עיבוד שפות טבעיות - תרגיל 3 - סיווג

שלב 1 - הגדרת המחלקות

עשינו קריאה לדאטה וחילקנו אותה בהתאם לפי עמודת סוג הפרוטוקול לשני סוגים:

Committee ו Plenary.

שלב 2 - חלוקה ליחידות סיווג

בשלב זה מימשנו פונקציית devide_into_chunks:

פונקציה זו מאחדת משפטים לצנקים בגודל 5.לאחר קריאת הדאטה וסיווגה לשני הסוגים.אנו משתמשים בפונקציה זו כדיליצור צנקים הכוללים 5 משפטים לשני סוגי הפרוטוקולים.

שלב 3 - איזון המחלקות

השתמשנו ב Random.sample(target size) כאשר:

target size = מספר הפריטים במחלקה הקטנה

כדי לבחור באופן רנדומלי פריטים מהמחלקה הגדולה כמספר הפריטים במחקלה הקטנה.

מספר הפריטים בכל מחלקה לפני ואחרי ה sampling-down:

	לפני	אחרי
Plenary	14104	5895
Committee	5895	5895

שלב 4 - יצירת וקטור מאפיינים

1.ניסינו להשתמש בשניהם אבל בסוף החלטנו להשתמש ב TFIDF מכיוון שסיפק לנו תוצאות יותר טובות.

TFIDF מתחשב בתדירות המונחים ב chunk וגם בחשיבותם בכל הקורפוס.דבר זה מסייע לזהות מונחים מיוחדים המבדילים לסיווג ובמקביל מנרמל את חשיבות המונחים לפי הופעותם בקורפוס.

בניגוד ל CountVectorizer,שסופר אך ורק מופעי מונחים ללא נורמליזציה.

TFIDF מתייחס לתדירות המונחים בקטע וגם לחשיבותם בכל הקורפוס. בכך הוא משפר את יכולת המסווג לזהות פיצרים חשובים לסיווג.

2.בחרנו ב 105 פיצרים להיות: 100 המילות החשובות ביותר ,עוד פיצר שהוא כמות המילים ב chunk ,עוד פיצר שהוא ממוצע גודל מילה ,עוד פיצר שהוא כמות המספרים ופיצר שהוא כמות punctuation mark ופיצר שהוא שונות אורך מילה.

פיצרים אלה הם גורמים חשובים כי הם משתנים מסוג פרוטוקול לאחר.

שלב 5 - אימון

1.אימון שני המסווגים:

```
svm_classifier = SVC(kernel="linear", random_state=42)
knn_classifier = KNeighborsClassifier(n_neighbors=51, n_jobs=-1)
```

.2

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1,
stratify=y, random_state=42, shuffle=True)
```

TFIDF:

Train-Test Split

```
SVM
 Train-Test Split:
              precision    recall  f1-score   support

   committee       0.93      0.94      0.93       590
     plenary       0.94      0.93      0.93       589

    accuracy                           0.93      1179
   macro avg       0.93      0.93      0.93      1179
weighted avg       0.93      0.93      0.93      1179

KNN
 Train-Test Split:
              precision    recall  f1-score   support

   committee       0.85      0.94      0.89       590
     plenary       0.93      0.83      0.88       589

    accuracy                           0.89      1179
   macro avg       0.89      0.89      0.89      1179
weighted avg       0.89      0.89      0.89      1179
```

Cross-Validation

```
SVM
Cross - Validation:
              precision    recall  f1-score   support

   committee       0.92      0.94      0.93      5895
     plenary       0.94      0.91      0.93      5895

    accuracy                           0.93     11790
   macro avg       0.93      0.93      0.93     11790
weighted avg       0.93      0.93      0.93     11790

KNN
Cross - Validation:
              precision    recall  f1-score   support

   committee       0.85      0.95      0.89      5895
     plenary       0.94      0.83      0.88      5895

    accuracy                           0.89     11790
   macro avg       0.89      0.89      0.89     11790
weighted avg       0.89      0.89      0.89     11790
```

Our Feature Vector:

Train-Test Split

```
SVM
 Train-Test Split:
              precision    recall  f1-score   support

   committee       0.83      0.87      0.85       590
     plenary       0.87      0.82      0.84       589

    accuracy                           0.85      1179
   macro avg       0.85      0.85      0.85      1179
weighted avg       0.85      0.85      0.85      1179

KNN
 Train-Test Split:
              precision    recall  f1-score   support

   committee       0.75      0.73      0.74       590
     plenary       0.74      0.76      0.75       589

    accuracy                           0.74      1179
   macro avg       0.74      0.74      0.74      1179
weighted avg       0.74      0.74      0.74      1179
```

Cross-Validation

```
SVM
Cross - Validation:
              precision    recall  f1-score   support

   committee       0.81      0.86      0.84      5895
     plenary       0.85      0.80      0.83      5895

    accuracy                           0.83     11790
   macro avg       0.83      0.83      0.83     11790
weighted avg       0.83      0.83      0.83     11790

KNN
Cross - Validation:
              precision    recall  f1-score   support

   committee       0.75      0.75      0.75      5895
     plenary       0.75      0.75      0.75      5895

    accuracy                           0.75     11790
   macro avg       0.75      0.75      0.75     11790
weighted avg       0.75      0.75      0.75     11790
```

שלב 6 - סיווג

בשלב זה מה שנתן לנו תוצאות הכי טובות זה:

TFIDF+SVM

תוצאות הסיווג הודפסו בהתאם לקובץ טקסט כפי שמבוקש.

שאלות סיכום:

1.כן היה הבדל ב precision ו recall בין שתי המחלקות :

ה precision עבור Committee נמוך במעט בהשוואה ל precision עבור Plenary.

זה מצביע על כך שהמודלים מדוייקים מעט יותר בזיהוי מקרים השייכים ל Committee בהשוואה למחלקה השנייה Plenary.

זאת אומרת שכאשר המודל חוזה שמסמך הינו שייך ל Committee, יש סיכוי מעט יותר גבוה שהוא עשוי להיות שגוי בהשוואה לכך שהוא חוזה מסמך Plenary.

ה recall עבור Committee גבוה יותר בהשוואה ל recall עבור Plenary.

זה מצביע על כך שהמודלים מצליחים יותר בלזהות נכון מופעים של מסמכים שבאמת שייכים ל Committee.

במילים אחרות, כאשר המודל חוזה שמסמך שייך ל Committee,סביר יותר שהוא יהיה נכון בהשוואה כשהוא חוזה מסמך שייך ל Plenary.

2. קיבלנו תוצאות שאכן דומות בשתי דרכי דיוק המסווגים.

ב train-test split 10%-90% אנו מאמנים 90% מהדאטה ובודקים על 10% ,

לעומת זאת ב cross validation שזה לעשות 10 פעמים : לאמן את 90% ולבדוק על 10%,

ב train-test split 10%-90% אימננו 90% מהדאטה ובדקנו 10% ,

לעומת זאת ב cross validation עשינו את אותו דבר שעשינו ב train-test split 10%-90% אבל כל פעם על 90% מהדאטה ששונות מהפעם הקודמת

בגלל שקיבלנו תוצאות דומות בשתי שיטות השערוך ,זה מעיד על כך שהמודל אכן מכליל טוב לדאטה אמיתית ולומד היטב, והדאטה מחולקת באופן יונפורמי ומאוזן.

3.היתרונות של המסווג SVM:

-הוא שואף למצוא את הmargin האופטימלי בין המחלקות מה שעוזר בהכללה לנתונים חדשים,מה שעוזר בהפחתת overfitting .

-יש לו את היכולת להתמודד עם נתונים מסובכים ולא לינאריים הודות ליכולתו למפות נתונים לממדים גבוהים יותר.

-הוא קשוח ביחס ל outliers במטרה למצוא את ההפרדה הטובה ביותר בין מחלקות.

החסרונות של המסווג SVM:

- אימון SVM יכול להיות time consuming במיוחד עבור דאטה סט גדולים. בנוסף דרישות ה memory של SVM עולות עם מספר דגימות האימון מה שיכול להוות מגבלה עבור דאטה סטים גדולים.

היתרונות של המסווג KNN:

-אינו לו תקופת אימון.הוא מאחסן את דאטה סט האימון ולומד ממנו רק בזמן ביצוע תחזיות בזמן אמת.זה הופך אותו למהיר יותר ממסווגים אחרים שדורשים אימון.

-הוא מאוד פשוט ובפרט הלמידה הינה טריוויאלית, הקלסיפיקציה יש מימושים שבהם היא יעילה מאוד.

החסרונות של המסווג KNN:

- דורש הרבה מקום בזכרון ומשאבים חישוביים לאחסון ועיבוד כל דאטה האימון.ככל שהדאטה גדלה, חישוב המרחק והחיפוש אחר השכנים הקרובים ייקח לו זמן יותר.

- לא תמיד מקבלים את התוצאות האופטימליות בפרט אם יש פיצרים לא רלוונטיים בפיצר וקטור שלנו ,הייצוג שלנו לא כל כך מוצלח וחלק מהפיצרים לא כל כך אינפורמטיביים לגבי ההחלטה אז כן יכולים לזרוק אותנו לכיוון לא רלוונטי בזכות פיצרים אלה.

← אז לפי דעתנו SVM עדיף על KNN עבור משימת הסיווג שלנו, מכיוון ש:

-קיבלנו שה- accuracy של SVM גדול יותר מהaccuracy שלKNN.

SVM שואף למצוא את הmargin האופטימלי בין שתי המחלקות מה שעוזר בהכללה לדאטה חדשה ,מה שעוזר בהפחתת overfitting .

SVM טוב בטיפול כמות גדולה של פיצרים ,בעוד KNN יכול להיאבק עם הרבה פיצרים בגלל שהדברים מתפזרים מדי.

4.היתרונות והחסרונות ליצירת יחידות הסיווג:

היתרונות:

-יחידות סיווג מאפשרות סיווג וניתוח מדויקים יותר.

-יחידות סיווג עשויים להפחית רעשים שנובעים ממאפיינים פחות חשובים או לא רלוונטיים.

החסרונות:

-ניהול ועיבוד מספר רב של יחידות סיווג יכול להגביר את מורכבות המערכת ולגרום לדרישות חישוב ואחסון גבוהות יותר.

ההשלכות אם נגדיל ואם נקטין:

אם נקטין גודל chunk יותר מדי זה גורם ל underfitting

אם נגדיל גודל chunk יותר מדי זה גורם ל overfitting

שאלת הבונוס:

ניסינו כמה גדלים והגענו שהגודל הוב ביותר הינו 29

מכיוון שנותן הדיוק הגבוה ביותר ללא overfitting.