DOI:
УДК 004.85
О.О. Шумейко, д.т.н., професор, *shumeiko_a@ukr.net*
В.С. Сотник, магістр, *vova.sotnyk@gmail.com*
І.І. Жульковська, к.т.н., доцент, *inivzh@gmail.com*
О.О. Жульковський, к.т.н., доцент, *olalzh@ukr.net*
Дніпровський державний технічний університет, м. Кам'янське

ВИКОРИСТАННЯ МЕТОДІВ DATA MINING ДЛЯ ОБРОБКИ МОВНОЇ ІНФОРМАЦІЇ

Зі збільшенням обсягів інформації, отриманої у результаті роботи інформаційних систем і процесів, у ході діяльності підприємств або іншої діяльності людства, обробка й аналіз даних стають значно складними. Для первинної обробки інформації з метою її структурування, виділення характерних ознак, узагальнення, сортування тощо застосовують Data Mining або інтелектуальний аналіз даних.

Важливим складником Data Mining є обробка текстової інформації. Такого роду задачі опираються на поняття класифікації й кластеризації.

Як показали отримані результати, наївний баєсівський класифікатор достатньо ефективно може використовуватися для розробки програмного забезпечення з обробки мовної інформації. Проте, у подальшому бажано як параметри розглядати також ланцюжки з декількох слів. У самому алгоритмі для запобігання втрат точності на довгих текстах потрібно використовувати замість перемножування ймовірностей (частот) додавання їх логарифмів.

***Ключові слова**: мовна інформація, класифікація, кластеризація, баєсівський класифікатор, алгоритм обробки текстів, C#*

With the increase in the amount of information obtained as a result of the work of information systems and processes, in the course of enterprises or other activities of mankind, data processing and analysis become much more complex. Data Mining or data mining is used for the primary processing of information for the purpose of its structuring, selection of characteristic features, generalization, sorting, etc.

An important component of Data Mining is the processing of textual information. Such problems are based on the concepts of classification and clustering.

As the results show, the naive Bayesian classifier can be used quite effectively to develop software for processing language information. Of course, this classifier did not show high accuracy in all cases. To do this, it is necessary to make certain changes. For example, words must be lemmatized or perform a family stemming operation before processing. In the algorithm itself, to prevent loss of accuracy in long texts, you need to use instead of multiplying the probabilities (frequencies) of adding their logarithms and so on.

***Keywords**: language information, classification, clustering, Bayesian classifier, word processing algorithm, C#*

Постановка проблеми

Зі збільшенням обсягів інформації, отриманої у результаті роботи інформаційних систем і процесів, у ході діяльності підприємств або науково-дослідної діяльності, її обробка й аналіз стають значно складними. Отже виникає необхідність первинної обробки інформації для її структурування, виділення характерних ознак, узагальнення, сортування.

З цією метою застосовують Data Mining або інтелектуальний аналіз даних, в основі якого лежать процеси класифікації й кластеризації, що дозволяють проводити первинну обробку інформації для її наступного аналізу [1—5].

Цей напрямок пов'язаний із широким спектром задач — від розпізнавання розмитих образів до створення пошукових машин [6]. Важливим складником Data Mining є обробка текстової інформації. Такого роду задачі опираються на поняття класифікації й кластеризації [3, 7, 8]. Класифікація полягає у визначенні приналежності деякого елемента одному із заздалегідь ство-

рених класів. Кластеризація має на меті розбивку множини елементів на кластери, кількість яких визначається локалізацією елементів заданої множини у межах деяких природних центрів цих кластерів. Реалізація задачі класифікації із самого початку повинна опиратися на задані постулати, основні з яких: апріорна інформація про первинну множину об'єктів і міра близькості елементів і класів.

Практично всі методи класифікації і кластеризації [9] засновані на гіпотезі компактності. Її можна сформулювати у такий спосіб — об'єкти, що відносяться до одного класу, повинні бути розташовані компактно хоча б в одному з можливих просторів опису об'єкта.

Перед роботою ставиться задача аналізу та застосування сучасних алгоритмів Data Mining для обробки мовної інформації.

Аналіз останніх досліджень та публікацій

Умовно методи кластеризації розбиваються на два класи — ієрархічні й неієрархічні. У неієрархічних алгоритмах присутня наявність умови зупинки й кількості кластерів. Основою цих алгоритмів є гіпотеза про порівняно невелике число схованих факторів, які визначають структуру зв'язку між ознаками. Ієрархічні алгоритми не зав'язані на кількості кластерів. Ця характеристика визначається динамікою злиття й поділу кластерів під час побудови дерева вкладених кластерів (дендрограми). У свою чергу, ієрархічні алгоритми діляться на агломеративні, які будуються шляхом об'єднання елементів, тобто зменшенням кількості кластерів, і дивизимні, засновані на поділі (розщепленні) існуючих груп (кластерів) [7, 10, 11].

Ієрархічні методи. Агломеративні алгоритми

На першому кроці з кожним елементом $\Im$ зв'язується свій кластер $c_i^0 = \{x_i\}$. На кожному наступному кроці два найбільш близькі кластери c_i^ν й c_i^μ поєднуються в один, результатом чого буде множина з *n-1* кластера $c_1^1 = \{c_\nu^0, c_\mu^0\}$, $c_i^1 = c_1^0 (i \neq \{\nu, \mu\})$.

Далі цей процес повторюється. У результаті одержимо один кластер, що збігається з $\Im$.

При об'єднанні ν і μ-го кластерів у *i*-й кластер потрібно обчислити відстань від нового кластера до *j*-го кластера. Традиційно для перерахування відстані під час злиття кластерів використовуються старі значення відстаней. Як правило, при цьому використовують наступні критерії:

– мінімальна відстань $d_{\min}(c_i, c_j) = \min\{\|x - y\| \mid x \in c_i, y \in c_j\}$.

Даний алгоритм сприяє росту витягнутих кластерів. До недоліків алгоритму, насамперед, варто віднести чутливість до шуму;

– максимальна відстань $d_{\max}(c_i, c_j) = \max\{\|x - y\| \mid x \in c_i, y \in c_j\}$.

Алгоритм на основі максимальної відстані сприяє формуванню компактних кластерів. До недоліків відноситься ефект руйнування витягнутих кластерів;

– середня відстань $\tilde{d}(c_i, c_j) = \frac{1}{n_i n_j} \sum_{x \in c_i} \sum_{x \in c_j} \|x - y\|$;

– відстань між центрами кластерів $d_\mu(c_i, c_j) = \|\mu_i - \mu_j\|$.

На практиці досить непогано працюють алгоритми, засновані на середній відстані, але з обчислювальної точки зору, більш ефективними є алгоритми, засновані на відстані між центрами кластерів.

Дивизимні алгоритми

Ідеологія дивизимних алгоритмів — двоїста агломеративним. На першому кроці вся множина $\Im$ представляється як один кластер, і на кожному кроці один з існуючих кластерів розбивається на два.

Найпростіший з такого роду алгоритмів описаний С. Макнаотоном у 1965 р. Його суть полягає у тому, що спочатку вибирається елемент кластера $c_0^1 = \Im$, що найбільш віддалений від

центра кластера, і цей елемент формує новий кластер c_2^1. Елементи, що залишилися, формують $c_1^1 = c_0^1 \setminus c_2^1$. На кожному наступному кроці елемент із c_1^1, для якого різниця між відстанню до центра кластера c_2^1 й відстанню до центра кластера c_1^1 — найбільша, переноситься у c_2^1 Цей процес триває доти, поки ця різниця не стане від'ємною, тобто поки не будуть обрані всі елементи з c_1^1, які ближче до центра кластера c_2^1. Результатом буде розщеплення одного кластера на два. Далі цей процес ітераційно триває. Вибір кластера, що розщеплюється, може проводитися, виходячи з різних міркувань, наприклад, вибирається кластер з найбільшим діаметром $\max\limits_{i,j}\left\|x_i - x_j\right\|$, де елементи x_i і x_j лежать в одному кластері. Як правило, дивизимні алгоритми використовуються у випадку малих вибірок.

Неієрархічні алгоритми

Неієрархічні алгоритми набули більшу популярність через те, що в їх основі лежить та або інша задача оптимізації, тобто групування вихідної множини об'єктів у кластери є розв'язком деякої екстремальної задачі.

Далі розглянуто найбільш популярні з методів.

Найбільш популярним із методів чисельного аналізу є метод найменших квадратів. Для задачі кластеризації він формулюється наступним чином:

$$\sum_{j=1}^{k}\sum_{i=1}^{n_j}\left|x_i - s_j\right|^2 \to \min$$

для усіх S_j і k.

Чисельна реалізація цієї задачі називається методом k-середніх.

Ідея методу k-середніх полягає у наступному. Спочатку вибирається k довільних вихідних центрів із множини $\Im$. Далі всі об'єкти розбиваються на k груп, найбільш близьких до відповідного центра. На наступному кроці обчислюються центри знайдених кластерів. Процедура повторюється доти, поки центри кластерів не стабілізуються.

Алгоритм розбивки об'єктів x_i *(i=0, 1, ..., n)* заснований на мінімізації міжкластерної відстані у випадку, якщо у якості відстані використовується середньоквадратична норма l_2, тобто цільовою функцією є:

$$S = \sum_{j=1}^{k}\sum\left\{\left|x_i - \mu_j\right|^2 \middle| x_i \in c_j\right\},$$

де x_i — *i-й* об'єкт, а c_j — *j-й* кластер із центром μ_j.

Структура алгоритму полягає у наступному:

1) для ініціалізації алгоритму вибираємо k центрів кластерів;

2) кожному з n об'єктів ставимо у відповідність кластер, виходячи з мінімізації l_2 норми між об'єктом і центром відповідного кластера;

3) перелічуємо центри знову отриманих кластерів;

4) для кожного *i,* такого, що $x_i \in c_j$ обчислюємо $h = \arg\min\left\{\dfrac{n_r\left\|x_i - \mu_r\right\|_2}{n_r - 1}\right\}$, де n_r — число об'єктів кластера C_r.

Для розв'язання цієї задачі серед усіх елементів кластера $x \in c_i$ знаходимо елемент z, який мінімізує відхилення $\sum\limits_{x\in c_i}\left\|x - z\right\|_2^2$, для чого знайдемо розв'язок:

$$\frac{\partial}{\partial z}\sum_{x\in c_i}\left\|x - z\right\|_2^2 = \frac{\partial}{\partial z}\sum_{x\in c_i}\left(\left\|x\right\|_2^2 - 2x^T z + \left\|z\right\|_2^2\right) = \sum_{x\in c_i}\left(-x + z\right) = 0, \text{тобто } z = \frac{1}{n_i}\sum_{x\in c_i} x;$$

5) якщо виконується умова $\frac{n_h \|x_i - \mu_h\|_2}{n_h - 1} < \frac{n_j \|x_i - \mu_j\|_2}{n_j - 1}$, то варто перемістити об'єкт x_i із кластера c_j в кластер c_h, після чого перелічити значення центрів кластерів;

6) якщо $i<n$, то переходимо до кроку 4, інакше — до кроку 3.

Критерієм зупинки алгоритму може служити або досягнення заданого числа ітерацій алгоритму, або досягнення функцією цілі заданого значення порога.

Метод є ефективним у випадку, якщо дані діляться на компактні групи, які можна описати сферою. Використання індикаторної функції дозволяє спростити запис базового алгоритму й записати його у наступному вигляді:

Нехай $C = \{c_i\}_{i=1}^k$ — множина кластерів із центрами $\mu_i = \frac{\sum\{x_j \mid x_j \in c_i\}}{\sum\{1 \mid x_j \in c_i\}} = \frac{\sum_{j=1}^n u_j^i x_j}{\sum_{j=1}^n u_j^i}$,

де індикаторна функція:

$$u_j^i = \begin{cases} 1, & \text{якщо } x_j \in c_i, \\ 0, & \text{інакше} \end{cases}. \quad (1)$$

Цільова функція: $S(C,\Im) = \sum_{i=1}^k \sum_{j=1}^n u_j^i d(x_j, \mu_i)$ і умови:

$$\sum_{i=1}^n u_j^i = 1, 0 < \sum_{j=1}^k u_j^i \le n, \quad (2)$$

тобто кожний елемент може бути тільки в одному кластері, і кластер не може бути порожнім або містити елементів більше, ніж їх вихідна кількість.

Умова зупинки виконання алгоритму після ν-го кроку буде мати вигляд:

$$\left| S^\nu(C,\Im) - S^{\nu-1}(C,\Im) \right| < \varepsilon, \quad (3)$$

де ε — обраний поріг.

Важливо, що при обчисленні критерію приналежності, можна враховувати розмір кластера, що дозволяє поліпшити ефективність алгоритму. Критерій того, що *j*-й елемент належить *i*-му, а не *k*-му кластеру буде мати вигляд:

$$\frac{n_i}{n_i - 1} d(x_j, \mu_i) < \frac{n_k}{n_k - 1} d(x_j, \mu_k),$$

де n_i — кількість елементів, співвіднесених кластеру c_i. Швидкість збіжності методу — *O(n)*.

До недоліків методу, насамперед, потрібно віднести: наявність апріорної інформації про кількість кластерів; чутливість до ізольованих вилучених елементів; істотну залежність швидкості збіжності методу від початкового вибору центрів кластерів.

Алгоритм Fuzzy *k*-means є узагальненням попереднього у випадку, коли кластери є нечіткими множинами, і елемент може належати різним кластерам з різним ступенем надійності.

Нехай $w \in (1,\infty)$ (зазвичай береться $w = 2$) — ваговий коефіцієнт нечіткості, а $C = \{c_i\}_{i=1}^k$ — множина кластерів із центрами $s_i = \sum_{j=1}^n \left(u_j^i\right)^w x_j \Big/ \sum_{j=1}^n \left(u_j^i\right)^w$, де індикаторна функція (1).

Цільова функція $S(C,\Im)=\sum_{i=1}^{k}\sum_{j=1}^{n}\left(u_j^i\right)^w d\left(x_j,\mu_i\right)$, і умови (2).

Умова зупинки виконання алгоритму після ν-го кроку буде мати вигляд (3).

Швидкість збіжності методу — *O(n)*.

Кластеризація Гюстафсона-Кесселя — по суті це той самий алгоритм, але використовує кореляційні залежності кластера, завдяки чому кластери замість сферичної форми стають еліпсоїдами. Це підвищує якість розбивки у випадку, коли елементи $\Im$ витягнуті уздовж яких-небудь напрямків. Іншими словами, якщо існують стійкі словосполучення, що визначають кластер, то краще розглядати елементи на приналежність до кластера уздовж цих сполучень.

Отже це той самий метод *k*-середніх, але використовується відстань Махаланобіса.

FOREL (формальний елемент) — одна з модифікацій *k*-середніх. Відмінність полягає у тому, що під близькістю елементів у кластері розуміється покриття їх сферою заданого радіуса.

Схема роботи алгоритму полягає у наступному. Береться центр кластера (на першому кроці це довільний елемент) і до кластера приписуються всі елементи, що віддалені від центра не більше, ніж на задану відстань R. Потім перелічується центр, у якості якого береться середня точка отриманого кластера, і заново перелічуються елементи кластера. Так триває доти, поки центр не стабілізується.

Метод *k*-середніх, як і його модифікації, опирається на апріорну інформацію про кількість класів. Це можна обійти. Для цього застосовується метод *k*-середніх або його модифікація послідовно для кожного *k* і обчислюється максимальна похибка. Як тільки значення цієї похибки стабілізується, тобто кількість кластерів устоялося, так відразу й зупиняємося. Такий підхід є довгим.

В основі цього методу кореляційних плеяд лежить ідея побудови кластерів, виходячи з максимізації кореляційного зв'язку, тобто сума модулів коефіцієнтів кореляції між параметрами однієї групи повинна бути досить велика, а зв'язок між параметрами з різних груп — малий. За кореляційною матрицею об'єктів будується граф, який потім розбивається на підграфи. Елементи, що відповідають кожному з підграфів, і утворять кластер.

Розглядається кореляційна матриця $C=\{c_{i,j}\}$, *i*, *j*=1, 2, …, *n* вихідних об'єктів і проводиться упорядкування, що створюється на підставі принципу максимального кореляційного шляху — всі *n* об'єктів зв'язуються за допомогою *n*-1 ліній (ребер) так, аби сума модулів коефіцієнтів кореляції була максимальною.

Для побудови графа спочатку розглядаються вершини графа, що відповідають об'єктам x_l і x_m. Ребру, що з'єднує ці вершини ставиться у відповідність вага $c^{(1)}=\left|c_{l,m}\right|$. Потім, виключивши $x_{l,m}$, знаходиться найбільший коефіцієнт в *m*-ом стовпці матриці (це відповідає знаходженню елемента, що найбільше сильно після x_l «зв'язаний» з x_m) і найбільший коефіцієнт в *l*-му рядку матриці (це відповідає знаходженню елемента, найбільш міцно після x_m «зв'язаного» з x_l). Зі знайдених у такий спосіб двох коефіцієнтів кореляції вибирається найбільший, наприклад $c^{(2)}=\left|c_{l,j}\right|$. Вершина x_j з'єднується з x_l, і відповідному ребру проставляється значення $c^{(2)}$. Потім знаходяться об'єкти, найбільш зв'язані з x_l, x_m і x_j і вибирається зі знайдених коефіцієнтів кореляції найбільший, наприклад $c^{(3)}=\left|c_{j,q}\right|$. Встановлюється умова, аби на кожному кроці з'являвся новий елемент, тому вже використовувані елементи, виключаються. Отже $q\neq l, q\neq m, q\neq j$. Далі вершина графа, що відповідає x_q, з'єднується з x_j і т.д. На кожному кроці визначаються елементи, найбільш сильно пов'язані із двома останніми розглянутими елементами, а потім вибирається один з них, що відповідає більшому коефіцієнту кореляції. Процедура закінчується після *n*-1-го кроку. Граф складається з *n* вершин, з'єднаних *n*-1-м ребром. Після цього задається граничне значення ε, і всі ребра, що відповідають меншим за ε коефіцієнтам кореляції, виключаються із графа. Отримані підграфи формують кластери.

Формулювання мети дослідження

На основі проведеного аналізу встановлено, що для створення ефективної системи обробки мовної інформації існує кілька методів і технологій Data Mining, серед яких — ієрархічні й неієрархічні. У свою чергу, ієрархічні алгоритми діляться на агломеративні, які будуються шляхом об'єднання елементів, тобто зменшенням кількості кластерів, і дивизимні, засновані на поділі (розщепленні) існуючих груп (кластерів). Тому актуальним є дослідження та застосування існуючих методів та класифікаторів для обробки мовної інформації.

Метою даної роботи є застосування наївного баєсівського класифікатора для розробки програмного забезпечення обробки текстів.

Виклад основного матеріалу

На багатьох сайтах, присвячених фільмам, книгам, товарам, є можливість залишати відгуки-рецензії. У роботі ставиться модельна задача «навчити» класифікатор відрізняти негативну рецензію від позитивної.

Для цього підготуємо кілька прикладів позитивних відгуків і негативних, тобто будемо імітувати так зване навчання із учителем. Параметрами, за якими класифікатор буде намагатися визначати емоційне фарбування тексту, будуть окремі слова, що є присутніми у текстах, а точніше — частоти їхнього входження до текстів.

Будемо використовувати наступну модель задачі класифікації.

Нехай: Ω — множина об'єктів розпізнавання (простір образів); $\omega\in\Omega$ — об'єкт розпізнавання (образ); $g(\omega)$*:* $\Omega\to\Re$*,* $\Re=\{1, 2,\ldots, n\}$ — індикаторна функція, що розбиває простір образів Ω на n класів Ω^1, Ω^2, …, Ω^n, що не перетинаються. Індикаторна функція невідома спостерігачеві; X — простір спостережень, що сприймається спостерігачем (простір ознак); $x(\omega)$: $\Omega\to X$ — функція, що ставить у відповідність кожному об'єкту ω точку $x(\omega)$ у просторі ознак. Вектор $x(\omega)$ — це образ об'єкта, що сприймається спостерігачем.

У просторі ознак визначені непересічні множини точок $\Xi[i]\subset X$, $i=1, 2..., n$, що відповідають образам одного класу. $\varphi(x)$*:* $X\to\Re$ вирішальне правило — оцінка для $g(\omega)$ на підставі $x(\omega)$, тобто $\varphi(x)=\varphi(x(\omega))$.

Нехай $x_\nu = x(\omega_\nu)$, $\nu=1, 2, \ldots N$ — доступна спостерігачеві інформація про функції $g(\omega)$ і $x(\omega)$, але самі ці функції спостерігачеві невідомі. Тоді (g_ν, x_ν), $\nu=1, 2, \ldots N$ — є множина прецедентів.

Задача полягає у побудові такого вирішального правила $\varphi(x)$, щоб розпізнавання проводилося з мінімальним числом помилок.

Баєсівський класифікатор ґрунтується на тому, що відомі апріорні ймовірності гіпотез $P(c_i)$, тобто імовірності випадання класів c_i $(i=1,2,...,k)$. Відповіді на питання: «Як їх знайти?», баєсівський класифікатор не дає [12]. Наївний баєсівський класифікатор дозволяє вирішити цю проблему, щоправда, не дуже якісно, проте це краще, ніж нічого. Найбільше часто використовувана область наївного баєсівського класифікатора ставиться до задачі класифікації текстів, де цей метод дозволяє одержати досить непогані результати [12]. Наївний баєсівський класифікатор припускає умовну незалежність атрибутів, зокрема при обробці текстів. Припущення наївного класифікатора зовсім бентежне — ймовірність появи слова не залежить від інших слів у документі і, більш того, не залежить від розміру документа. Як не дивно, наївний баєсівський класифікатор для обробки текстів виявився досить ефективним і одержав широке поширення, зокрема для фільтрації спама [13].

Передбачається, що алгоритм наївної баєсівської класифікації працює на деякій множині документів $D=\{b_i\}$. Уся множина документів розбивається на підмножини класів, що не перетинаються: $C = \{c_i\}, \bigcup_i b_i = D, c_i \bigcap_i c_j = \varnothing (i \neq j)$.

Задачею класифікації є визначення класу, до якого відноситься даний документ. Кожному елементу b ставиться у відповідність набір ознак $b=\{w_i\}$. Набір документів, що визначає клас, надалі будемо називати навчальною вибіркою.

Далі застосовується алгоритм класифікації для виділення документів, які найбільше відповідають заданому класу.

Для того, аби застосувати теорему Баєса для класифікації документів, робляться наступні припущення: $P(c_j) = n(c_j) \Big/ \sum_j n(c_j)$, де $n(c_j)$ — кількість термів у класі c_j.

Передбачається, що всі терми (слова, словосполучення) — незалежні, тобто $P(w_i|c_j) = n(w_i, c_j) \Big/ n(c_j)$, де $\{w_j\}$ — набір термів у документі b; $n(w_i, c_j)$ — кількість термів w_i у класі c_j.

Для визначення підходящої категорії документів для заданого документа потрібно одержати відповідну множину словоформ. За множиною словоформ будується структура з неповторюваних слів і їх лічильників (w_i, n_i)

Визначення підходящої категорії починається з кореня дерева множин статистики. Через M позначимо кількість множин статистики у даному вузлі дерева. Категорії, на приналежність до яких перевіряється документ, позначимо через c_j (j=0, 1, …, M-1). Для кожного слова w_i у кожній множині статистики знаходимо це слово й відповідний лічильник $n(w_i, c_j)$ (тут j=0, 1, .., M-1 — номер категорії (множини статистики)). Через $n(c_j)$ позначимо число документів в j-й категорії. Крім того, нехай $N_j(w_i) = n(w_i, c_j) \Big/ n(c_j)$ — нормований лічильник слова w_i в *j-й* категорії. Тоді ймовірність відповідності унікального (тобто кожне слово зустрічається тільки один раз) слова w_i *j-ї* категорії буде дорівнювати:

$$P(c_j|w_i) = \frac{N_j(w_i)}{S(w_i)} n(w_i, c_j), \tag{4}$$

де $S(w_i) = \sum_{j=0}^{M-1} N_j(w_i)$.

Якщо $P(c_j|w_i) = 0$, то потрібно взяти це число рівним малому значенню, наприклад, 0,0001. Тоді ймовірність того, що документ відповідає категорії c_j (j=0, 1, …, M-1) буде дорівнювати $P(c_j|\{w_i\}) = P(c_j) \prod_i P(c_j|w_i)$, де добуток береться по всіх словах досліджуваної множини словоформ та $P(c_j) = n(c_j) \Big/ \sum_{j=0}^{M-1} n(c_j)$ — апріорна ймовірність зустрічі категорії c_j.

Зауважимо, якщо документ містить велику кількість слів, які не зустрічаються в категорії c_j (j=0, 1, …, M-1), то значення $P(c_j|\{w_i\})$ може вийти за межі визначення змінної. Тому значення $P(c_j|\{w_i\})$ потрібно контролювати, і, якщо воно виходить за межі значення змінної, то обмежувати заданим малим числом, наприклад, нулем.

У випадку, якщо не відкидаються стоп-слова (тобто такі, що не несуть інформацію про тематику тексту, наприклад, «тоді», «якщо», «але» тощо), то має сенс зменшити вплив слів, які зустрічаються у великій кількості категорій. Для цього має сенс використовувати конструкцію:

$$P(c_j|w_i) = \frac{N_j(w_i)}{S(w_i)} n(w_i, c_j) \log \frac{M + 1/2}{M(w_i) + 1/2},$$

де $M(w_i)$ — кількість категорій, у яких зустрічається слово w_i

Зауважимо, що у цьому випадку, якщо всі категорії містять всі слова документа, то алгоритм покаже невідповідність. Таким чином, цей випадок (коли всі категорії містять усі слова) повинен оброблятися з використанням формули (4).

Після першого кроку визначаємо k категорій з найбільшим значенням $P(c_j|\{w_i\})$. Зберігаємо їх назву й значення $P(c_j|\{w_i\})$. Відповідно до назви кожного із цих множин заходимо у

відповідний пункт і проводимо обробку. Якщо інформація відсутня, то цей пункт пропускається. Після того, як будуть оброблені всі дочірні (по визначеним на першому кроці) категорії, з них і збережених на попередньому кроці вибирається k категорій з найбільшим значенням $P\left(c_j \middle| \{w_i\}\right)$. Зберігаємо їх назву й значення $P\left(c_j \middle| \{w_i\}\right)$, після чого переходимо до наступного кроку, але тільки за тими категоріям, які не були збережені на попередньому кроці. Процес продовжуємо доти, поки не буде категорій, за якими можна здійснювати перевірку.

Результатом виконання програми буде k категорій, найбільш імовірних для досліджуваного документа з погляду критерію Баєса.

Для реалізації задачі було використано мову C#, яка претендує на справжню об'єктно-орієнтованість, покликана реалізувати компонентно-орієнтований підхід до програмування, зменшуючи машинно-архітектурні залежності результуючого програмного коду, приводить до більшої гнучкості, переносимості і легкості повторного використання програм, є найпридатнішою для створення додатків у середовищі Microsoft .NET, оскільки найтісніше та ефективно інтегрована з нею.

Обчислювальні експерименти проводилися з використанням інфраструктури, представленої у табл. 1.

Таблиця 1. Інфраструктура обчислювального експерименту

Процесор (CPU)	Intel Core i5-7400 (4 cores, 3-3.5 GHz),cache 6 MB
Пам’ять (RAM)	Kingston HyperX DDR4 (8GB, 2133 MHz, 17000 MB/s) + Kingston HyperX DDR4 (8GB, 3200 MHz, 23500 MB/s)
Операційна система (OS)	Microsoft Windows 10
Середовище розробки (IDE)	Microsoft Visual Studio, 2017

Результати роботи додатку представлені на рис. 1.

```
Відгук:

            У цілому фільм варто сприймати як приємну
            милу картину, що змушує зворушуватися, посміхатися,
            співчувати героям. Раджу до перегляду дівчатам,
            які мріють про принца на білому коні й вірять у любов

Гіпотеза: +, імовірність 4,84045320064746E-69
Гіпотеза: -, імовірність 7,04909421309603E-70

Відгук:

            Поганий, нудний фільм. Поки я його догледів, поспав
            три рази. Не варто гаяти час на це непорозуміння

Гіпотеза: -, імовірність 3,11910811752617E-41
Гіпотеза: +, імовірність 8,62700828239524E-42
```

Рис. 1. Приклад роботи консольного додатку

Як видно, результат цілком правдоподібний. Звичайно, не варто розраховувати, що даний класифікатор покаже більшу точність в усіх випадках. Для цього до нього необхідно внести деякі зміни. Наприклад, слова перед їх обробкою необхідно лематизувати — приводити до основної форми або хоча б виконувати родинну операцію стемінгу. Це особливо актуально для багатих у плані морфології російської й української мов.

Висновки

Як показали результати роботи, наївний баєсівський класифікатор достатньо ефективно може використовуватися для розробки програмного забезпечення з обробки мовної інформації. Проте, у подальшому бажано як параметри розглядати також ланцюжки з декількох слів. У самому алгоритмі для запобігання втрат точності на довгих текстах потрібно використовувати замість перемножування ймовірностей (частот) додавання їх логарифмів.

Список використаної літератури

1. Айвазян С.А., Бежаева Э.Н., Староверов О.В. Классификация многомерных наблюдений. М., 1974. 238 с.
2. Дюран М.Б. Кластерный анализ. М.: Финансы и статистика, 1977. 128 с.
3. Классификация и кластер / под ред. Дж. Вэн Райвин; пер. с англ. под ред. Ю.И. Журавлева. М.: Мир, 1978.
4. Мандель И.Д. Кластерный анализ. М.: Финансы и статистика, 1988. 176 с.
5. Fraley C., Raftery A. How many clusters? Which clustering method? Answers via model-based cluster analysis. *The Computer Journal*. 1998. 41. P. 578–588.
6. Mercer D.P. Clustering large datasets. URL: http://ldc.usb.ve/~mcuriel/Cursos/WC/Transfer.pdf
7. Jain A.K., Murty M.N., Flynn P.J. Data clustering: a review. *ACM Computing Surveys*. 1999. V.31, №3. P. 264-323.
8. Gordon A. Classification. Chapman and Hall, London, 1999.
9. Райзен Дж. В. Классификация и кластер. *Трудыы науч.семинара*. М.: Мир, 1980.
10. Jain A.K. Data Clustering. URL: http://www.csee.umbc.edu/ nicholas/clustering/p264-jain.pdf
11. Mercer D.P. Clustering large datasets. URL: http://ldc.usb.ve/~mcuriel/Cursos/WC/Transfer.pdf
12. Айвазян С.А., Бухштабер В.М., Енюков И.С., Мешалкин Л.Д. Прикладная статистика: Классификация и снижение размерности. М.: Финансы и статистика, 1989. 450 с.
13. Ферстер Э., Ренц Б. Методы корреляционного и регрессионного анализа. М.: Финансы и статистика, 1983. 299 с.

USING DATA MINING METHODS FOR PROCESSING LANGUAGE INFORMATION

Shumeyko O., Sotnyk V., Zhulkovska I., Zhulkovskyi O.

With the increase in the amount of information obtained as a result of the work of information systems and processes, in the course of enterprises or other activities of mankind, data processing and analysis become much more complex. Data Mining or data mining is used for the primary processing of information for the purpose of its structuring, selection of characteristic features, generalization, sorting, etc.

An important component of Data Mining is the processing of textual information. Such problems are based on the concepts of classification and clustering.

Based on the analysis, it is established that to create an effective system of language information processing there are several methods and technologies of Data Mining, among which – hierarchical and non-hierarchical. In turn, hierarchical algorithms are divided into agglomerative, which are built by combining elements, that is reducing the number of clusters, and divisional, based on the division (splitting) of existing groups (clusters). Therefore, it is important to study and apply existing methods and classifiers for processing language information.

The purpose of this work is to use a naive Bayesian classifier for software development for word processing.

As the results show, the naive Bayesian classifier can be used quite effectively to develop software for processing language information. Of course, this classifier did not show high accuracy in

all cases. To do this, it is necessary to make certain changes. For example, words must be lemmatized or perform a family stemming operation before processing. This is especially actual for the rich in terms of morphology of Russian and Ukrainian languages.

Also in the future it is desirable to consider chains of several words as parameters. In the algorithm itself, to prevent loss of accuracy in long texts, you need to use instead of multiplying the probabilities (frequencies) of adding their logarithms and so on.

References

[1] Ayvazyan, S.A., Bezhaeva, E.N., Staroverov, O.V. (1974). *Klassifikatsiya mnogomernyih nablyudeniy [Classification of multivariate observations]*. Moscow [in Russia].
[2] Dyuran, M.B. (1977). *Klasternyiy analiz [Cluster Analysis]*. Moscow: Finansyi i statistika [in Russia].
[3] *Klassifikatsiya i klaster [Classification and cluster]* (1978). Dzh. Ven Rayvin (Ed.). (Yu.I. Zhuravleva, Trans). Moscow: Mir [in Russia].
[4] Mandel, I.D. (1988). *Klasternyiy analiz [Cluster Analysis]*. Moscow: Finansyi i statistika [in Russia].
[5] Fraley, C. (1998). How many clusters? Which clustering method? Answers via model-based cluster analysis. *The Computer Journal, 41*, 578–588 [in English].
[6] Mercer, D.P. *Clustering large datasets.* Retrieved from: http://ldc.usb.ve/~mcuriel/Cursos/WC/Transfer.pdf [in English].
[7] Jain, A.K. (1999). Data clustering: a review. *ACM Computing Surveys*, *31 (3),* 264-323 [in English].
[8] Gordon, A. (1999). *Classification*. London: Chapman and Hall [in English].
[9] Rayzen, Dzh.V. (1980). *Klassifikatsiya i klaster [Classification and cluster]*. Moscow: Mir [in Russia].
[10] Jain A.K. *Data Clustering*. Retrieved from: http://www.csee.umbc.edu/ nicholas/clustering/p264-jain.pdf [in English].
[11] Mercer, D.P. *Clustering large datasets*. Retrieved from: (http://ldc.usb.ve/~mcuriel/Cursos/WC/Transfer.pdf) [in English].
[12] Ayvazyan, S.A., Buhshtaber, V.M., Enyukov, I.S., Meshalkin, L.D. (1989). *Prikladnaya statistika: Klassifikatsiya i snizhenie razmernosti [Applied Statistics: Classification and Dimension Reduction]*. Moscow: Finansyi i statistika [in Russia].
[13] Ferster, E., Rents, B. (1983). *Metodyi korrelyatsionnogo i regressionnogo analiza [Correlation and regression analysis methods]*. Moscow: Finansyi i statistika [in Russia].