Západočeská univerzita

Fakulta aplikovaných věd

Informatika

Semestrální práce z předmětu KIV/UIR

David Markov

A18B0262P

markovd@students.zcu.cz

Obsah

1 Zadání

KIV/UIR - Semestrální práce pro ak. rok 2019/20

Klasifikace dokumentů

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

- Použijte data z českého historického periodika „Posel od Čerchova“, která jsou k dispozici na https://drive.google.com/drive/folders/1mQbBNS43gWFRMHDYdSkQug47cuhPTsHJ? usp=sharing. V původní podobě jsou data k dispozici na http://www.portafontium. eu/periodical/posel-od-cerchova-1872?language=cs.
- Pro vyhodnocení přesnosti implementovany´ch algoritmů bude NUTNE vybrané dokumenty ručně označkovat. Každý´ student ručně anotuje 10 stran zadaného textu – *termín 31. 3. 2020*. Za dodržení termínu obdrží student **bonus 10b**.
- Přiřazení konkrétních textů jednotlivým studentům spolu s návodem na anotaci a příklady je uloženo spolu s daty na výše uvedené adrese, konkrétně:
  - **0** - vzorová složka (takhle by měl výsledek vypadat)
  - **1, 2, .. , 15, 101, 102, ..** - data k anotaci
  - **prirazeni_souboru_studentum.xlsx** - určení, jaké soubory má jaký student anotovat. Až budete mít anotaci hotovou, doplňte sem informaci.
  - **Anotační příručka** - návod, jak články anotovat.
  - **Klasifikace dokumentů - kategorie.xlsx** - seznam kategorií k anotaci s příklady.
  - **sem prace20.pdf** - Zadaní semestrální práce
- 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:
    název klasifikátoru
    soubor_se_seznamem_klasifikačních_tříd,
    trénovací_množina, testovací_množina,
    parametrizační_algoritmus, klasifikační_algoritmus,
    název_modelu

program natrénuje klasifikátor na dané trénovací množině, použije zadaný parametrizační a klasifikační algoritmus, zároveň vyhodnotí úspešnost klasifikace a natrénovaný´ model uloží do souboru pro pozdější použití (např. s GUI).

  - spuštění s jedním parametrem:
- **název klasifikátoru**
  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), kde jako správnou klasifikaci uvažujte takovou, kde se klasifikovaná třída nachází mezi anotovanými. Otestujte všechny konfigurace klasifikátoru (tedy celkem 6 výsledků).

Poznámky:

- pro vlastní implementaci není potřeba čekat na dokončení anotace. Pro průběžné testování můžete použít korpus současné češtiny, který je k dispozici na http://ctdc.kiv.zcu.cz/ (uvažujte pouze první třídu dokumentu podle názvu, tedy např. dokument_05857_zdr_ptr_eur.txt náleží do třídy „zdr“ - zdravotnictví).

- další informace, např. dokumentace nebo forma odevzdávání jsou k dispozici na CW pod záložkou *Samostatná práce*.

Bonusové úkoly:

- vyzkoušejte již nějakou hotovou implementaci klasifikátoru (scikit-learn, Weka, apod.) a výsledky srovnejte s Vaší implementací [až 10b navíc].

- vyzkoušejte shlukování (klasifikaci bez učitele, např. k-means) a výsledky porovnejte s výsledky klasifikace s učitelem [až 10b navíc].

- implementujte navíc klasifikační algoritmus založený na neuronové síti typu MLP s využitím knihoven Keras a Tensorflow [až 10b navíc].

- vyzkoušejte klasifikaci anglických dokumentů, korpus Reuters je k dispozici na adrese http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html [až 20b navíc].

2 Analýza problému

V rámci zadání jsou k dispozici dvě složky – jedna s trénovacími dokumenty a jedna s dokumenty testovacími. Dále je dán soubor se seznamem dostupných klasifikačních tříd. Dokumenty jsou již anotovány a mají následující formát:

- **první neprázdný řádek textu:** seznam klasifikačních tříd, do kterých dokument patří
- **druhý neprázdný řádek textu:** začátek textového obsahu dokumentu

Text je nutno převézt do nějaké reprezentativní podoby, která nám bude vyhovovat pro naše potřeby klasifikace. Ideálně potřebujeme spočítat „hodnotu“ každého slova pro daný dokument nebo klasifikační třídu. Existuje mnoho parametrizačních algoritmů, např. *Dokumentová frekvence*, *TF-IDF*, *Mutual Information*, *Náhodný výběr* nebo *Binární reprezentace*. Pro naše účely vybereme Binární reprezentaci, Dokumentovou frekvenci a TF-IDF, převážně z hlediska jednoduchosti implementace.

Dále je třeba vybrat vhodný klasifikační algoritmus, který natrénujeme pomocí vypočtených hodnot slov z parametrizačního algoritmu. Variantami jsou např. *Naivní Bayesův klasifikátor*, *Metoda podpůrných vektorů*, *rozhodovací stromy*, nebo *k - nejbližších sousedů*. Ze zadání plyne, že je nutno implementovat Naivní Bayesův klasifikátor, a jako druhý k němu přidáme klasifikátor k - nejbližších sousedů.

3 Návrh řešení

3.1 Parametrizační algoritmy

3.1.1 Binární reprezentace

Tento algoritmus pracuje na jednoduchém principu. Projde všechna unikátní slova v trénovací množině dokumentů a nastaví mu binární hodnotu 1, pokud se slovo v daném dokumentu nachází, nebo 0, pokud se slovo v dokumentu nevyskytuje.

3.1.2 Dokumentová frekvence

Algoritmus dokumentové frekvence hledá celkový počet výskytů daného slova v daném dokumentu a ten pak normalizuje počtem slov v dokumentu (viz. Obr. 1).

$$TF_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}}$$

- $n_{i,j}$ ... počet výskytů slova t_i v dokumentu d_j
- Dělitel ... součet počtu výskytů všech slov v dokumentu d_j (délka)

Obrázek 1: Hodnota slova v algoritmu dokumentové frekvence. Zdroj: https://home.zcu.cz/~pkral/uir/pr6-materialy/uir6-doplnujici-informace.pdf

3.1.3 TF-IDF

Tento algoritmus, zvaný *„Term frequency – inverse document frequency“* využívá dříve zmíněného algoritmu dokumentové frekvence a upravuje ho o inverzní dokumentovou frekvenci (viz. Obr. 2). Spočítá navíc četnost výskytu daného slova v celé trénovací množině. Proto tento algoritmus lépe reprezentuje důležitost slova. Pokud se slovo ve všech dokumentech vyskytuje často, není velmi důležité. Hodnotu slova *TF-IDF* pak vyjádříme jako: *TF * IDF*.

$$IDF_i = \log \frac{|D|}{|\{j : t_i \in d_j\}|}$$

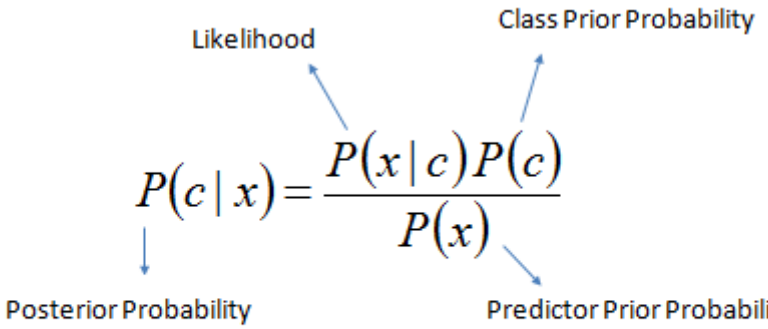
- $|D|$ ... počet všech dokumentů
- $|\{j : t_i \in d_j\}|$ ... počet dokumentů, kde se vyskytuje slovo t_i.

Obrázek 2: Inverzní dokumentová frekvence. Zdroj: https://home.zcu.cz/~pkral/uir/pr6-materialy/uir6-doplnujici-informace.pdf

3.2 Klasifikační algoritmy

3.2.1 Naivní Bayesův klasifikátor

Bayesův klasifikátor je jedním z pravděpodobnostních klasifikačních algoritmů, tedy počítá, s jakou pravděpodobností náš dokument patří do které klasifikační třídy a jako výsledek vybere třídu s nejvyšší pravděpodobností (Obr. 3). Naivita tohoto algoritmu spočívá v tom, že předpokládáme nezávislost jednotlivých prvků, v našem případě slov. Můžeme tedy počítat pravděpodobnost příslušnosti každého slova ke klasifikační třídě.



$$P(c \mid \mathrm{X}) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

Obrázek 3: Výpočet pravděpodobnosti Naivním Bayesovým klasifikátorem. Zdroj: https://www.saedsayad.com/naive_bayesian.htm

V praxi tedy násobíme pravděpodobnost `P(c)` příslušnosti jakéhokoliv dokumentu do třídy `c` s pravděpodobnostmi $P(x_i|c)$ příslušnosti každého slova v dokumentu do třídy `c`. Tyto pravděpodobnosti lze vyjádřit jako:

$$P(slovo|třída) = \frac{(počet\_výskt\text{ů}\_slova\_ve\_tříd\check{e} + 1)}{(počet\_slov\_ve\_tříd\check{e} + počet\_různých\_slov\_v\_trénovací\_množin\check{e})}$$

Výraz v tomto tvaru platí pro binární reprezentaci textu. Pro dokumentovou frekvenci a *TF-IDF* je třeba vyjádřit:

$$počet\_výskyt\text{ů}\_slova\_ve\_tříd\check{e} = součet\ hodnot\ TF(nebo\ TF - IDF)\ slova\ ve\ všech\ dokumentech\ patřících\ do\ dané\ třídy$$

$$počet\_slov\_ve\_tříd\check{e} = součet\ hodnot\ TF(nebo\ TF - IDF)\ všech\ slov\ v\ této\ tříd\check{e}$$

3.2.2 K – nejbližších sousedů

Tento algoritmus uvažuje jednotlivé dokumenty jako body v dvojrozměrném souřadném systému. Na základě jejich „polohy“ spočítá euklidovskou vzdálenost (Obr. 4) klasifikovaného dokumentu s každým trénovacím dokumentem. Podle toho pak vybere klasifikační třídu, ke které patří největší počet dokumentů z k nejbližších dokumentů.

$$dist(i,j) = \sqrt{(D_i - D_j)^2}$$

Obrázek 4: Euklidovská vzdálenost dokumentu i a dokumentu j.

Rozdíl mezi dokumenty pod odmocninou lze vyjádřit jako rozdíl hodnot jejich společných slov. Pokud jsou dokumenty totožné, získáme nulovou vzdálenost.

4 Implementace

Program je kompletně napsán v jazyce `Java` verze `11. 0. 6`. Problém je dekomponován do 4 balíků:

- **app**: obsahuje hlavní spouštěcí třídu aplikace
- **feature**: obsahuje logiku spojenou s parametrizačními algoritmy
- **classifier**: obsahuje logiku spojenou s klasifikátory
- **utils**: obsahuje pomocné objekty např. pro ukládání a načítání dat

4.1 App

4.1.1 DocumentClassifierApp

Třída je vstupním bodem celé aplikace. Přebírá parametry v metodě `main`, podle kterých rozhoduje, jestli se spustí načítání trénovacích dat a klasifikaci dat testovacích, nebo jestli dojde k načtení natrénovaného modelu a ke klasifikaci uživatelského vstupu přes `GUI`. Třída je zprostředkovatelem všech nutných operací pro běh programu.

4.2 Classifier

4.2.1 KNN

Implementace algoritmu k – nejbližších sousedů. Konstanta k je volena staticky jako atribut této třídy.

4.2.2 NaiveBayesClassifier

Reprezentace Naivního Bayesovského klasifikátoru.

4.3 Feature

4.3.1 TermBinary

Reprezentuje parametrizační algoritmus binárního výběru. Projde všechna unikátní slova a nastaví jejich binární hodnonu v každém dokumentu.

4.3.2 TermFrequency

Tato třída implementuje algoritmus dokumentové frekvence. Zjistí počet výskytů slova v dokumentu a normalizuje hodnotu celkovým počtem slov v dokumentu.

4.3.3 TFIDF

Reprezentace algoritmu *TF-IDF*. Využívá implementace dokumentové frekvence a obohacuje ji o inverzní dokumentovou frekvenci.

4.4 Utils

4.4.1 Document

Datový typ, který uchovává informace o načteném dokumentu.

4.4.2 FileLoader

Tato třída je zprostředkovatel všech operací spojených s načítáním dat ze souborů.

4.4.3 FileSaver

Třída je pravým opakem třídy `FileLoader`, je zprostředkovatelem operací ukládání dat do souborů.

4.4.4 Model

Reprezentuje natrénovaný model, který je možno uložit do souboru, nebo ho z něj načíst pro pozdější použití.

5 Uživatelská příručka

5.1 Spuštění

Program se spouští přes artefakt DocumentClassifier.jar z příkazové řádky příkazem:

$$java - jar\ DocumentClassifier.jar < vstupní\_parametry >$$

Pro spuštění trénování klasifikátoru na trénovací množině je třeba předat celkem 6 parametrů v tomto pořadí:

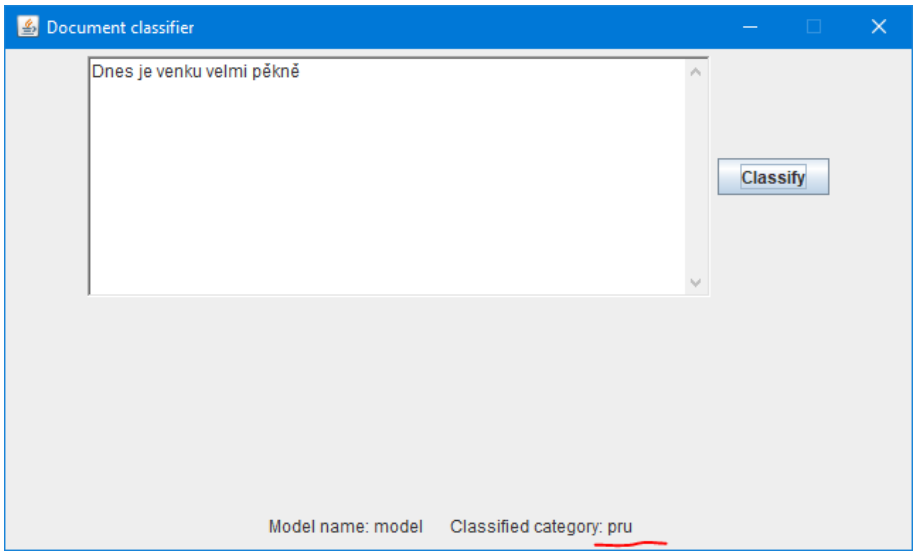
- cesta k souboru se seznamem klasifikačních tříd
- cesta ke složce s trénovacími dokumenty
- cesta ke složce s testovacími dokumenty
- název parametrizačního algoritmu
  - **binary** = binární reprezentace
  - **tf** = dokumentová frekvence
  - **tfidf** = dokumentová frekvence – inverzní dokumentová frekvence
- název klasifikátoru
  - **bayes** = Naivní Bayesův klasifikátor
  - **knn** = k – nejbližších sousedů
- název modelu, podle kterého bude model uložen s příponou .model

Například:

$$jar\ DocumentClassifier.jar\ classes.txt\ Train\ Test\ binary\ bayes\ binary\_bayes$$

Spuštění klasifikace uživatelského vstupu vyžaduje jako vstupní parametr název souboru s uloženým natrénovaným klasifikátorem, například:

$$jar\ DocumentClassifier.jar\ binary\_bayes$$



Obrázek 4: Uživatelské rozhraní pro klasifikaci textu. Výsledná klasifikační třída se zobrazí v dolní části rozhraní.

6 Závěr

6.1 Zhodnocení kvality klasifikátoru

V tabulce přesností (Obr. 5) můžeme vidět, že nejlépe si při klasifikaci vedl *Naivní Bayesův klasifikátor* natrénovaný pomocí algoritmu *TF – IDF*, s přsností přes 30%. Naopak nejhorších výsledků dosahoval klasifikátor k – nejbližších sousedů v kombinaci s *TF – IDF* při přesnosti 11,1%. Nízké přesnosti klasifikace jsou pravděpodobně způsobeny zanesením klasifikátorů přebytečnými slovy jako jsou předložky a spojky, které nemají žádnou informativní hodnotu, ale ovlivňují klasifikátor. Roli také hraje to, že trénovací dokumenty jsou tvořeny textem rozpoznaným pomocí *OCR* z historického periodika, tudíž se v textu mohou vyskytovat nesmyslné výrazy, nebo více výrazů spojených do jednoho.

	Binární reprezentace	Dokumentová frekvence	TF-IDF
Naivní Bayesův klasifikátor	17/99 17,2%	15/99 15,2%	32/99 32,3%
K – nejbližších sousedů (k = 3)	12/99 12,1%	12/99 12,1%	12/99 12,1%
K – nejbližších sousedů (k = 1)	13/99 13,1%	14/99 14,1%	11/99 11,1%

Obrázek 5: Tabulka správných klasifikací. Přesnost určujeme jako počet_správně_klasifikovaných_dokumentů / počet_klasifikovaných_dokumentů