דו"ח תרגיל בית 2

מודלי שפה – n-grams

**שלב 1**

**שלב 1.1:**

הפונקציה calculate\_prob\_of\_sentence מומשה, היא מקבלת כקלט משפט, ומחזירה את לוג ההסתברות של המשפט. היא מחשבת את לוג ההסתברות בעזרת נוסחת ה- Trigram מההרצאה, כאשר ההסתברויות מחושבות באמצעות MLE עם החלקת לפלס כנדרש.

בהינתן המשפט , הפונקציה קודם כל מוסיפה למשפט שני טוקני דמה , כך שהמשפט הופך להיות , ומחשבת את ההסתברות שלה באמצעות הנוסחה הבאה:

כאשר:

כך ש-

* להסביר על המקדמים...

**שלב 1.2:**

הפונקציה generate\_next\_token מומשה, היא מקבלת כקלט מחרוזת (רצף של טוקנים), ומחזירה Tuple בעל שני אלמנטים: הטוקן בעל ההסתברות הכי גבוהה שיהיה ההמשך של המחרוזת, וההסתברות שלו. לשם כך, הפונקציה עוברת על כל הטוקנים השונים (או באופן יותר מדויק, על כל ה- unigrams, מכיוון שהם שקולים), מוסיפה את הטוקין למחרוזת ושולחת את התוצאה של החיבור לפונקציה calculate\_prob\_of\_sentence שבתורה מחזירה את לוג ההסתברות של המשפט. הפונקציה תחזיר את הטוקין בעל לוג ההסתברות (log(probability)) הגדול ביותר. כמו כן, בשל כך שאנחנו נדרשים להחזיר את ההסתברות, אז הפונקציה מחזירה את .

**שלב 2**

**שלב 2.1:**

הפונקציה get\_k\_n\_t\_collocations, הפונקציה מחזירה אוסף של קולקציות העונות על הדרישות הבאות שמתקבלות כקלט: מספרם הוא k, אורך כל אחת מהן הוא n, ומספר המופעים של כל אחת לא פחות מ- t בקורפוס מסויים, שהוא pandas dataframe. כמו כן, הפונקציה מקבלת כקלט את סוג המדד שלפיו מחליטים איזה k מבין כל הקולקציות העונות על הדרישות צריך לבחור.

חילוץ הקולקציות מתבצע ע"י לעבור על כל המשפטים בקורפוס, ועבור כל אחת שאורכה לפחות n, הוא מחלץ את כל הקולקציות באורך n השונות (כל המחרוזות של טוקנים רצופים בגודל n).

הפונקציה אח"כ מסננת את הקולקציות ומשאירה רק אלה שמספר הופעתם בכלל הפרוטוקולים הוא לפחות t.

הפונקציה אח"כ מסדרת את הקולקציות מהכי נפוצה לפחות לפי המדד הנבחר. שני המדדים הם "Frequency" ו- "Tfidf", כאשר הראשון בוחר לפי כמות ההופעות, והשני לפי הנוסחה הנתונה. ומחזירה את ה- k קולקציות הראשונות בסדר. במקרה שאין k קולקציות כאלה, הפונקציה מחזירה את כל הקולקציות שהתקבלו.

לשם חישוב k- הקולקציות הכי נפוצות לפי מדד TF-IDF. הפונקציה משתמשת בשני מילונים של מילונים (מילונים שה- values שלהם מילונים). כאשר המילון הראשון משמש לשמירת עבור כל קולקציה את הפרוטוקולים (שמיוצגים ע"י מספרם) בהם הופיעה, ועבור כל אחד את כמות הופעותיו (כלומר המפתחות החיצוניות הם קולקציות, הערכים החיצוניים\ המפתחות הפנימיים הם מספרי פרוטוקולים, והערכים הפנימיים הם כמות ההופעות של אותה קולקציה בפרוטוקול). ובשני נשמרים עבור כל פרוטוקול את הקולקציות שהופיעו בו ואת מספר הופעתם (כלומר המפתחות החיצוניות הם מספרי הפרוטוקולים, הערכים החיצוניים\ המפתחות הפנימיים הם הקולקציות, והערכים הפנימיים הם כמות ההופעות של אותה קולקציה בפרוטוקול). המלון הראשון משמש לחישוב ה- Frequency עבור הקולקציה, ששווה לסכום הופעותיו בפרוטוקולים השונים, כמו כן, משתמשים בו בשליפת כמות הופעות הקולקציה בפרוטוקול מסוים בחישוב ה- tf, ובמספר הפרוטוקולים השונים שבהם הקולקציה מופיעה בחישוב ה- idf. בנוסף המלון משמש בשלב סינון הקולקציות. המלון השני משמש לחישוב הקולקציות השונות שבפרוטוקול מסויים בחישוב ה- tf, ובמספר הפרוטוקולים השונים בחישוב ה- idf.

**שלבים 2.2 – 2.4:**

הקוד שמדפיס את הדרוש נמצא ב- main ומשתמש בפונקציה מהשלב 2.1 לשם כך. הפלט מצורף כקובץ txt כנדרש.

**שלב 3**

**שלב 3.1:**

הפונקציה mask\_tokens\_in\_sentences מומשה. הפונקציה מקבלת כקלט רשימה של משפטים ומספר בטווח (0,100). הפונקציה ממסכת x% מהטוקנים במשפט (מחליפה אותם ב- [\*]). ומחזירה משפט חדש עם הטוקנים הממוסכים. במקרה ש- x% לא מספר שלם היא מקרבת אותו לשלם הכי קרוב, ובשל כך, משפט בעל פחות מ- 5 טוקנים (במקרה של המשפטים שלנו, משפט בעל 4 טוקנים) לא ימוסך אף טוקין בו. את הטוקנים שהיא ממסכת הפונקציה בוחרת באקראי.

**שלב 3.2:**

הקוד מומש ב- main, והפלט נמצא בשני קבצי txt מצורפים כנדרש.

**שלב 3.3:**

הקוד מומש ב- main, והפלט נמצא בקובץ txt המצורף כנדרש.

**שלב 3.4:**

הקוד מומש ב- main. להלן התוצאות שקיבלנו, שנמצאות גם כן בקובץ txt המצורף כנדרש.

הנוסחה שהשתמשנו בה לחישוב ה- Perplexity היא אותה נוסחה שהוזכרה בהרצאה עם השינוי שהשתמשנו רק בטוקנים שהחליפו טוקנים ממוסכים. מתמטית, נסמן ב- s את המשפט, וב- את הטוקנים שהחליפו טוקנים ממוסכים. כמו כן, נסמן ב- s[:i] את החלק במשפט עד הטוקן ה- i (לא כולל). אז הנוסחה היא: