataset zvlášť.

Dataset CORA

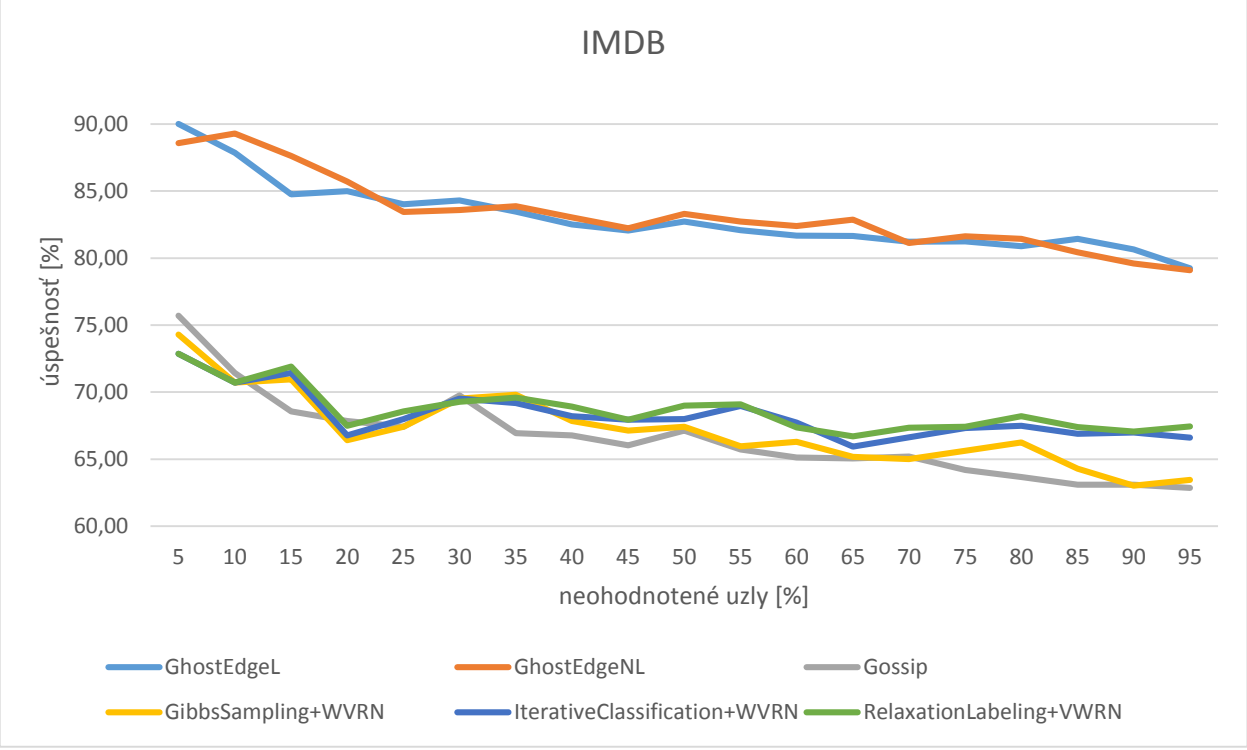


GRAF 4

Dataset CORA má najvyšší počet uzlov i najvyšší počet hrán, priemerne však na jeden uzol pripadá len necelých 17 uzlov, čo je druhý najmenší počet. Je to tiež dataset s najvyššou mierou homofílie zo všetkých nami testovaných datasetov, čo sa prejavilo pri výsledkoch algoritmov, ktorých úspešnosť je jej prítomnosťou podmienená. Veľmi dobre si teda počínali collective inference algoritmy, z nich najlepšie Relaxation Labeling s Weighted Vote

klasifikátorom, ktorý bol konštantne úspešný pre akýkoľvek počet neohodnotených uzlov. Až po 80 percentné zastúpenie neohodnotených uzlov si veľmi dobre počínal i GhostEdgeL prístup, potom ale úspešnosť náhle klesla. Práve GhostEdgeL vykazoval na všetkých datasetoch najväčšie skoky, čo je zrejme spôsobené existenciou istých hraníc pre počet tréningových uzlov. Keďže počet iterácií v procese učenia je pevne daný, pri týchto hraniciach dochádza buď k nedostatočnému tréningu, alebo naopak, k pretrénovaniu regresií. Zaujímavé je aj zistenie, že práve v tomto bode výraznejšie rastie uspešnosť GhostEdgeNL aj Gossip algoritmu. Náš Gossip algoritmus práve pri datasete CORA skončil najhoršie a bol s jednou výnimkou vždy prekonaný. Nakoľko je u datasetu CORA pomerne malý priemerný počet susedov, dosiahli sme na ňom najvyššie hodnoty diaľky, do ktorej sa gossip vysielal. V prípade veľmi riedko ohodnotených grafov to bolo až do diaľky siedmich uzlov. U ostatných datasetoch nebolo potrebné odosielať gossipy na tak veľkú vzdialenosť pretože to boli jednak menšie datasety a tiež bol väčší priemerný počet susedov. Kompletné výsledky uvádzame v prílohe C.

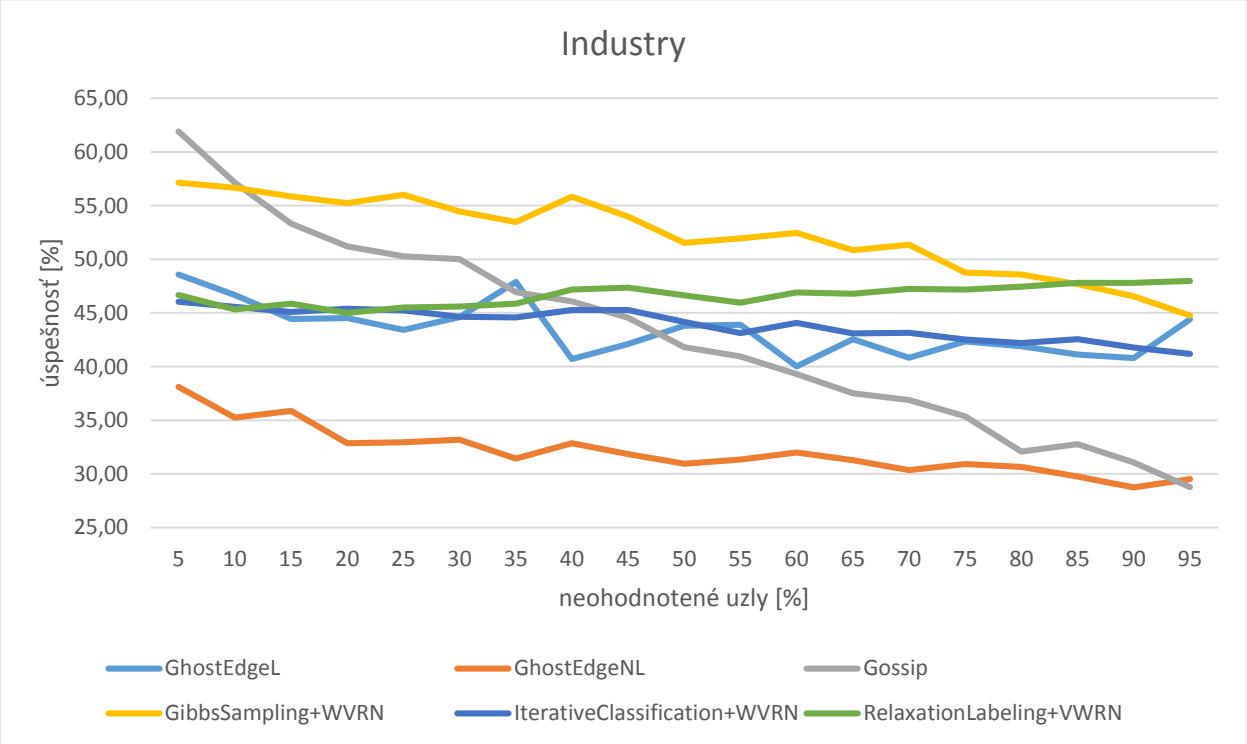
Dataset IMDB



GRAF 5

Dataset IMDB sa od ostatných najvýraznejšie odlišoval v hodnotách váh hrán. Drvivá väčšina hrán mala minimálne ohodnotenie (1) a bol tu tiež najmenší rozdiel medzi minimálnou a maximálnou váhou. IMDB je jediný dataset klasifikujúci uzly len do dvoch tried a je tu najvyšší počet uzlov bez jediného suseda – takmer 19 percent uzlov je bez susedov. Tento fakt výrazne znevýhodňuje algoritmy patriace do collective inference i náš Gossip algoritmus, nakoľko uzly bez susedov nie je možné týmito algoritmami ohodnotiť. Najúspešnejšími boli teda dva GhostEdge algoritmy, rozdiel medzi nimi je takmer vždy len v desatinách. Keďže klasifikujeme len do dvoch tried, GhostEdgeL nedokázal získať výhodu nad GhostEdge algoritmom bez učenia. U ostatných datasetoch, kde je tried viac, je spravidla GhostEdge algoritmus s učením úspešnejší. Collective inference algoritmy a náš Gossip algoritmus v prípade tohto datasetu boli približne na rovnakej úrovni. Smerom k 95 percentnému pomeru neohodnotených uzlov je pozorovateľné mierne zaostanie algoritmov Gossip a Gibbs Sampling. Kompletné výsledky uvádzame v prílohe D.

Dataset Industry

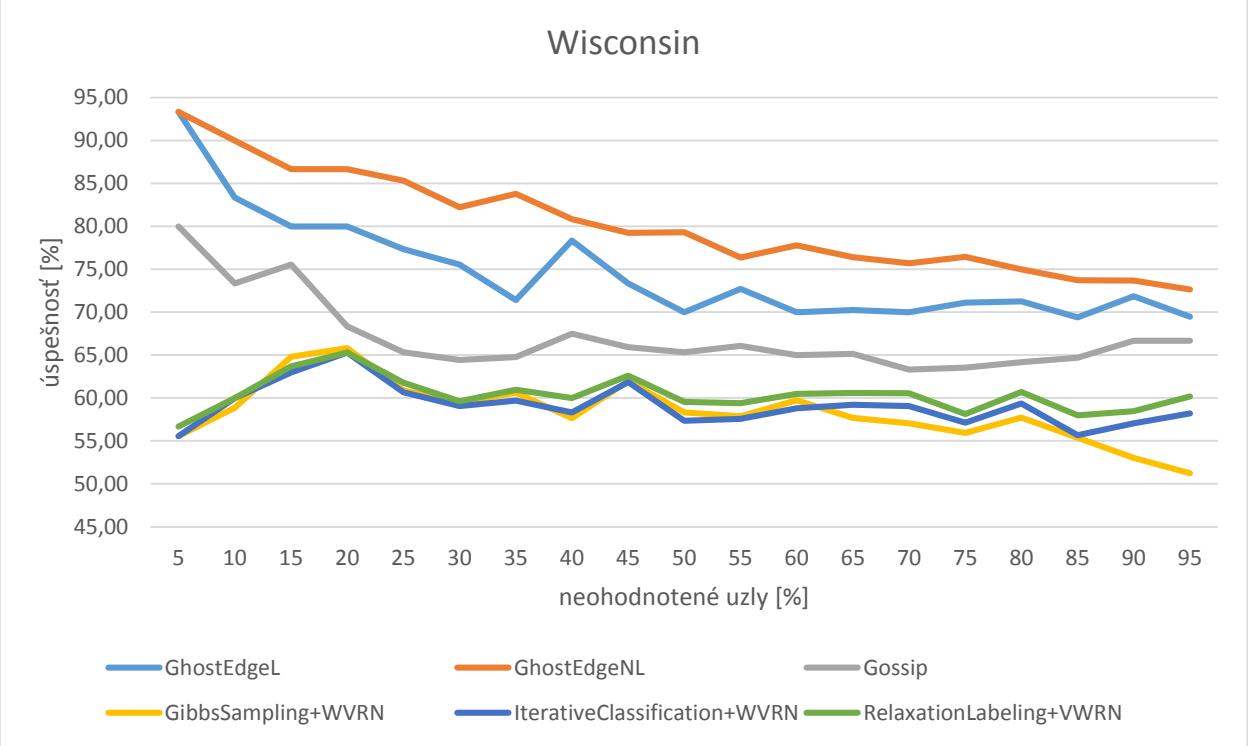


GRAF 6

Dataset Industry má najvyšší počet tried, najvyšší priemer váh hrán a druhý najvyšší počet uzlov. Na druhej strane má však najnižší priemerný počet susedov a tiež najnižšiu homofíliu.

Je to teda najviac extrémny dataset kde sme predpokladali, že by tu mohol byť najúspešnejší GhostEdgeL algoritmus. Nakoniec sa ale ukázal najúspešnejší algoritmus Gibbs Sampling a práve na tomto datasete boli najvýraznejšie rozdiely i v rámci collective inference algoritmov. Kým na iných datasetoch sa ich úspešnosť líšila často len v desatinách, tu bol až do 85 percent jasne najlepší Gibbs Sampling, pri najredších grafoch prevzal vedenie algoritmus Relaxation Labeling. Pri tomto datasete bol najväčší rozdiel aj medzi GhostEdge algoritmami – ako sme už spomínali, čím je vyšší počet tried, tým má viac navrch GhostEdge algoritmus s učením. Náš Gossip algoritmus tu jediný krát dokázal poraziť všetky ostatné algoritmy, avšak jeho úspešnosť začala klesať veľmi rýchlo až nakoniec ostal posledný pri najredšie ohodnotenom grafe. Prejavilo sa tu , že pri malom počte susedov Gossip algoritmus nefungoval dobre – i pri datasete CORA, ktorý mal druhý najnižší priemerný počet susedov, bol Gossip algoritmus najmenej úspešný. Za zmienku stojí ešte fakt, že Relaxation Labeling tu, i keď len o malú hodnotu, postupne zvyšoval svoju úspešnosť, čo súvisí s tým, že čím je vyšší počet neohodnotených uzlov, tým väčší je priestor na využitie tzv. simulovaného žíhania, kedy graf sa snaží dostať do svojho globálneho minima – čo v tomto prípade znamená maximálny počet správne ohodnotených uzlov. Kompletné výsledky uvádzame v prílohe E.

Dataset Wisconsin



GRAF 7

Dataset Wisconsin je počtom uzlov najmenší, ale naopak, počtom priemerného počtu susedov vysoko prevyšuje všetky ostatné datasety. Dva GhostEdge algoritmy prevýšili ostatné metódy a keďže klasifikujeme do šiestich tried, je vidieť vyššiu úspešnosť GhostEdge algoritmu s učením. Čo sa týka collective inference algoritmov je toto dataset, podobne ako IMDB, kde sú ich výsledky veľmi podobné a výraznejší náskok má až Relaxation Labeling pri veľmi riedko ohodnotených grafoch. Potvrdil tým, čo bolo vidno i pri ostatných datasetoch, že je pri riedko ohodnotených grafoch vždy najúspešnejší spomedzi collective inference metódach. Pri tomto datasete sa objavil i zaujímavý paradox collective inference algoritmov – lepšie dokážu klasifikovať neohodnotené uzly v riedko ohodnotených grafoch ako v husto ohodnotených grafoch. Pri 5 percentom podiele neohodnotených uzlov majú algoritmy Iterative Classification aj Relaxation Labeling najhoršie výsledky. Je to spôsobené veľkým počtom susedov a nemožnosťou využiť výhody simulovaného žíhania. Gossip algoritmus síce nedokázal poraziť GhostEdge algoritmy, ale celkom jasne je pred collective inference metódami. Je to jednak vďaka veľkému počtu hrán, ale zrejme pozitívny vplyv má aj vysoké priemerné ohodnotenie hrán. Tento fakt dokáže Gossip algoritmus pretaviť do zvýhodnenia vysoko ohodnotených hrán v grafe kade gossipy potečú s vyššou váhou ako cez ostatné váhy. Iste, váhy hrán berú do úvahy aj collective

inference algoritmy, avšak tie v každej jednej iterácii prepočítajú ohodnotenie každého neohodnoteného uzla. Informácia od zdrojového k cieľovému uzlu je tak pozmenená (pokiaľ medzi nimi je aspoň jeden uzol). Gossip algoritmus v jednej iterácii rozpošle všetky gossipy a smerujú od zdroja priamo k cieľu, nikde počas cesty svoju hodnotu nemenia a tým pádom výsledné ohodnotenie je iné a práve na tomto datasete je vidno, že takýmto spôsobom možno dosiahnuť lepšie výsledky. Kompletné výsledky uvádzame v kapitole F.

6 Záver

V našej práci o problematike relačnej klasifikácie sme sa na úvod vyjadrili k všeobecnému poňatiu pojmu klasifikácie v informatike. Následne sme sa vyjadrili i k predchodcovi relačnej klasifikácii – atribútovej klasifikácii. Tu sme aspoň stručne charakterizovali najpoužívanejšie atribútové klasifikátory.

Najväčší priestor sme venovali relačnej klasifikácii – popísali sme rozdiel medzi atribútovo orientovanou a relačne orientovanou klasifikáciou, v prehľade relačných metód sme uviedli tie najznámejšie, rovnako tak sme predstavili najznámejšie metódy v collective inference, opísali a priblížili sme ako funguje relačne orientovaný klasifikačný systém – jeho rozdelenie na 3 hlavné časti a prepojenia medzi nimi.

Venovali sme sa aj špeciálnemu typu grafov – riedko ohodnoteným grafom, kde bežné techniky relačnej klasifikácie zlyhávajú a podrobne sme popísali dva relačné klasifikátory techniky Ghost Edge na riedko ohodnotené grafy.

Hlavnou časťou práce je popis nášho algoritmu – myšlienky, ktorá nás naštartovala a cesty, ktorou sme sa dopracovali k výslednému algoritmu.

Nakoniec sme predstavili výsledky a namerané hodnoty počas testovania. Našu metódu sme porovnali s inými algoritmami, ktoré sme v práci popísali a odôvodnili sme výsledky. Nepodarilo sa nám prekonať úspešnosť existujúcich riešení, avšak dokázali sme sa im priblížiť a v prípade niektorých datasetov sme klasifikovali s vyššou úspešnosťou ako niektoré algoritmy.

Do budúcnosti by preto malo zmysel zapodievať sa ďalšími úpravami algoritmu.

Jednou z ciest by bolo zakomponovanie iteratívnej metódy podobne, ako je tomu v prípade collective inference algoritmoch. Otvárajú sa tak nové možnosti ako rozposielané gossipy využiť – ich vynulovanie pred každou iteráciou, alebo ich kumulovanie do konca a klasifikovať zo všetkých gossipov rozposlaných počas všetkých iterácií. Zaujímavé výsledky by mohlo priniesť i zakomponovanie prvkov simulovaného žíhania a regulovať tak váhu gossipov i počtom iterácií – kým v prvých iteráciách by sme dovolili putovať skrz graf gossipom s vysokou váhou, postupne by sme ich váhy znižovali.

Druhou cestou by bolo využitie niektorej z techník učenia s učiteľom. Vhodným roztriedením gossipov by sme vyrobili atribúty, ktoré by slúžili ako vstup učiacemu algoritmu. Takto postavený algoritmus by mohol byť robustnejší voči prítomnosti homofílie alebo iných javov v dátach.

7 Literatúra

MACSKASSY, S.A. – PROVOST, F. Classification in Networked Data: A Toolkit and a Univariate Case Study. In *Journal of Machine Learning Research 8*, 2007, p: 935-983.

GALLAGHER, B. –Tong, H. – ELIASSI-RAD, T. – Faloutsos, Ch. Using ghost edges for classification in sparsely labeled networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2008, p: 256-264.

VOJTEK, P. *Príspevok k relačnej klasifikácii s využitím predpokladu homofílie* : dizertačná práca. Bratislava : Slovenská technická univerzita, 2010. 135 s.

KATONA, A. Relačná klasifikácia : seminárna práca. Bratislava : Slovenská technická univerzita, 2008. 10 s.

8 Prílohy

A

Porovnanie jednotlivých funkcií prezentovaných v kapitole 4.3.1.1 Funkcie pre určenie váhy ohodnotených uzlov.

% neohodnotených uzlov	FUNKCIA 1	FUNKCIA 2	FUNKCIA 3	FUNKCIA 4	FUNKCIA 5
5	65,71	66,67	74,29	67,62	75,71
10	65,71	66,67	71,67	67,62	75,24
15	65,71	65,40	73,02	66,19	75,40
20	64,40	64,76	71,90	65,12	74,29
25	64,67	64,29	70,67	65,90	73,62
30	64,44	64,52	71,11	65,63	72,78
35	62,93	63,27	69,80	64,22	72,65
40	62,44	61,96	68,75	62,86	71,79
45	63,86	64,44	69,05	65,03	70,95
50	59,90	60,10	67,38	61,81	70,29
55	60,43	60,39	66,02	61,47	71,95
60	61,31	61,47	67,30	61,90	70,52
65	55,86	55,71	61,25	56,34	69,93
70	54,83	54,83	63,33	55,03	67,45
75	54,57	54,83	60,38	55,08	64,70
80	48,87	49,35	57,65	49,43	67,20
85	41,37	41,34	52,61	41,54	65,35
90	38,78	38,94	49,97	38,57	65,77
95	22,98	22,93	37,52	22,88	68,52

B

Porovnanie jednotlivých prístupov pre určenie váhy gossipu prezentovaných v kapitole 4.3.2 Určenie váhy gossipu.

% neohodnotených uzlov	VZOREC (1)	VZOREC (2)	VZOREC (3)	MAXIMUM
5	67,14	65,71	72,14	75,71
10	62,86	57,14	69,03	71,43
15	64,29	56,19	67,43	68,57
20	62,50	52,50	65,57	67,86
25	63,14	54,29	64,29	67,43
30	63,57	56,19	65,05	69,76
35	63,06	53,88	63,94	66,94
40	63,39	51,79	63,21	66,79
45	63,65	52,38	63,35	66,03
50	64,00	53,00	64,29	67,14
55	64,16	53,64	63,36	65,71
60	62,26	52,62	63,31	65,12
65	61,98	55,38	62,49	65,05
70	61,94	50,51	62,71	65,20
75	61,90	51,05	61,71	64,19
80	60,89	52,32	60,75	63,66
85	60,92	49,33	60,29	63,11
90	61,27	47,70	61,27	63,10
95	61,58	47,67	60,38	62,86

C

Namerané výsledky algoritmov relačnej klasifikácie na datasete CORA.

% neohodnotených uzlov	GhostEdgeL	GhostEdgeNL	Gossip	GIBBS+WV	IC+WV	RL+VW
5	80,48	80,00	75,71	80,95	82,38	82,38
10	82,38	77,14	75,24	79,76	80,48	81,43
15	80,16	77,14	75,40	80,95	80,79	81,59
20	81,07	75,48	74,29	78,81	80,12	80,71
25	80,95	75,71	73,62	80,38	80,76	80,38
30	80,63	74,84	72,78	79,21	79,76	80,32
35	81,22	75,03	72,65	80,20	81,70	81,50
40	80,65	74,58	71,79	78,81	79,52	80,54
45	80,79	74,29	70,95	79,52	80,26	80,79
50	80,24	73,57	70,29	78,71	80,00	81,19
55	80,56	73,42	71,95	78,31	80,52	81,00
60	79,72	73,06	70,52	77,42	79,05	79,96
65	81,03	73,08	69,93	77,11	79,45	80,11
70	78,50	72,04	67,45	76,84	79,12	80,61
75	79,65	70,86	64,70	77,05	79,08	80,60
80	69,20	71,49	67,20	75,98	78,72	80,33
85	69,92	68,82	65,35	75,55	78,74	80,36
90	70,98	68,02	65,77	74,97	78,65	80,66
95	76,19	64,06	68,52	73,91	78,52	80,50

D

Namerané výsledky algoritmov relačnej klasifikácie na datasete IMDB.

% neohodnotených uzlov	GhostEdgeL	GhostEdgeNL	Gossip	GIBBS+WV	IC+WV	RL+VW
5	90,00	88,57	75,71	74,29	72,86	72,86
10	87,86	89,29	71,43	70,71	70,71	70,71
15	84,76	87,62	68,57	70,95	71,43	71,90
20	85,00	85,71	67,86	66,43	66,79	67,50
25	84,00	83,43	67,43	67,43	68,00	68,57
30	84,29	83,57	69,76	69,52	69,52	69,29
35	83,47	83,88	66,94	69,80	69,18	69,59
40	82,50	83,04	66,79	67,86	68,21	68,93
45	82,06	82,22	66,03	67,14	67,94	67,94
50	82,71	83,29	67,14	67,43	68,00	69,00
55	82,08	82,73	65,71	65,97	68,96	69,09
60	81,67	82,38	65,12	66,31	67,74	67,38
65	81,65	82,86	65,05	65,16	65,93	66,70
70	81,22	81,12	65,20	65,00	66,63	67,35
75	81,24	81,62	64,19	65,62	67,33	67,43
80	80,89	81,43	63,66	66,25	67,50	68,21
85	81,43	80,42	63,11	64,29	66,89	67,39
90	80,63	79,60	63,10	63,02	66,98	67,06
95	79,25	79,10	62,86	63,46	66,62	67,44

E

Namerané výsledky algoritmov relačnej klasifikácie na datasete Industry.

% neohodnotených uzlov	GhostEdgeL	GhostEdgeNL	Gossip	GIBBS+WV	IC+WV	RL+VW
5	48,57	38,10	61,90	57,14	46,03	46,67
10	46,67	35,24	57,14	56,67	45,56	45,32
15	44,44	35,87	53,33	55,87	45,08	45,87
20	44,52	32,86	51,19	55,24	45,40	45,00
25	43,43	32,95	50,29	56,00	45,24	45,49
30	44,60	33,17	50,00	54,44	44,63	45,58
35	47,89	31,43	46,94	53,47	44,58	45,87
40	40,71	32,86	46,07	55,83	45,28	47,18
45	42,12	31,85	44,55	53,97	45,26	47,35
50	43,81	30,95	41,81	51,52	44,17	46,65
55	43,90	31,34	40,95	51,95	43,13	45,95
60	40,00	31,98	39,29	52,46	44,09	46,90
65	42,56	31,28	37,51	50,84	43,08	46,80
70	40,82	30,34	36,87	51,36	43,14	47,24
75	42,35	30,92	35,37	48,76	42,51	47,16
80	41,90	30,65	32,08	48,57	42,19	47,44
85	41,12	29,75	32,77	47,68	42,54	47,82
90	40,79	28,73	31,06	46,56	41,76	47,81
95	44,41	29,52	28,77	44,76	41,17	47,99

F

Namerané výsledky algoritmov relačnej klasifikácie na datasete Wisconsin.

% neohodnotených uzlov	GhostEdgeL	GhostEdgeNL	Gossip	GIBBS+WV	IC+WV	RL+VW
5	93,33	93,33	80,00	55,56	55,56	56,67
10	83,33	90,00	73,33	58,89	60,00	60,00
15	80,00	86,67	75,56	64,81	62,96	63,70
20	80,00	86,67	68,33	65,83	65,28	65,28
25	77,33	85,33	65,33	60,89	60,67	61,78
30	75,56	82,22	64,44	59,44	59,07	59,63
35	71,43	83,81	64,76	60,63	59,68	60,95
40	78,33	80,83	67,50	57,64	58,33	60,00
45	73,33	79,26	65,93	61,98	61,85	62,59
50	70,00	79,33	65,33	58,33	57,33	59,56
55	72,73	76,36	66,06	57,88	57,58	59,39
60	70,00	77,78	65,00	59,72	58,80	60,46
65	70,26	76,41	65,13	57,69	59,23	60,60
70	70,00	75,71	63,33	57,06	59,05	60,56
75	71,11	76,44	63,56	55,93	57,11	58,15
80	71,25	75,00	64,17	57,71	59,38	60,69
85	69,41	73,73	64,71	55,36	55,69	57,97
90	71,85	73,70	66,67	53,02	57,04	58,46
95	69,47	72,63	66,67	51,23	58,19	60,18