Semestrální práce z KIV/UIR

Klasifikace dokumentů

Celková doba vypracování: 40 hodin

Vypracoval: Michal Fiala

Studentské číslo: A15B0028P

Studentský email: michalf@students.zcu.cz

Obsah

1. Zadání problému

Ve zvoleném programovacím jazyce navrhněte a implementujte program, který umožní klasfikovat textové dokumenty do tříd podle jejich obsahu, např. počasí,sport,politika, apod. Pří řešení budou splněny následující podmínky:

- použijte korpus dokumentů v českém jazyce, který je k dispozici na http://home. zcu.cz/~pkral/sw/ (uvažujte pouze první třídu dokumentu podle názvu, tedy např. dokument 05857_zdr_ptr_eur.txt náleží do třídy ” zdr“ - zdravotnictví.)

- implementujte alespoň tři různé algoritmy (z přednášek i vlastní) pro tvorbu příznaků reprezentující textový dokument

- implementujte alespoň dva různé klasifikační algoritmy (klasifikace s učitelem):

    - Naivní Bayesův klasifikátor
    - klasifikátor dle vlastní volby

- funkčnost programu bude následující:

    - spuštění s parametry: trénovací_množina, testovací_množina, použije zadaný parametrizační/klasifikační algoritmus, zároveň vyhodnotí úspěšnost klasifikace a natrénovaný model uloží do souboru pro pozdější použití (např. s GUI).

    - spuštění s jedním parametrem = název_modelu : program se spustí s jednoduchým GUI a uloženým klasifikačním modelem. Program umožní klasifikovat dokumenty napsané v GUI pomocí klávesnice (resp.překopírované ze schránky).

- ohodnoťte kvalitu klasifikátoru na dodaných datech, použijte metriku přesnost (accuracy). a uloženým klasifikačním modelem. Otestujte všechny konfigurace klasifikátorů (tedy celkem 6 výsledků).

2. Analýza problému

Klasifikace textových dokumentů je velmi složitý problém. Tento problém je založený na strojovém učení. Abychom mohli textové dokumenty klasifikovat, tedy zařadit je do určitých kategorií, je potřeba daný klasifikační algoritmus (klasifikace s učitelem) natrénovat na trénovacích datech, kterým jsou přiřazeny příznakové vektory. Abychom vůbec mohli trénovacím datům přiřadit příznakové vektory, je třeba vhodným způsobem vyfiltrovat nežádoucí elementy v textu (například tečky, čárky apod.).

2.1 Výběr příznaků

Texty sloužící jako zdrojová data pro klasifikátory jsou obvykle reprezentovány jako vektory příznaků. Příznaky zde obvykle zastupují slova a jejich ohodnocení příznakovým algoritmem. Existuje velké množství příznakových algoritmů, ale ne všechny nejsou úspěšné.

3. Návrh řešení

3.1 Parametrizace textu

Textový dokument načteme, vezmeme pouze vyfiltrovaná slova a ty uložíme do jeho množiny slov. Tyto slova potom ohodnotíme příznakem a vložíme do příznakového vektoru slov.

3.2 Příznaky

Vzhledem k volbě parametrizace textu můžeme zvolit různé typy příznaků. Typickým je binární příznakový vektor, který vezme pouze všechna unikátní slova z textu a ohodnotí je 1, pokud se slovo v textu nenachází, ohodnotíme ho 0. Dalším je například frekvenční příznakový vektor, který vezme pouze všechna unikátní slova z textu a ohodnotí je počtem jeho výskytů v daném textu.

3.3 Klasifikátory

Jako klasifikátory volíme Naivní Bayesův klasifikátor a k-NN (Nearest-Neighbour).

4. Popis řešení

4.1 Parametrizace

Textové soubory jsou načteny pomocí třídy *ReadAllFiles*, která vrací množinu všech souborů *File*. Tato množina je poté vložena do třídy *DocumentCreator*, která vytvoří reprezentaci každého textového dokumentu. Tato reprezentace obsahuje název souboru, vyfiltrovaný text souboru a jeho správnou kategorii.

4.2 Příznaky

Každý ze zvolených příznaků vytvoří každé reprezentaci textových dokumentů příznakový vektor.
Zvolené příznaky:

- BinaryWordCounter
    - Používá pouze binární reprezentaci dokumentu. Zda se unikátní slovo v textu nachází
      (1) ano nebo ne (0).

- SimpleWordCounter
    - Vypočítá pro každé unikátní slovo počet výskytu v dokumentu.

- LengthWordCounter
    - Počítá výskyt všech obsažených délek slov dokumentu.

4.3 Klasifikátory

4.3.1 Naivní Bayes

Klasifikátor je vytvořen z *ArrayList<PropabilityTable>,* kde *ProbabilityTable* reprezentuje *HashMapu<String, Double>,* ve které je vždy uloženo slovo a k němu přiřazena pravděpodobnost s jakou se v textu dané třídy nachází, tato třída slouží jako přepravka. HashMapu pro výskyt slov v celé třídě získáme spojením HashMap jednotlivých dokumentů se stejnou kategorií. K tomuto procesu nám slouží třída ClassRepre. Vzhledem k problémům s přesností pro hodnoty double byl implementován Bayes s logaritmickou reprezentací.

$$logP(Ci|D) = logP(Ci) + logP(D|Ci)$$

4.3.2 Nearest Neighbour (1-NN)

Klasifikátor je tvořen vstupními množinami a to jsou natrénovaná množina a testovací množina. Poté prochází množinu všech testovacích dat a porovnává ji s množinou trénovacích dat pomocí euklidovské vzdálenosti, tyto výsledky ukládá do *HashMap<Document, Double>,* pro každý testovací soubor, kde *Document* je odkaz na dokument, ke kterému je vzdálenost Double.

5. Uživatelská dokumentace

Pro spuštění aplikace je nutné mít nainstalovanou Javu 1.8 a správné nastavení systémových cest k jejím funkcím. Aplikace se dále spouští v příkazovém řádku.

5.2.1 Trénování klasifikátoru

Tento mód slouží k natrénování klasifikátoru se zvolenými příznaky, jeho následné otestování a následné uložení již natrénovaného klasifikátoru.

Mód spustíme voláním příkazu:

java –jar Seminární_práce.jar <trénovací_množina> <testovací_množina> <příznaky> <klasifikátor> <název_složky_export>

- Trénovací množina
    - Je možné zadat cestu ke složce, nebo název složky pokud se složka nachází ve stejném adresáři jako samotná aplikace

- Testovací množina
    - Je možné zadat cestu ke složce, nebo název složky pokud se složka nachází ve stejném adresáři jako samotná aplikace

- Příznaky
    - binary
        - binární reprezentace
    - simple
        - frekvenční reprezentace
    - length
        - délková reprezentace
- Klasifikátor

    - bayes
        - Naivní Bayes
    - k-NN
        - po zavolání tohoto příkazu vyčkejte a zadejte číslo k po žádosti programu
        - Nearest Neighbour

- Název souboru export

    - jméno pro složku, kam se exportuje klasifikátor

5.2.2 Volná klasifikace

V tomto módu se spustí jednoduché GUI s možností vepsat, či vkopírovat text a následně ho nechat klasifikovat importovaným klasifikátorem.

Mód spustíme voláním příkazu:

java –jar Seminární_práce.jar <název_složky_import>

- Jméno souboru import
    - Jméno složky s uloženým klasifikátorem.

6. Závěr

6.1 Výsledky

6.1.1 Navní Bayes

Niavní Bayes s dostatečnou trénovací množinou (10500 článků) dosahuje přesností velmi kvalitních výsledků, kromě délkového příznaku.

- Příznaky
    - binární
        - 87,1%
    - frekvenční
        - 88,7%
    - délkový
        - 14,5%

6.1.2 K-NN

Zvolený klasifikátor 3-NN. Tento klasifikátor měl při testování mnohem menší přesnost než Naivní-Bayes.

- Příznaky
    - binární
        - 67%
    - frekvenční
        - 66%
    - délkový
        - 41%

U tohoto klasifikátoru si můžeme všimnout, že s délkovým příznakem tato klasifikace spolupracuje lépe než předchozí.

6.2 Shrnutí

Naivní Bayes má pro klasifikaci dokumentů znatelně větší úspěšnost, navíc jeho klasifikování je výpočetně jednodušší, tudíž se jeví jako vhodná varianta. Avšak testováním jsem dospěl názoru, že příznaková metoda, která počítá délky slov není vhodná pro klasifikaci.

aplikaci implementována, ale z důvodu menší přesnosti při některých příznacích nebyla ve finální práci použita.