שאילתות באינטרנט

תרגיל 1 – מבנה האינדקס

1. ערכת הנתונים

נתון קובץ המכיל אוסף של ביקורות על מוצרים. כל אחת מהביקורות היא מהמבנה הבא:

product/productId: B001E4KFG0

review/userId: A3SGXH7AUHU8GW

review/profileName: delmartian

review/helpfulness: 1/1

review/score: 5.0

review/time: 1303862400

review/summary: Good Quality Dog Food

review/text: I have bought several of the Vitality canned dog food products and have found them all to be of good quality. The product looks more like a stew than a processed meat and it smells better. My Labrador is finicky and she appreciates this product better than most.

מכיוון שלביקורות אין מזהה (ID), נמספר אותן בסדר עולה. כלומר הביקורת הראשונה תקבל את המזהה 1, הביקורת השנייה תקבל את המזהה 2 וכן הלאה.

מתוך הנתונים שיש לכל אחת מהביקורות, נתעניין בשדות הבאים:

- product/productId (string containing 10 characters)
- review/helpfulness (two integers)
- review/score (integer between 1 and 5)
- review/text

עיבוד הטקסט של הביקורות למילים יתבצע באופן הבא:

- חלוקת הטקסט למילים נפרדות בכל מקום שבו יש תו שאינו אלפאנומרי (אינו אות מהאלפבית האנגלי או ספרה). התווים שאינם אלפאנומריים צריכים להיות מושלכים.
- נרמול הטקסט על ידי הפיכת כל תווי האותיות לאותיות קטנות (lowercase)

הערה: מכיוון שזהו מידע אמתי, הוא עלול להכיל תוכן משונה. למשל ייתכן כי יימצא תו של newline באמצע profileName, או מילים מאוד ארוכות. עליכם לכתוב את התכנית בצורה שתתמודד בצורה טובה עם הפתעות שכאלה.

לצורך הרצת התכנית, ניתן להוריד מאתר הקורס שני קבצים של נתונים גולמיים. קובץ אחד מכיל 100 ביקורות והקובץ השני 1000 ביקורות.

2. תיאור התרגיל

בהינתן קובץ הקלט עם הנתונים הגולמיים, עליכם ליצור אינדקס שיאפשר גישה יעילה למידע. על האינדקס להיות מאוחסן על הדיסק כדי שיהיה אפשר להשתמש בו כאשר מבצעים שאילתות על מוצרים שונים. לכן, קובצי האינדקס צריכים להישאר על הדיסק גם כאשר התכנית שלכם אינה רצה.

המבנה המדויק של האינדקס מהווה חלק מהחלטות התכנון שתקבלו. מימוש אינדקס עבור נתונים טקסטואליים יידון במסגרת השבועות הראשונים של הקורס. התפקיד שלכם הוא להשתמש ברעיונות שיילמדו (או להציע רעיונות אחרים משלכם) כדי לבנות את האינדקס שישרת את דרישות הפרויקט.

להלן כמה מגבלות על המימוש:

- אסור להשתמש במערכת של מסד נתונים כדי לשמור את המידע. עליכם לממש את האחסון בעצמכם.
- ניתן להשתמש ביותר מקובץ אחד כדי לאחסן את האינדקס. אולם, מספר הקבצים שייווצרו צריך להיות קבוע ולא תלוי במספר חוות הדעת או גודל המילון וכדומה.
- יש ליצור את האינדקס באופן כזה שלאחר יצירתו כבר אין צורך בקובצי הנתונים הגולמיים. כלומר, כל המידע הנדרש לצורך מענה לשיטות צריך להימצא באינדקס שנוצר.

בשלב זה אין צורך לדאוג לכך שתהליך בניית האינדקס יהיה יעיל (זו מטרתו של התרגיל השני). עם זאת, מבנה האינדקס והמימוש צריך להיות ניתן לשדרוג כך שיוכל לאפשר בנייה ואחסון של כמות נתונים גדולה מאוד כך שתוכלו להשתמש בו כמו שהוא כאשר נרצה לבנות אינדקס עבור כמות גדולה יותר של נתונים.

3. דרישות הקוד

התכנית תכיל לפחות את שתי המחלקות הבאות: (ככל הנראה התכנית תכלול מחלקות נוספות הנחוצות לצורך מימושן)

3.1. NonCompressedIndexWriter: בהינתן נתונים גולמיים, המחלקה תיצור אינדקס על הדיסק שאפשר יהיה לגשת אליו מאוחר יותר. כל הנתונים שישתמשו בהם מאוחר יותר צריכים להיות מאוחסנים באינדקס שעל הדיסק.

המילה "NonCompressed" בשם המחלקה מעידה על כך שהאינדקס הנבנה אינו דחוס. בתרגיל זה ניתן להניח כי כאשר בונים את האינדקס, כל הנתונים יכולים להיות מאוחסנים בזיכרון.

המחלקה מאפשרת גם למחוק את האינדקס מהדיסק על ידי מחיקת כל הקבצים מהספרייה של האינדקס.

3.2. IndexReader: לאחר שנוצר אינדקס על הדיסק ניתן להשתמש במחלקה כדי לגשת למגוון רב של נתונים הקיימים באינדקס. השתמשו במתודות המוגדרות כהכוונה והדרכה לתכנון מבנה האינדקס. כלומר מבנה האינדקס צריך לתמוך במימוש יעיל של מתודות אלו. ניתן להניח כי המתודות יופעלו רק לאחר שהאינדקס ייבנה על ידי NonCompressedIndexWriter. המימוש צריך להיות באופן שהוא יהיה יעיל גם כאשר האינדקס יכיל כמויות עצומות של נתונים.

תיאור של הממשק שצריך להיות ממומש מתואר בעמודים הבאים.

```
class NonCompressedIndexWriter:
     def __init__(self, inputFile, dir):
          """Given product review data, creates an on
          disk index
          inputFile is the path to the file containing
          the review data
          dir is the path of the directory in which all
          index files will be created
          if the directory does not exist, it should be
          created"""

     def removeIndex(self, dir):
          """Delete all index files by removing the given
          directory
          dir is the path of the directory to be
          deleted"""
```

```
class IndexReader
     def __init__(self, dir):
          """Creates an IndexReader object which will
          read from the given directory
          dir is the path of the directory that contains
          the index files"""

     def getProductId(self, reviewId):
          """Returns the product identifier for the given
          review
          Returns None if there is no review with the
          given identifier"""

     def getReviewScore(self, reviewId):
          """Returns the score for a given review
          Returns None if there is no review with the
          given identifier"""

     def getReviewHelpfulnessNumerator(self, reviewId):
          """Returns the numerator for the helpfulness of
          a given review
          Returns None if there is no review with the
          given identifier"""

     def getReviewHelpfulnessDenominator(self, reviewId):
          """Returns the denominator for the helpfulness
          of a given review
          Returns None if there is no review with the
          given identifier"""

     def getReviewLength(self, reviewId):
          """Returns the number of tokens in a given
          review
          Returns None if there is no review with the
          given identifier"""

     def getTokenFrequency(self, token):
          """Return the number of reviews containing a
          given token (i.e., word)
          Returns 0 if there are no reviews containing
          this token"""
```

```
def getTokenCollectionFrequency(self, token):
    """Return the number of times that a given
    token (i.e., word) appears in all the reviews
    indexed (with repetitions)
    Returns 0 if there are no reviews containing
    this token"""

def getReviewsWithToken(self, token):
    """Returns a series of integers of the form id-
    1, freq-1, id-2, freq-2, ... such
    that id-n is the n-th review containing the
    given token and freq-n is the
    number of times that the token appears in
    review id-n
    Note that the integers should be sorted by id
    Returns an empty Tuple if there are no reviews
    containing this token"""


def getNumberOfReviews(self):
    """Return the number of product reviews
    available in the system"""

def getTokenSizeOfReviews(self):
    """Return the number of tokens in the system
    (Tokens should be counted as many times as they
    appear)"""

def getProductReviews(self, productId):
    """Return the ids of the reviews for a given
    product identifier
    Note that the integers returned should be
    sorted by id
    Returns an empty Tuple if there are no reviews
    for this product"""
```

4. בדיקת התרגיל

בבדיקת התרגיל ייעשה שימוש בערכת נתונים קטנה בסדרי גודל של הערכות הנמצאות באתר הקורס.

בדיקת התרגיל נעשית בעזרת מערכת אוטומטית. כדי שבדיקת התרגיל שלכם לא תיכשל (ותגרום להורדה בציון) הקפידו היטב על ההנחיות שבסעיף 5.

5. הגשת התרגיל

- התרגיל יוגש דרך אתר המכללה בפורמט ZIP.
- עבור כל זוג יש להגיש רק הגשה אחת. שם הקובץ צריך להיות ID1_ID2.zip כאשר ID1 ו ID2 הם מספרי הזהות של הסטודנטים המגישים את הפרויקט.
- **קבצי הקוד צריכים לכלול שני קבצים עם השמות py.**NonCompressedIndexWriter **ו IndexReader.py. תכנית הבדיקה תייבא (import) קבצים אלו כך שחשוב שתקפידו על השמות הנכונים (כולל אותיות גדולות וקטנות) קבצים אלו יכילו את המחלקות שנדרשתם לפתח.**
- **במידה והפרויקט שלכם מכיל קבצי קוד נוספים (מחלקות נוספות), באחריותכם לייבא אותם (import) מתוך הקבצים NonCompressedIndexWriter.py ו IndexReader.py.**
- יש לוודא כי הקובץ שהועלה הוא בפורמט הנכון ומכיל את כל הקבצים הרלוונטיים.