**יבש 1 – NLP**

מגישים: אמיר בלדר – 204179659, רועי גנץ - 204506349

חלק 1

1. בעיית ה- OOV הינה בעיה מוכרת, כיצד נוכל לסווג מילים שלא ראינו קודם לכן באימון?   
   ישנם מספר סוגים של "משפחות" מאפיינים שיכולים לעזור בכדי לנתח בעיית OOV.
2. מאפיינים שתמסתמכים על ה-tags הקודמים למילה שלא נראתה.
3. מאפיינים שתמסתמכים על ה-tags האוחרים למילה שלא נראתה.
4. מאפיינים שתמסתמכים על המילים שלפני המילה שלא נראתה.
5. מאפיינים שתמסתמכים על המילים שלפני המילה שלא נראתה.

כמו שבני אדם לא בהכרח מבינים את כל המילים במשפט שהם קוראים ויכולים להבין מילים מההקשר שלהן, כך גם המודל יכול לעשות בעזרת פרמטרים מתאימים. כלומר, אפשר להתסמך על המילים והתיוגים שהיו לפני ואחרי המילה הלא מוכרת בכדי לסווגה נכונה.

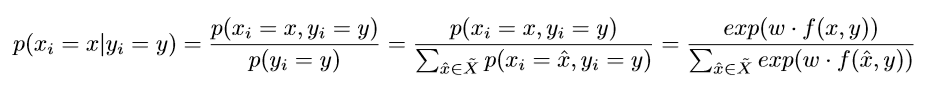
בנוסף, אנחנו יכולים להסתכל גם על משפחות מאפיינים שבהן יש משמעות לחלקים מסוימים במילה, לדוגמא:

1. שימוש באותיות גדולות/קטנות – מה שיכול לרמז על שם של דבר כלשהו.
2. שימוש בתחיליות/סופיות שחוזרות על עצמן.
3. שימוש במספרים בתוך מילים.

שימוש במאפיינים מעין אלה יהווה פתח לניתוחן של מילים רבות שלא ראינו עד כה, ולהוות לפחות ניתוח ראשוני ובסיסי עבור אלה בעיקר השימוש בתחיליות/סופיות שחוזרות על עצמן יכול להיות יעיל, שכן לרוב מדובר בתוספת למילה אחרת, שיתכן שכן הופיעה מוקדם יותר במודל.

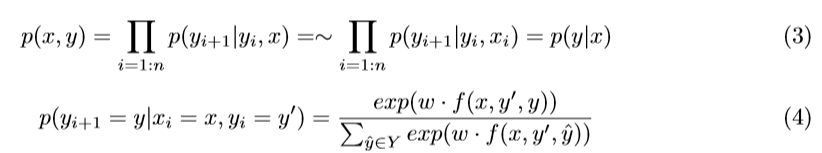
1. מדובר בעצם במרחב האפשרויות של וקטורי הייצוג. מרחב זה יהיה מורכב מכמות התיוגים והמילים האפשרויות. לכל מילה יש 100,000 אפשרויות, ולכל תיוג 25 אפשרויות ולכן:

כלומר, שמספר המאפיינים האפשריים הוא .

1. הפרמטרים עבור המודל לא יכולים להיות מחושבים בצורה יעילה בעבור מודל בגודל שכזה, שכן מרחב האפשרויות של כל מאפיין הינו גדול **מאוד**. במודל "סביר" ישנם בערך אפשרויות לכל וקטור מאפיינים. אנחנו מבינים שבמקרה זה קיבלנו מספר אפשרויות גדול בעשרות מונים, כלומר במספר סדרי גודל לא מבוטל. המשמעות תהיה כמובן שברוב הכולל של המקומות בכל וקטור מאפיינים יהיו אפסים. זה המצב ברוב וקטורי המאפיינים, אך כפי שציינו במקרה זה יהיו **מספר סדרי גודל יותר של אפסים**, מה שיהפוך את החישובים עצמם ללא יעילים ואת ריצת האלגוריתם ללא יעילה.
2. נתחיל מלציין עובדה חשובה – פונקציית ה-exp הינה פו' מונטונית עולה, שהינה גדולה מאפס. על כן, השימוש בה, יחד עם הנרמול במכנה, משאיר את התוצאה בתחום [0,1], כיאה להסתברות. בנוסף, בשל היותה מונוטונית עולה, השימוש בה לא ישנה את ה"מגמה" ואת מציאת הכיוון הנכון של הגרדיאנט לו נזדקק לעדכון המשקלים בהמשך. אם כן, השימוש ב-exp משאיר אותנו בתחום בין [0,1], ואם לא היינו משתמשים בה היינו יכולים לקבל גם תוצאות שליליות, מה שיכול היה לסבך את חישוב הגרדיאנט בהמשך.
3. בהנתן בחירה בין שימוש בייצוג בינארי עבור מאפיינים לבין ייצוג בעזרת מספרים, נעדיף להשתמש בייצוג בינארי. זאת ממספר טעמים. הראשון שבהם הוא עניין "קרבת" המאפיינים. כלומר, בהנתן שיש לנו onehot vector, כל המאפיינים קרובים זה לזה במידה שווה, בהיבט של מרחק האמינג. לעומת זאת, בייצוג מספרי, נקבל כי מאפיינים מסוימים קרובים יותר למאפיינים אחרים ללא הצדקה בהכרח. לדוגמא, אם התיוג NN (שם עצם) יקבל את הספרה 1, התיוג JJ (תואר) את הספרה 2 והתיוג VB (פועל) את הספרה 3. נקבל בפועל כי תואר דומה יותר לפועל מאשר ששם עצם דומה לפועל. כמובן שזאת לא בהכרח הנחה נכונה...   
   הייצוג הבינארי ימנע מצבים מעין אלה. בנוסף, הייצוג הבינארי ניתן לדחיסה בצורה יעילה יחסית בעזרת ספריות מוכרות בפייתון, כך ששימוש נבון בו הוא לרוב יעיל למדי.
4. נבחן את הנוסחא:  
   

נשים לב כי החישוב של המכנה הוא בעל אפשרויות רבות מאוד:  


כלומר, החישוב של המכנה ידרוש מעבר על המון אפשרויות שונות, ביצוע exp על כל אחת מהן ולאחר מכן סכימתן. כפי שציינו קודם לכן, מדובר על סדרי גודל גדולים למדי של אפשרויות שונות, ועל כן, החישוב עצמו יארך זמן רב מאוד. על כן, ככל הנראה שלא יהיה משתלם להשתמש בנוסחא המדוברת.

1. עתה אנו מסתכלים על נוסחאות אחרת:  
   

כלומר שבמקרה הזה אנו מסתכלים על המכנה:   
כאשר הוא קטן משמעותית מאשר . מהווה את מרחב התיוגים השונים, כאשר הדוגמא שראינו בשאלה 2, גודל המרחב היה 25 תיוגים, ובתרגיל הבית שבו אנו השתמשנו במרחב תיוגים סטדנדרטי הוא היה בגדול 45. על כן, מדובר עתה בבעיה שניתן באמת לחשבה בצורה סבירה.

1. המודל של HMM הינו מודל גנרטיבי והמודל של MEMM הינו דיסקרימינטיבי. ככלל אצבע, מודל גנרטיבי משערך את המודל ההסתברותי של כל אחד מה-class-ים האפשריים בבעיה, בעוד שמודל דיסקרימנטיבי משערך את הגבולות המבדילים בין ה-class-ים האפשריים. אם כן, שימוש במודל גנרטיבי יכול להיות טוב יותר למשימות מסוימות, לדוגמא אם נרצה ליצור משפטים.   
   בנוסף, HMM הוא מודל פשוט הרבה יותר להבנה, בדיוק בשל הסיבה שהוא משערך את ההסתברות של כל class ישירות. כך בעצם קל יותר לנתח את התוצאות ולהבין איפה המודל טעה, לפחות קל יותר מאשר MEMM.