תרגיל בית 4 – וקטורי מילים

מטלה זו עוסקת בייצוג באמצעות word embeddings. המשימה הספציפית זהה למשימה בתרגיל 3 - להבדיל בין טקסטים של שני דוברים בקורפוס הכנסת (ראובן ריבלין ויולי אדלשטיין).

בתרגיל זה השתמשו רק ב20,000 משפטים (10,000 לכל דובר).

לרשותכם קובץ של word embeddings של 100,000 המילים הנפוצות בעברית. הוקטורים אומנו על טקסט כללי גדול מאוד. כל מילה מיוצגת על ידי וקטור באורך 300.

כדי לטעון את קובץ הembeddings השתמשו בספריה gensim ובמודול KeyedVectors.

הפקודה לטעינת הקובץ:

```
from gensim.models import KeyedVectors

word2vec_model = KeyedVectors.load_word2vec_format(MODEL_FILE,
binary=False)
```

1. הכנה

לפני התחלת התרגיל בדקו שהמודל שנטען הגיוני. השתמשו בפונקציה similarity שמקבלת שתי מילים ומחזירה את קוסינוס הזווית בין שני הוקטורים שלהם. הערך הזה מייצג את הדמיון בין שתי המילים.

חישבו איזה זוג מילים דומות יותר "גבר","אישה" או "גבר","חתול"; הריצו את שתי הפקודות הבאות:

```
word2vec_model.similarity("גבר","אישה")

word2vec_model.similarity("גבר","חתול")
```

ובדקו האם השערתכם הייתה נכונה.

בנוסף, אתם יכולים לבצע פעולות על הוקטורים ולקבל תוצאות מעניינות כמו:

"ילדה" = "אישה" + "גבר"-"ילד"

זה נעשה בצורה הזו:

```
word2vec_model.most_similar(positive=['אישה', 'ילד'], negative=['גבר'])
```

2. דמיון בין משפטים

ענו על השאלה לפני מימוש התרגיל.

הפונקציה n_similarity מקבלת שני משפטים (בצורה של מערך) ומחזירה ערך שקובע עד כמה הם דומים. הדמיון נמדד ע"י חישוב קוסינוס הזווית בין שני הוקטורים הממוצעים של המשפטים.

בחרו כרצונכם 10 זוגות משפטים מהקורפוס של שני הדוברים ובדקו עד כמה הם דומים.

```
Sentence1 = "אני מסכים להעברת הנושא לוועדה ".split()

Sentence2 = "הצעת החוק מטעם הממשלה ".split()

word2vec_model.n_similarity(Sentence1, Sentence2)
```

פרטו בדו״ח את זוגות המשפטים ואת התוצאות שקיבלתם. נסו להסביר מה גורם למידת המרחק לקבוע דמיון בין משפטים. בפרט, נסו לאפיין באיזה מובן משפטים שהמרחק ביניהם קטן הם אכן דומים זה לזה.

3. סיווג

השתמשו במסווג logistic regression הממומש ב-scikit-learn. בנו את התכונות עבור המסווג בעזרת word embeddings כך: ה-feature vector של משפט הוא הממוצע המשוקלל של ערכי הוקטורים של <u>כל</u> המילים במשפט: $(\sum_i w_i \times v_i)/k$, כאשר:

- w_i הוא המשקל של המילה ה-i (סקלאר, ראו הסבר להלן)
- v_i הוא וקטור ה embedding של המילה ה-i (וקטור באורך 300)
- k הוא מספר המילים במשפט.

אם מילה במשפט לא נמצאת בקובץ המילים שיש עבורם וקטור – התעלמו ממנה (למשל, אפשר להגדיר וקטור של אפסים).

אמנו את המסווג עבור בשלוש דרכים שונות, סווגו את הקורפוס ודווחו על הדיוק המתקבל ב 10 fold cross validation.

א. לכל מילה המשקל הוא קבוע, 1, כלומר $w_i = 1$ לכל i (ממוצע רגיל)

ב. עבור 3 המילים הראשונות בכל משפט הגדירו משקל 1, עבור שאר המילים הגדירו משקל 0.1. באופן הזה אנחנו מניחים שלמילים בתחילת המשפט יש השפעה גדולה יותר על הסיווג מאשר שאר המילים.

ג. חישבו על דרך אחרת למשקל את וקטורי המילים כדי לקבל תוצאות סיווג טובות יותר.

דגשים:

- השתמשו בספרייה sklearn בשביל הסיווג.

הגשה:

הגישו קובץ פייתון בשם **hw4.py** שיופעל ע"י הפקודה

```
python hw4.py <input_dir1> <input_dir2> <word2vec> <outputFile>
```

כאשר:

input_dir1,input_dir2 הם תיקיות שבהן הקבצים של הדובר הראשון והשני בהתאמה

הקובץ **word2vec** הוא קובץ ה word embeddings

הקובץ **outputFile** הוא קובץ פלט

קובץ הפלט צריך להיות בפורמט הבא:

accuracy a: xxx

accuracy b: xxx

accuracy c: xxx

כאשר ה-accuracy הן התוצאות של המסווג עבור כל אחד מהסעיפים, בדיוק של 2 ספרות אחרי הנקודה.

דו"ח:

- ענו על סעיף 2 לפני התרגיל.

- האם שימוש בword embeddings הביא לשיפור לעומת תרגיל 3?

- פרטו על השיטה שלכם בסעיף ג'.

יש להקפיד על עבודה עצמאית. צוות הקורס יתייחס בחומרה להעתקות או שיתופי קוד.
תאריך הגשה: 23.12.2018 , עד השעה 23:59.

שאלות על התרגיל אפשר לשאול בפורום תרגילי בית.