עיבוד שפות טבעיות - תרגיל 3

סמסטר א' תשפ"ה

מגישים

שם	ת.ז
אור ישלח	314861584
עודד דואק	039945522

חלק 1 - קוד

שלב 1: הגדרת המחלקות

1. שני הדוברים שיהוו את המחלקות שלנו:

   1. ראובן ריבלין
   2. א' בורג

2. לאחר מציאת שני הדוברים המרכזיים בדקנו אם יש שמות דוברים אחרים שמאוד דומים לשני דוברים אלו. הוספנו ל-DataFrame עמודה בשם label ועברנו על כל השורות בקורפוס. במידה ושם דובר דומה מאוד לדובר הראשון (הדובר בעל המספר הגדול ביותר של משפטים) אז רשמנו first בשורה המתאימה, בעמודה label. בצורה זהה, במידה ושם דובר דומה מאוד לדובר השני (הדובר בעל המספר השני הגדול ביותר של משפטים) אז רשמנו second בשורה המתאימה, בעמודה label. אחרת, רשמנו other. יצרנו כמה DataFrames כאשר כל אחד מכיל את כל השורות בעלות ערך זהה בעמודה label (למשל DataFrame של כל השורות עם המחרוזת first בעמודה label). לאחר מכן איזנו ואיחדנו אותן.

הסבר לגבי ההחלטה אם השם דומה מאוד:
אם השם של הדובר זהה לשם שאותו בודקים (נקרא לו מועמד) אז ברור שמדובר באותו דובר. אחרת, חילקנו כל שם למילים המרכיבות אותו והורדנו גרש אם הוא מופיע בסוף מילה (כלומר, אם הוא קיצור של מילה).

אתחלנו 2 קבוצות: מילים זהות ומילים בשם המועמד שדומות למילים בשם הדובר (נקרא לה קבוצת המילים הדומות).
אתחלנו דגל מציאת מילים זהות ב-False ואתחלנו באפס מונה שסופר מילים זהות/דומות (נקרא לו מונה מילים דומות).
הרעיון שלנו הוא ששמות שונים יחשבו לאותו שם דובר אם לפחות מילה אחת בשם המועמד זהה למילה בשם הדובר ולפחות מילה נוספת בשם המועמד זהה או דומה למילה נוספת בשם הדובר.

עברנו על כל המילים שבשם של הדובר בלולאה ועבור כל מילה קראנו לפונקציה שעוברת על כל המילים של השם המועמד:
אם המילה בשם המועמד נמצאת בקבוצת מילים שכבר נמצאו זהות למילים בשם של הדובר אין צורך לבדוק אותה שוב ונעבור למילה הבאה בשם המועמד.

אם המילה זהה למילה בשם הדובר נחזיר שקיבלנו זהות, נסמן בדגל שיש זוג מילים זהות, נוסיף לקבוצת המילים הזהות ונוסיף אחד למונה מילים דומות.

אם המילה בשם המועמד לא נמצאת בקבוצת מילים דומות והמילה מוכלת בהתחלת המילה בשם הדובר או שהמילה בשם הדובר מוכלת בהתחלת מילה זו אז נוסיף את המילה לקבוצת המילים הדומות ונוסיף אחד למונה מילים דומות.
לאחר מעבר על כל המילים בשם הדובר, אם מצאנו שיש מילים בקבוצת המילים הדומות הנמצאות גם בקבוצת המילים הזהות (מקרה קצה שבו מתחשבים פעמיים במילים אלו) אז נאפס את קבוצת המילים הדומות, נוריד בהתאם את המונה ונתחיל שוב בלולאה על המילים בשם הדובר כאשר הפעם נתעלם מכל המילים שבקבוצת המילים הזהות.

לבסוף, אם הדגל של המילים הזהות ב-True (כלומר קיימות לפחות זוג מילים זהות) ומונה המילים הדומות גדול מאחד אז נחזיר True, כלומר שם המועמד מתאים לשם הדובר (יש לפחות זוג מילים זהות ולפחות זוג נוסף שהן זהות או דומות). במקרה ותנאים אלו לא מתקיימים נחזיר False.

שלב 2: איזון המחלקות

מספר הפריטים בכל מחלקה לפני ואחרי ה- down-sampling:

Binary classification

Class	Before down-sampling	After down-sampling
ראובן ריבלין	3103	2282
א' בורג	2282	2282

Multi-class classification

Class	Before down-sampling	After down-sampling
ראובן ריבלין	3103	2282
א' בורג	2282	2282
other	102374	2282

שלב 3: יצירת וקטור מאפיינים (vector feature)

בחרנו להשתמש ב-Tfidf. הסיבה לכך היא ששיטה זו בדרך כלל טובה יותר בשביל סיווג טקסט. בשיטה זו יש התחשבות במספר המופעים של מונח במסמך ובמספר המסמכים שבהם המונח מופיע, כך ניתן משקל גבוה יותר למילים ייחודיות ומשקל נמוך יותר למילים נפוצות. זה מאפשר לזהות דובר לפי מילים ייחודיות אלו ובכך לבצע סיווג טוב יותר של המשפטים.

במקרה שלנו, התקבלו תוצאות דומות של classification report בסיווג לאחר שימוש ב-CountVectorizer ובסיווג לאחר שימוש ב-Tfidf (אם כי טובות בקצת עם Tfidf במקרה שבו נבחר לסיווג בשלב 5). דמיון זה יכול לרמז על איזון בשימוש המילים בקורפוס.

שלב 4: אימון

Classification report המפרט את תוצאות ההערכה עבור כל משימה, עבור כל מסווג ועבור כל ווקטור מאפיינים:

```
================ BINARY CLASSIFICATION EVALUATION ==========
=== TF-IDF Features ===
KNN Results:
              precision    recall  f1-score   support

       first      0.838     0.865     0.851      2282
      second      0.860     0.833     0.846      2282

    accuracy                          0.849      4564
   macro avg      0.849     0.849     0.849      4564
weighted avg      0.849     0.849     0.849      4564

LogisticRegression Results:
              precision    recall  f1-score   support

       first      0.817     0.944     0.876      2282
      second      0.934     0.788     0.855      2282

    accuracy                          0.866      4564
   macro avg      0.875     0.866     0.866      4564
weighted avg      0.875     0.866     0.866      4564

=== Custom Features ===
KNN Results:
              precision    recall  f1-score   support

       first      0.915     0.918     0.917      2282
      second      0.918     0.915     0.916      2282

    accuracy                          0.917      4564
   macro avg      0.917     0.917     0.917      4564
weighted avg      0.917     0.917     0.917      4564

LogisticRegression Results:
              precision    recall  f1-score   support

       first      0.912     0.826     0.867      2282
      second      0.841     0.920     0.879      2282

    accuracy                          0.873      4564
   macro avg      0.876     0.873     0.873      4564
weighted avg      0.876     0.873     0.873      4564
```

```
================ MULTI-CLASS CLASSIFICATION EVALUATION ======
=== TF-IDF Features ===
KNN Results:
              precision    recall  f1-score   support

       first      0.722     0.511     0.598      2282
       other      0.585     0.808     0.679      2282
      second      0.849     0.776     0.811      2282

    accuracy                          0.698      6846
   macro avg      0.719     0.698     0.696      6846
weighted avg      0.719     0.698     0.696      6846

LogisticRegression Results:
              precision    recall  f1-score   support

       first      0.710     0.669     0.689      2282
       other      0.655     0.779     0.712      2282
      second      0.896     0.780     0.834      2282

    accuracy                          0.742      6846
   macro avg      0.754     0.742     0.745      6846
weighted avg      0.754     0.742     0.745      6846

=== Custom Features ===
KNN Results:
              precision    recall  f1-score   support

       first      0.724     0.829     0.773      2282
       other      0.824     0.654     0.729      2282
      second      0.829     0.880     0.854      2282

    accuracy                          0.788      6846
   macro avg      0.792     0.788     0.785      6846
weighted avg      0.792     0.788     0.785      6846

LogisticRegression Results:
              precision    recall  f1-score   support

       first      0.549     0.621     0.582      2282
       other      0.613     0.405     0.488      2282
      second      0.665     0.804     0.728      2282

    accuracy                          0.610      6846
   macro avg      0.609     0.610     0.599      6846
weighted avg      0.609     0.610     0.599      6846
```

שמנו לב לכך שבחירת פרמטרים שונה מברירת המחדל לא גרמה לשיפורים גדולים בתוצאות. בכל זאת, ניסינו לבחור פרמטרים כך שנקבל שיפור מסוים גם אם הוא לא גדול.

K-Nearest Neighbors

ניסינו לשנות את מספר השכנים (n_neighbors) לערכים 3, 7, 9, 11, 13, 15, 17 (ברירת המחדל היא 5). ערכים קטנים עלולים לגרום לרגישות גדולה יותר לרעש. ערכים גדולים יכולים לעזור לסווג (יותר שכנים משתתפים) אך גורמים גם לחישוב איטי יותר ועלולים לגרום לטעויות גדולות יותר.
ניסינו לשנות גם את המשקל שניתן לשכנים (weights) ל-distance, כלומר ששכנים קרובים יותר ישפיעו יותר.

לבסוף בחרנו ב-13 בתור מספר השכנים ובחרנו להשתמש ב-distance וזאת משום שעם ערכים אלו התקבל שיפור טוב (של 0.08 ב-accuracy) במקרה של Multi Class עם Tfidf ושאר התוצאות נשארו דומות למקרה של שימוש בפרמטרים ברירת המחדל.

Logistic Regression

ניסינו להשפיע על הרגולריזציה על ידי ערכי C שונים: 0.1, 0.5, 3, 7, 10, 100 (ברירת המחדל היא 1).התלבטנו בין C=1 (ברירת המחדל) לבין C=3 שנתנו תוצאות (classification report) דומות (בחלק מהמקרים הייתה עדיפות קטנה לאחד ובחלק לאחר). לבסוף בחרנו ב-C=1 מפני שעם C קטן יותר מתקבלת רגולריזציה טובה יותר מה שיכול להקטין overfitting.
ניסינו גם ערך penalty שונה (l1) וערכי solver שונים (saga, liblinear).
כדי שתהיה התכנסות במקרה של Custom feature vector שינינו את מספר האיטרציות המקסימלי ל-4000.
ברירות המחדל מתאימות להרכבי המשפטים וגודל המידע שלנו (למעט מספר איטרציות מקסימלי).

שלב 5:

בחרנו להשתמש ב- LogisticRegression עם Tfidf. תוצאות הולידציה יצאו לנו פחות טובות במקרה של Custom feature vector אם משתמשים רק בעמודה של המשפטים בלי עמודות נוספות.

חלק 2 - שאלות

שאלה 1

- מה הם האתגרים שיכולים להיווצר בשימוש במחלקה "אחר" במשימת הסיווג?

המחלקה הזאת מכילה משפטים שהם לא של 2 הדוברים המרכזיים. לפני down sampling היא גדולה מאוד ובשביל ליצור איזון מסירים ממנה המון משפטים מה שמקטין מאוד את גודל המידע של האימון. בנוסף, היא מכילה משפטים של הרבה דוברים שונים ולכן היא הטרוגנית ויכולה להכיל הרבה סגנונות שונים מה שעלול לגרום למודל קושי להבין את הסגנון של מחלקה זו. בנוסף, סיווג משפט למחלקה זו לא מאפשר לדעת מי הדובר הספיציפי של המשפט.

שאלה 2

- נניח שאתם משתתפים בתחרות מודלים לחיזוי בינארי שבה אם המודל שלכם יחזה נכון את כל הדוגמאות של הדובר הראשון, תקבלו פרס כספי גדול, ואם המודל שלכם יטעה על אפילו דוגמה אחת של הדובר הראשון תקבלו קנס כספי גבוה.
  מבין המדדים המופיעים ב report classification, איזה מדד תרצו למקסם? איזה מהמודלים שאימנתם תבחרו למטרה זו? הסבירו.

אם נקבל פרס כספי גדול במידה ונחזה נכון את כל הדוגמאות של הדובר הראשון ובמידה ונטעה על אפילו דוגמא אחת של הדובר הראשון נקבל קנס כספי גבוה, אז נרצה למקסם את מדד ה-recall של הדובר הראשון.
אם FN של הדובר הראשון הוא 0 אז זה אומר שאין טעויות בסיווג של משפטים של הדובר הראשון. TP גבוה מראה שחזינו נכון הרבה דוגמאות של הדובר.

הנוסחא למדד recall היא $\frac{TP}{TP+FN}$ ולכן לפי ההסבר נרצה מקסום של מדד זה בשביל הדובר הראשון (נרצה ש-FN יהיה קטן ככל האפשר ו-TP גדול ככל האפשר).

נבחר במודל LogisticRegression עם שימוש ב-Tfidf משום שבמקרה זה נקבל ערך recall של 0.944 והוא גדול משאר ערכי recall.

שאלה 3

- ענו שוב על 1 כאשר שינו את החוקים בתחרות וכעת אם המודל שלכם יסווג נכון את כל הדוגמאות של שני הדוברים תקבלו פרס כספי גבוה, אבל אם המודל שלכם יסווג אפילו דוגמה אחת בצורה לא נכונה, תקבלו קנס כספי גבוה.

אם נקבל פרס כספי גדל במידה ונסווג נכון את כל הדוגמאות של שני הדוברים ובמידה ונטעה על אפילו דוגמא אחת נקבל קנס כספי גבוה, אז נרצה למקסם את מדד accuracy (שדומה מאוד לתוצאות של macro avg f1 משום שהמחלקות מאוזנות). למעשה, נרצה ערכי FN ו-FP קטנים ככל האפשר מה שממקסם מדד accuracy (לפי הנוסחא $\frac{TP+TN}{TP+TN+FP+FN}$).

ערכי FN ו-FP קטנים ככל האפשר ממקסמים גם את precision ו-recall ולכן למעשה גם את f1-score. נרצה למקסם זאת לשני הדוברים ולכן במידה והמחלקות מאוזנות (המקרה שלנו) נרצה למקסם את f1-score הממוצע (macro avg f1).

נבחר במודל KNN עם שימוש ב-custom feature vector משום שבמקרה זה נקבל ערך accuracy של 0.917 והוא גדול משאר ערכי accuracy. אם אסור להשתמש בוקטור הזה בשאלה אז נבחר במודל LogisticRegression עם שימוש ב-Tfidf משום שבמקרה זה נקבל ערך accuracy של 0.866 והוא גדול מערך accuracy של KNN עם וקטור זה.

שאלה 4

- הסבירו מה היתרונות והחסרונות של שיטת הcross validation על פני חלוקה פשוטה למחלקת אימון ובדיקה. איזו משיטות ההערכה אמינה יותר לדעתכם?

יתרונות:

- שימוש נכון יותר במידע – הדגימות משמשות גם לצורך אימון וגם לצורך בדיקה (כאשר הן ב-fold של הבדיקה) לעומת המקרה של חלוקה לאימון ובדיקה שבו יש שימוש בפחות מידע לצורך האימון והבדיקה (מה שעלול לגרום ל-overfitting).
- Reduce Variance- משום שב-cross validation הדגימות משמשות גם לצורך אימון וגם לצורך בדיקה ומשום שמתבצע אימון ובדיקה מספר פעמים אז מתקבל variance נמוך יותר והערכה רובסטית יותר.

חסרונות:

- זמן ריצה – משום שכל המידע משמש לאימון ובדיקה ומבצעים מספר איטרציות (כמספר folds) אז יש הרבה יותר חישובים וזמן הריצה גדול יותר מאשר אימון פעם אחת ובדיקה פעם אחת לאחר חלוקה פשוטה.
- מסובך יותר - מימוש cross validation מסובך יותר למימוש מאשר המקרה של חלוקה פשוטה לאימון ובדיקה.

Cross-validation אמינה יותר מחלוקה פשוטה, מאחר והיא מספקת הערכה טובה יותר של הביצועים הצפויים של המודל בפועל (כפי שניתן להבין מהחלק של היתרונות).

שאלה 5

- הסבירו מהם היתרונות והחסרונות של שני סוגי המסווגים KNN, LogisticRegression בהם השתמשתם. האם לדעתכם אחד מהם עדיף על פני השני, עבור משימות הסיווג שבתרגיל?

Logistic Regression:

יתרונות

- יעיל יחסית – זמן ריצה טוב יותר מאשר KNN
- מתאים לשימוש על מידע בעל מימדים גבוהים
- רגולריזציה – על ידי שינוי ערך המקדם C ניתן לצמצם overfitting

חסרונות

- מתאים בעיקר להפרדה לינארית
- רגיש ל-Outliers

KNN:

יתרונות

- אין צורך באימון
- פשוט יחסית להבנה ולמימוש
- גמיש - יכול להתאים גם למקרים שבהם אין הפרדה ליניארית

חסרונות

- סיבכויות זמן גבוהה – יש צורך בחישובי מרחקים משכנים בזמן הסיווג
- לא מתאים במימדים גבוהים
- רגיש ל-Outliers

LogisticRegression בדרך כלל עדיף לסיווג טקסט על פני KNN. זאת בזכות היעילות שלו ויכולת הסיווג הטובה שלו גם במימדים גבוהים.

במקרה שלנו LogisticRegression עדיף כאשר משתמשים בוקטור Tfidf (ניתן לראות תוצאות ולידציה טובות יותר). כאשר משתמשים ב- custom feature vector אז דווקא מתקבלות תוצאות טובות יותר עם KNN.

שאלה 6

- יחידת הסיווג בתרגיל היא משפט אחד. אם במקום זאת, היינו מחליטים על יחידת סיווג שמאחדת יחד מספר משפטים מאותה מחלקה, כך שכל דוגמה לסיווג הייתה מקבץ של משפטים. מה היו היתרונות והחסרונות בכך? התייחסו בתשובתכם ליחידות סיווג של 2 ,5, 10 , 100 משפטים.

- **יתרונות**:
  - **הקשר רחב , זיהוי דפוסים מורכבים**: מקבוצת משפטים של אותו דובר ניתן לקבל הקשר ולחלץ סגנון דיבור מה שיכול לעזור בסיווג.
  - **הקטנת רעש**: מקבץ של משפטים יכול לעזור בהקטנת רעשים שנוצרים משימוש בדוגמאות קטנות יותר.
- **חסרונות**:
  - **מורכבות:** יש להחליט על גודל יחידות הסיווג וגם על איזה משפטים לקבץ ביחד (באקראי או תוך שמירה על הקשר).
  - **פחות דגימות:** איחוד משפטים ליחידות סיווג גדולות (למשל של 10 או 100) גורם למספר הדגימות לקטון מאוד.
  - **רעשים בסגנון הדובר:** שימוש ביחידות סיווג גדולות מדי (למשל מקבץ של 100) יכול להכניס רעשים אם הדובר משנה את סגנונו.
- **יחידות סיווג שונות**:
  - **2 משפטים**:
    קרוב לסיווג משפט יחיד, מתאים לקורפוסים קטנים או למטרות שבהן נדרש רק הקשר מינימלי.
  - **5 משפטים**:
    מאזן טוב לקורפוסים בינוניים עד גדולים, מספק הקשר משמעותי.
    יכול לכלול רעש, אך הוא יהיה מועט בד"כ.
  - **10 משפטים**:
    מספק הקשר רחב, אך מקטין את מספר הדגימות בצורה משמעותית.
    מתאים לקורפוסים גדולים.
  - **100 משפטים**:
    מקטין את מספר הדגימות בצורה משמעותית מאוד. עלול להכניס הרבה רעש שלא מאפשר לבצע סיווג.

שאלה 7

- איזה גודל של יחידת סיווג עדיף לדעתכם (1, 2, 5, 10, 100) במשימות שלנו? הסבירו.

במקרה שלנו כמות המידע לא גדולה ולא כדאי לצמצם אותה מאוד, מה שעלול לקרות אם משתמשים ביחידות סיווג גדולות. לכן, לדעתנו לא כדאי להשתמש ביחידות סיווג 10 ו-100.

לדעתנו הגודל המומלץ ליחידת סיווג במשימות שלנו הוא **2 או 5 משפטים**: יכול לעזור בזיהוי דפוסים מורכבים ובהקטנת רעש מבלי להקטין מדי את כמות הדגימות לאימון.