Лабораторна робота № 2

Тема: "Наївний баєсів класифікатор для аналізу тональності текстів"

```
import nltk
import numpy as np
import re
import string
import json
from collections import defaultdict
from nltk.corpus import stopwords, twitter_samples
from nltk.stem import PorterStemmer
from nltk.tokenize import TweetTokenizer
```

ЗАВДАННЯ 1: Завантаження та попередня обробка

```
def process_tweet(tweet):
    """
    Обробляє твіт: токенізація, видалення стоп-слів, стемінг.
    (Функція надана в описі лабораторної)
    """
    stemmer = PorterStemmer()
    stopwords_english = stopwords.words('english')

    # Видалення $GE, RT, посилань
    tweet = re.sub(r'\$\w*', '', tweet)
    tweet = re.sub(r'^RT[\s]+', '', tweet)
    tweet = re.sub(r'https?://[^\s\n\r]+', '', tweet)
    # Видалення хештегів (#)
    tweet = re.sub(r'#', '', tweet)

    # Токенізація
    tokenizer = TweetTokenizer(preserve_case=False, strip_handles=True, reduce_len=True)
    tweet_tokens = tokenizer.tokenize(tweet)

    tweets_clean = []
    for word in tweet_tokens:
        if (word not in stopwords_english and  # видалення стоп-слів
            word not in string.punctuation):   # видалення пунктуації
            stem_word = stemmer.stem(word)  # стемінг
            tweets_clean.append(stem_word)

    return tweets_clean
```

ЗАВДАННЯ 2: Побудова словника частотності

```
def count_tweets(result, tweets, ys):
    """
    Будує словник частот freqs[(word, label)] = count.
    (Функція надана в описі лабораторної)
    """
    # Переконуємось, що ys - це звичайний список
    if isinstance(ys, np.ndarray):
        yslist = np.squeeze(ys).tolist()
    else:
        yslist = ys

    for y, tweet in zip(yslist, tweets):
        for word in process_tweet(tweet):
            pair = (word, y)
            result[pair] = result.get(pair, 0) + 1

    return result
```

ЗАВДАННЯ 3 & 4: Обчислення Log Prior та Log Likelihood

```
def train_naive_bayes(freqs, train_y):
    """
    Навчає модель: обчислює logprior та loglikelihood.
    (Функція надана в описі лабораторної, з виправленням міток)
    """
    loglikelihood = {}
```

```
    logprior = 0

    # Отримуємо унікальні слова (словник V)
    vocab = set([pair[0] for pair in freqs.keys()])
    V = len(vocab)

    # N_pos, N_neg - загальна кількість слів у кожному класі
    N_pos = N_neg = 0
    for pair, count in freqs.items():
        # pair[1] - це мітка (1.0 або 0.0)
        if pair[1] > 0: # Позитивний (1.0)
            N_pos += count
        else: # Негативний (0.0)
            N_neg += count

    # D - загальна к-сть документів (твітів)
    D = len(train_y)
    # D_pos, D_neg - к-сть позитивних/негативних документів
    D_pos = np.sum(train_y) # Сума всіх '1'
    D_neg = D - D_pos

    # ЗАВДАННЯ 3: Обчислення Log Prior
    logprior = np.log(D_pos) - np.log(D_neg)

    # ЗАВДАННЯ 4: Обчислення Log Likelihood
    for word in vocab:
        # Отримуємо частоти слів (з уникненням KeyError)
        freq_pos = freqs.get((word, 1.0), 0)
        freq_neg = freqs.get((word, 0.0), 0)

        # Обчислюємо ймовірності P(W|Pos) та P(W|Neg) зі згладжуванням Лапласа (+1)
        p_w_pos = (freq_pos + 1) / (N_pos + V)
        p_w_neg = (freq_neg + 1) / (N_neg + V)

        loglikelihood[word] = np.log(p_w_pos / p_w_neg)

    return logprior, loglikelihood
```

ЗАВДАННЯ 5: Реалізація функції класифікатора

```
def naive_bayes_predict(tweet, logprior, loglikelihood):
    """
    Прогнозує тональність твіту.
    (Функція надана в описі лабораторної)
    """
    word_l = process_tweet(tweet)

    # Починаємо з апріорної ймовірності
    p = logprior

    for word in word_l:
        if word in loglikelihood:
            # Додаємо логарифм правдоподібності для цього слова
            p += loglikelihood[word]

    # Якщо p > 0, твіт позитивний, інакше - негативний
    return p
```

ЗАВДАННЯ 6: Оцінка точності

```
def test_naive_bayes(test_x, test_y, logprior, loglikelihood):
    """
    Оцінює точність класифікатора на тестовій вибірці.
    (Функція надана в описі лабораторної, з виправленням типів)
    """
    y_hats = [] # Список наших прогнозів

    for tweet in test_x:
        # Отримуємо прогноз (score)
        p = naive_bayes_predict(tweet, logprior, loglikelihood)

        # Якщо score > 0, мітка 1.0 (Pos), інакше 0.0 (Neg)
        if p > 0:
            y_hat_i = 1.0
        else:
            y_hat_i = 0.0
        y_hats.append(y_hat_i)

    # Порівнюємо прогнози (y_hats) з реальними мітками (test_y)
```

```
    # np.mean(y_hats == test_y) - це елегантний спосіб порахувати точність
    error = np.mean(np.absolute(np.array(y_hats) - test_y))
    accuracy = 1 - error

    return accuracy
```

ЗАВДАННЯ 7: Аналіз слів

```
def lookup(freqs, word, label):
    """(Функція надана в описі лабораторної)"""
    n = 0
    pair = (word, label)
    if (pair in freqs):
        n = freqs[pair]
    return n

def get_ratio(freqs, word):
    """(Функція надана в описі лабораторної, з виправленням міток)"""
    pos_neg_ratio = {'positive': 0, 'negative': 0, 'ratio': 0.0}
    pos_neg_ratio['positive'] = lookup(freqs, word, 1.0) # Використовуємо 1.0
    pos_neg_ratio['negative'] = lookup(freqs, word, 0.0) # Використовуємо 0.0

    pos_neg_ratio['ratio'] = (pos_neg_ratio['positive'] + 1) / (pos_neg_ratio['negative'] + 1)
    return pos_neg_ratio

def get_words_by_threshold(freqs, label, threshold):
    """(Функція надана в описі лабораторної)"""
    word_list = {}
    for key in freqs.keys():
        word, _ = key
        pos_neg_ratio = get_ratio(freqs, word)

        if label == 1 and pos_neg_ratio['ratio'] >= threshold:
            word_list[word] = pos_neg_ratio
        elif label == 0 and pos_neg_ratio['ratio'] <= threshold:
            word_list[word] = pos_neg_ratio

    return word_list
```

ГОЛОВНИЙ БЛОК ВИКОНАННЯ

```
if __name__ == "__main__":

    # 1. Завантаження NLTK даних
    nltk.download('twitter_samples', quiet=True)
    nltk.download('stopwords', quiet=True)
    nltk.download('punkt', quiet=True) # Потрібен для TweetTokenizer

    # 2. (Завдання 1) Завантаження та розділення корпусу
    print("Завантаження та розділення даних twitter_samples...")
    all_positive_tweets = twitter_samples.strings('positive_tweets.json')
    all_negative_tweets = twitter_samples.strings('negative_tweets.json')

    train_pos = all_positive_tweets[:4000]
    test_pos = all_positive_tweets[4000:]
    train_neg = all_negative_tweets[:4000]
    test_neg = all_negative_tweets[4000:]

    train_x = train_pos + train_neg
    test_x = test_pos + test_neg

    # Створюємо мітки: 1.0 для позитивних, 0.0 для негативних (як у build_freqs)
    train_y = np.append(np.ones(len(train_pos)), np.zeros(len(train_neg)))
    test_y = np.append(np.ones(len(test_pos)), np.zeros(len(test_neg)))
    print(f"Тренувальна вибірка: {len(train_x)} твітів")
    print(f"Тестова вибірка: {len(test_x)} твітів")

    # 3. (Завдання 2) Побудова словника частотності
    print("\nПобудова словника частот...")
    # Передаємо мітки train_y (1.0/0.0)
    freqs = count_tweets({}, train_x, train_y)
    print(f"Словник частот побудовано. {len(freqs)} унікальних пар.")

    # 4. (Завдання 3 & 4) Навчання моделі
    print("\nНавчання моделі (обчислення logprior та loglikelihood)...")
    logprior, loglikelihood = train_naive_bayes(freqs, train_y)
    print(f"Навчання завершено. LogPrior = {logprior:.4f}")

    # 5. (Завдання 6) Оцінка точності
```

```
print("\n--- Завдання 6: Оцінка точності ---")
accuracy = test_naive_bayes(test_x, test_y, logprior, loglikelihood)
print(f"Точність наївного баєсового класифікатора: {accuracy * 100:.2f}%")

# 6. (Завдання 7) Аналіз слів
print("\n--- Завдання 7: Аналіз слів ---")
print("Найбільш позитивні слова (ratio >= 10):")
pos_words = get_words_by_threshold(freqs, label=1, threshold=10)
for word, ratio in sorted(pos_words.items(), key=lambda x: x[1]['ratio'], reverse=True)[:10]:
    print(f"  {word}: {ratio['ratio']:.2f} (Pos: {ratio['positive']}, Neg: {ratio['negative']})")

print("\nНайбільш негативні слова (ratio <= 0.1):")
neg_words = get_words_by_threshold(freqs, label=0, threshold=0.1)
for word, ratio in sorted(neg_words.items(), key=lambda x: x[1]['ratio'])[:10]:
    print(f"  {word}: {ratio['ratio']:.2f} (Pos: {ratio['positive']}, Neg: {ratio['negative']})")

# 7. (Завдання 8) Аналіз помилок
print("\n--- Завдання 8: Аналіз помилок ---")
print('True\tPred\tTweet (перші 20 помилок)')
print('-------------------------------------')
errors_found = 0
for x, y in zip(test_x, test_y):
    if errors_found >= 20: # Обмежимо вивід
        break

    y_hat_score = naive_bayes_predict(x, logprior, loglikelihood)
    y_hat = 1.0 if y_hat_score > 0 else 0.0

    if y != y_hat:
        errors_found += 1
        # .encode('ascii', 'ignore') - щоб уникнути помилок виводу
        print(f'{y}\t{y_hat}\t{x.encode("ascii", "ignore")}')

# 8. (Завдання 9) Тестування на власних твітах
print("\n--- Завдання 9: Тест на власному твіті ---")

my_tweet_1 = 'I am happy because I am learning :)'
p1 = naive_bayes_predict(my_tweet_1, logprior, loglikelihood)
print(f"Твіт: '{my_tweet_1}'")
print(f"  Score: {p1:.4f} -> Прогноз: {'Позитивний' if p1 > 0 else 'Негативний'}")

my_tweet_2 = 'This is a terrible and awful experience. I hate it.'
p2 = naive_bayes_predict(my_tweet_2, logprior, loglikelihood)
print(f"\nТвіт: '{my_tweet_2}'")
print(f"  Score: {p2:.4f} -> Прогноз: {'Позитивний' if p2 > 0 else 'Негативний'}")

my_tweet_3 = "This movie was not bad, actually. I kind of liked it."
p3 = naive_bayes_predict(my_tweet_3, logprior, loglikelihood)
print(f"\nТвіт (складний): '{my_tweet_3}'")
print(f"  Score: {p3:.4f} -> Прогноз: {'Позитивний' if p3 > 0 else 'Негативний'}")

# 9. (Завдання 10) Збереження моделі для GitHub
print("\n--- Завдання 10: Збереження моделі ---")
model_data = {
    'logprior': logprior,
    'loglikelihood': loglikelihood
}

try:
    with open('naive_bayes_model.json', 'w', encoding='utf-8') as f:
        json.dump(model_data, f, indent=4)
    print("Модель (logprior та loglikelihood) успішно збережено у 'naive_bayes_model.json'")
except Exception as e:
    print(f"Помилка збереження моделі: {e}")
```

```
Завантаження та розділення даних twitter_samples...
Тренувальна вибірка: 8000 твітів
Тестова вибірка: 2000 твітів

Побудова словника частот...
Словник частот побудовано. 11397 унікальних пар.

Навчання моделі (обчислення logprior та loglikelihood)...
Навчання завершено. LogPrior = 0.0000

--- Завдання 6: Оцінка точності ---
Точність наївного баєсового класифікатора: 99.55%

--- Завдання 7: Аналіз слів ---
Найбільш позитивні слова (ratio >= 10):
  :): 987.00 (Pos: 2960, Neg: 2)
  :-): 553.00 (Pos: 552, Neg: 0)
  :d: 524.00 (Pos: 523, Neg: 0)
  :p: 106.00 (Pos: 105, Neg: 0)
```

```
  stat: 52.00 (Pos: 51, Neg: 0)
  bam: 45.00 (Pos: 44, Neg: 0)
  warsaw: 45.00 (Pos: 44, Neg: 0)
  blog: 28.00 (Pos: 27, Neg: 0)
  fback: 27.00 (Pos: 26, Neg: 0)
  followfriday: 24.00 (Pos: 23, Neg: 0)

Найбільш негативні слова (ratio <= 0.1):
  :(: 0.00 (Pos: 1, Neg: 3675)
  :-(: 0.00 (Pos: 0, Neg: 386)
  ♛: 0.00 (Pos: 0, Neg: 210)
  》: 0.00 (Pos: 0, Neg: 210)
  >:(: 0.02 (Pos: 0, Neg: 43)
  believ: 0.03 (Pos: 0, Neg: 35)
  will: 0.03 (Pos: 0, Neg: 35)
  justin: 0.03 (Pos: 0, Neg: 35)
  ｓｅｅ: 0.03 (Pos: 0, Neg: 35)
  ｍｅ: 0.03 (Pos: 0, Neg: 35)

--- Завдання 8: Аналіз помилок ---
True    Pred    Tweet (перші 20 помилок)
-------------------------------------
1.0     0.0     b'@jaredNOTsubway @iluvmariah @Bravotv Then that truly is a LATERAL move! Now, we all know the Queen Bee i
1.0     0.0     b'A new report talks about how we burn more calories in the cold, because we work harder to warm up. Feel
1.0     0.0     b"Harry and niall and -94 (when harry was born) ik it's stupid and i wanna change it :D https://t.co/gHAt8
1.0     0.0     b'off to the park to get some sunlight : )'
1.0     0.0     b'@msarosh Uff Itna Miss karhy thy ap :p'
0.0     1.0     b'@rcdlccom hello, any info about possible interest in Jonathas ?? He is close to join Betis :( greatings'
0.0     1.0     b'@phenomyoutube u probs had more fun with david than me : ('
0.0     1.0     b'pats jay : ('
0.0     1.0     b'Sr. Financial Analyst - Expedia, Inc.: (#Bellevue, WA) http://t.co/ktknMhvwCI #Finance #ExpediaJobs #Job

--- Завдання 9: Тест на власному твіті ---
Твіт: 'I am happy because I am learning :)'
  Score: 9.5605 -> Прогноз: Позитивний

Твіт: 'This is a terrible and awful experience. I hate it.'
  Score: -4.4578 -> Прогноз: Негативний
```