מבוא למערכות לומדות - 236756 - תרגיל בית מס' 5

ת"ז: 300816634, 300551140

Data Preparation

הרצנו את מניפולציות ה־$Data\ Preparation$ מגליון 2:

- $Outliers$ - ע"י $Z-Score$ ומי שחורג יותר מדי, מסומן כ־$Outlier$. החלטנו לא להפטר מרשומה, אלא להפוך את הערך החורג שלה ל־NaN שבשלב ה־$Imputation$ יוחלף ע"י ערך הגיוני יותר.
- $Imputation$ - בסט ה־$Train$ לפי $Mode$ (עבור ערכים נומינליים) או $Mean$ פר לייבל, ובסט ה־$Validate$ וה־$Test$ (וגם ה־$Test$ החדש) בלי להתייחס ללייבל.
- $Scaling$ - לפי $Standard\ Scaler$.
- $Feature\ Type$ - זיהינו פיצ'רים קטגוריאליים. עבור פיצ'רים קטגוריאליים שבהם הסדר משנה, וידאנו שהקידוד הוא לפי הסדר, והשתמשנו בקידוד. עבור פיצ'רים קטגוריאליים שאינם בינאריים, והסדר לא משנה בהם, המרנו ל־$One-Hot$, כדי לא להשרות סדר.
- $Feature\ Set$ - במשימה זו ניתן לנו סט הפיצ'רים ה"נכון", ולכן השתמשנו בו. כן נתון לשיקולינו האם להשתמש בכל הפיצ'רים שנוצרו מ־Most_Important_Issue (שהומר ל־$One-Hot$). דירגנו את הפיצ'רים ב־3 שיטות. לפי MI, לפי $Wrapper$ עם SVC, ולפי $Embedded$ עם $ExtraTreesClassifier$ - ועדת עצים רנדומיים שבוחרים לפי הממוצע וכחלק מאלגוריתם האימון, מדרגים את הפיצ'רים לפי חשיבות. על שלושת השיטות האלה לקחנו ממוצע, וכך יצרנו דירוג משלנו לפיצ'רים. ניסינו ללא הפיצ'רים שדורגו נמוך (רק מבין הפיצ'רים של Most_Important_Issue, כי הם היחידים שהיו נתונים לשיקולינו) - הרצנו מספר $Classifiers$, וראינו שהתוצאות נוטות לרדת, ולכן החלטנו להשאר עם כל סט הפיצ'רים שניתן לנו, כולל כל הפיצ'רים של Most_Important_Issue.

המשימות שניתנו לנו בסט ה־$Test$ החדש

1. לנבא את המפלגה הזוכה.
2. לנבא את התפלגות ההצבעות למפלגות.
3. לכל מצביע, לנבא לאיזו מפלגה יצביע.
4. למצוא קואליציה יציבה.

את משימה 3 נבצע ע"י הסט הישן - תחילה חלוקה שלו ל־$Train,\ Validate,\ Test$, וביצוע אופטימיזציית היפר־פרמטרים ב־CV על $Train$, אח"כ השוואת המודלים השונים עם $Validate$, ולבסוף כדי לקבל מושג כלשהו על יכולות המודל הסופי שלנו, שימוש ב־$Test$.
לבסוף, שימוש ב**כל** הסט הישן ($Train,\ Validate,\ Test$) בתור סט האימון, וניבוי על סט ה־$Test$ החדש.

בנוסף, את משימות 1 ו־2 ניתן לבצע ע"י שימוש בתוצאות שלנו ממשימה 3. ההצדקה לזה היא שסט ה־$Test$ החדש נבחר רנדומלית, ולכן גם הוא מדגם מייצג של התפלגות הבוחרים. לכן ניבוי שלנו פר בוחר בסט ה־$Test$ החדש יתן קירוב להתפלגות המפלגות (וכך נוכל גם להחליט מי המפלגה הצפויה לזכות).

את משימה 4 נבצע ע"י שימוש במודל קלאסטרינג (על סט ה־$Test$ החדש כמובן) אשר ימצא לנו מספר קלאסטרים שונים מאוד, על מנת

לקבל מושג כללי על מבנה הבוחרים. לאחר נשתמש בלייבלים שמצאנו במשימה 3, כדי להחליט לאיזו מפלגה הכי סביר שכל בוחר יצביע. כך נוכל ביחד עם הקלאסטרים והלייבלים, לחפש קואליציה יציבה. לאחר מכן, נגדיר באופן ידני המסתמך על תוצאות הקלאסטרינג, מספר רב של קואליציות אפשריות. נדרג את כל האפשרויות ע"י $Internal\ Evaluation$ של קלאסטרינג, אשר יבטא את דרישות הקואליציה ההומוגנית, כלומר קואליציה צפופה עם פריטים דומים, ועם זאת, מאוד שונה מהאופוזיציה. נבחר את הקואליציה אשר תקבל את הציון הטוב ביותר.

התשובות הסופיות בחלקים - $1-3.Final\ Results$ (עמוד 9) ו-$4.Coalition-Final\ Results$ (עמוד 14).

1-3. Per Voter Prediction, Party Distribution & Likely Winner

כפי שתיארנו באופן כללי קודם, עתה נרחיב. כמו בתרגיל 3, נבדוק מספר מודלים שונים, ונבחר את הטוב מביניהם.

אופטימיזצית ה־$Hyperparameters$

מטריקת $Weighted\ F1\ Score$:

לאחר ה־$Data\ Preparation$, עברנו לבדוק מודלים ע"י $\boldsymbol{Grid\ Search}$ עם $\boldsymbol{Cross-Validation}$.

מטריקת המטרה שלנו ב־$\boldsymbol{GridSearchCV}$ וגם בהמשך על סט ה־$Validation$ היא $\boldsymbol{Weighted\ F1\ Score}$ מ־2 סיבות:

- כי $Weighted$. במקרה של Multilabel Classification (המקרה שלנו - 11 מפלגות), אם התפלגות ה־$Labels$ לא אחידה (בדיוק המקרה שלנו - מפלגות - חלקן קטנות וחלקן גדולות) שימוש ב־$Accuracy$ יכול להוות בעיה, מכיוון שאחוז טעות גדול על מפלגות קטנות יתכן ולא יבוא לידי ביטוי. עדיין ניתן להשיג תוצאות $Accuracy$ גבוהות, כי בסה"כ סופרים את מספר הטעויות, ולכן יתכן מספר טעויות נמוך, אבל על לייבל קטן מסוים אחוז טעויות גבוה. כלומר $Accuracy$ יכול להטות אותנו לטובת לייבל נפוץ. עבור $Recall,\ Precision$ ו־$F1$ שהוא שילוב שלהם, ניתן לבחור במצב $average =' weighted'$ ולקבל שלכל לייבל משקל שווה, וכך נמנעים מההטיה הזאת.
- כי שילוב של $Precision$ ו־$Recall$ (הרי להתרכז באחד מהם היא החלטה ספציפית לבעיה, וכאן אין השלכות מיוחדות לטעות מסוימת, ולכן לקחנו $F1$ שהוא ממוצע הרמוני של שניהם).

המודלים:

החלטנו לבדוק מספר מודלים פשוטים - SVC, KNN ומספר מודלים יותר מורכבים Random Forest, Gradient Boosting, Multi Layer Preceptron.

כמו כן, ניסינו שימוש גם במטא־מודלים כמו $Bagging$ ו־$Voting$, אבל לא הצלחנו להגיע לתוצאות טובות יותר מ־GBC.

בהתחלה בדקנו הרבה מהפרמטרים, וראינו שחלקם נוטים פחות להשפיע, ובנוסף עם קריאה על הפרמטרים של כל אחד מהמודלים, וההבנה שחלקם יותר משמעותיים מאחרים, החלטנו לבדוק לכל מודל 3־2 פרמטרים ומספר ערכים יחסית קטן לכל פרמטר, על מנת שנוכל להציג את התוצאות בצורה סבירה, ועדיין לקבל תוצאות לא רעות.

בכל אחד מהפרמטרים השתדלנו להראות נקודה אופטימלית. לפני שמגיעים אליה, יש $Underfitting$, כלומר ככל שעולים לכיוון הנקודה האופטימלית, מקבלים שיפור בביצועים, ואחרי שעוברים אותה, ככל שמתרחקים ממנה מקבלים הרעה בביצועים שנובעת מ־$Overfitting$. לדוגמא עבור עצים והפרמטר $Max-Depth$. עץ מעומק קטן מדי יפגע ביכולות ההכללה. עץ מעומק גדול מדי יגרום ללמידה טובה מדי של סט האימון, ולכן נקבל $Overfitting$.

כמובן שהשתמשנו ב־$Grid\ Search$ - כלומר, נבדקו כל השילובים האפשריים של כל הפרמטרים.
לכל מודל נציג את התוצאות. מאופי הבעיה (מספר פרמטרים בין 3־2, ושמות הערכים הגדולים) קשה להציג את התוצאות בגרף. לכן נציג בטקסט, ונשתדל להסביר את התוצאות, ואם ניתן נציג בגרף.

המודלים מוצגים מהדף הבא.

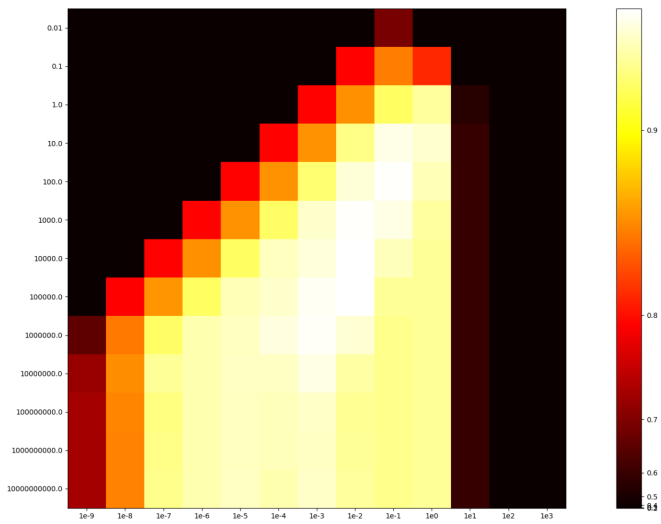
- SVC:

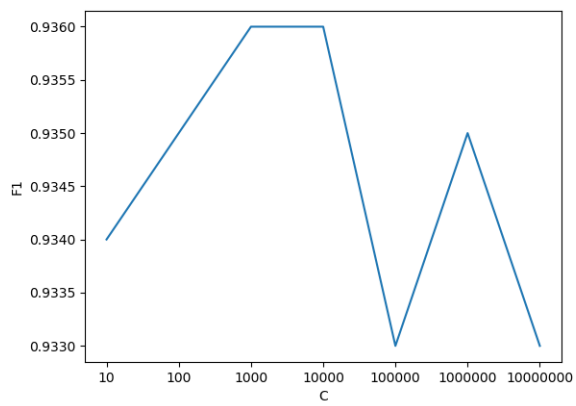
בדקנו $Kernel$ לינארי או RBF. קיבלנו באופן גורף ש־RBF טוב יותר. התוצאות עבור קרנל לינארי כצפוי, עד נקודה מסוימת מקבלים שיפור, והחל ממנה ירידה.

בדקנו $Gamma$ בין $1e-9$ לבין $1e2$ (בקפיצות כפול 10 בכל פעם, כלומר 12 ערכים).

בדקנו C בין $1e-2$ לבין $1e10$ (שוב בקפיצות כפול 10 בכל פעם, כלומר 13 ערכים).

תוצאות הניסוי (בצד ימין עבור קרנל לינארי, בצד שמאל עבור קרנל RBF):



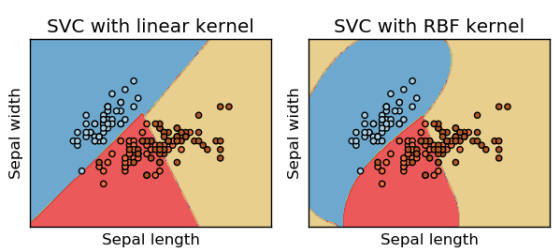


מבחינת הצבעים ב־RBF, השתמשנו ב־$PowerNorm$ כדי שיהיה קל יותר להבדיל בהבדלים המינוריים לקראת הערכים האופטימליים, שנמצאים על האלכסון (נסביר בהמשך).

המודל בעל הפרמטרים האופטימליים מבין אלה שבדקנו היה בעל $kernel =' rbf'$, $C = 100000$ ו־$gamma = 0.01$.

הסבר:

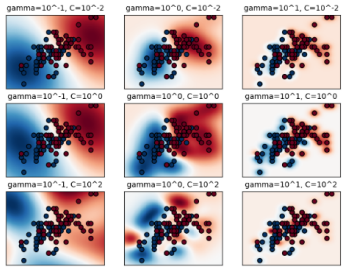
- $Kernel$ - לינארי או RBF:

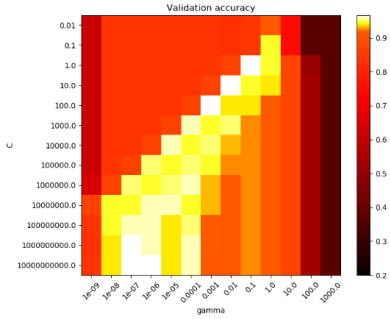


קרנל לינארי מוגבל יותר. RBF מאפשר ליצור מושגים מורכבים יותר.

- C ו־$Gamma$:

סדרת ניסוים שהתבצעה על ה־$Iris\ Dataset$ מה־$Documentation$ של $sklearn$, דומה מאוד לתוצאות שקיבלנו (למרות שכאן בודקים $Accuracy$):

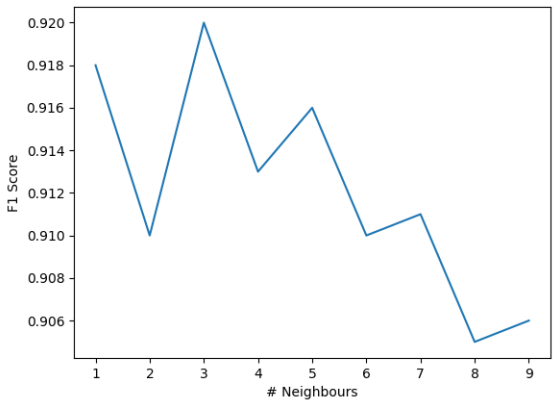




* C: מספר ה־$Support\ Vectors$ שנבחרים בזמן האימון. מייצג $Trade\ off$ בין טעות על סט האימון לבין פשטות המודל. ככל ש־C גדול יותר, רמת הדיוק על סט האימון עולה (עד גבול מסוים שגורם ל־$Overfitting$).
* $Gamma$: גודל איזור ההשפעה של כל $Support\ Vector$. $Gamma$ קטנה מאוד אומרת שאיזור ההשפעה של כל וקטור יהיה כל סט האימון, וככל שנגדיל איזור ההשפעה יקטן, וכך יאפשר יצירת צורת מורכבות יותר, כלומר נוכל ללמוד מושגים מורכבים יותר, עד גבול מסוים של למידה טובה מדי של סט האימון, כלומר $Overfitting$.

גם בתוצאות שלנו, וגם בתוצאות שהצגנו עכשיו (נשים לב שציר ה־y מסודר בסדר יורד), ניתן לראות את השפעות ה־$Overfitting$ של שני הפרמטרים.
לכן אם מגדילים אחד, מקטינים את האחר. התוצאות האופטימליות על האלכסון - המקום בו אכן מתבצע האיזון הזה.

- KNN:



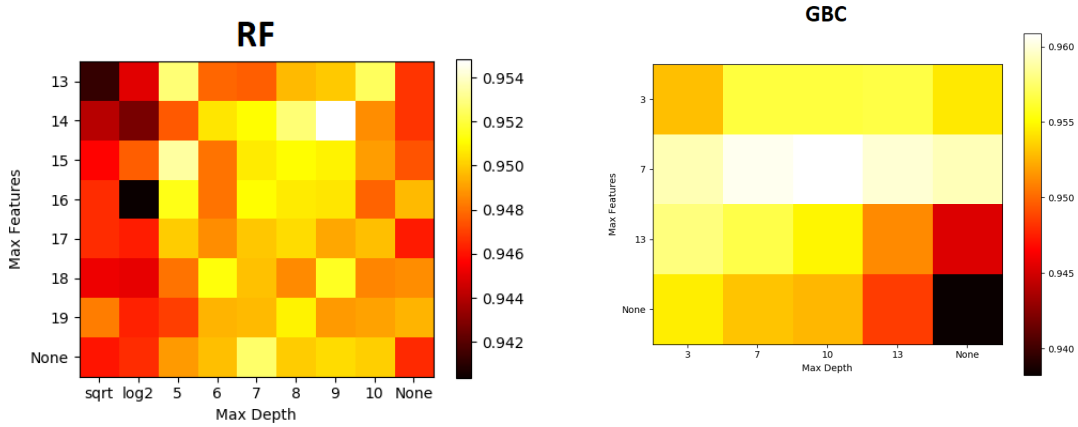
המודל האופטימלי שקיבלנו בעל $k = 3$.

- k: עבור דגימה שנרצה לסווג, k הוא מספר השכנים הקרובים ביותר שניקח, ונסתכל על ההחלטה שלהם.

ככל ש־k קטן יותר, אנו לומדים את סט האימון טוב יותר (עבור $k = 1$ דיוק של 100% על האימון), מה שפוגע ביכולות ההכללה, וגורם ל־$Overfitting$. הגדלת K תמנע זאת (תוריד את הדיוק על סט האימון בתמורה לשיפור יכולות הכללה), אבל החל משלב מסוים תתחיל להקטין את יכולת ההכללה (כאשר מצב הקיצון הוא k שווה למספר הדגימות בסט האימון - כלומר כל הסיווגים יהיו אותו הדבר). בחירת k אי זוגי מונעת בעיות של תיקו. לפי התוצאות נראה ש־k אי זוגי אכן נותן תוצאות טובות יותר.

- $Tree\ Based$ - RF & GBC:
  - $n\_estimators$ כמה שיותר יותר דיוק, על חשבון ביצועים (מהירות) - ניסינו 500 ו־1000, אין הבדל גדול מספיק כדי להצדיק את זמן הניסוים. נשארנו עם הערך הדיפולטי.
  - max_depth ככל שהעץ עמוק יותר - לומד את סט האימון טוב יותר. עמוק מדי - $Overfitting$. לא עמוק מספיק - $Underfitting$.
  - $max\_features$ - עבור RF ו־GBC - מספר הפיצ'רים שנבחרים באופן רנדומלי בכל פיצול.

RF ו־GBC:



ב־RF המודל האופטימלי שקיבלנו בעל $max\_depth = 9$, ו־$max\_features = 14$.
ב־GBC המודל האופטימלי שקיבלנו בעל $max\_depth = 7$, ו־$max\_features = 10$.
כצפוי, הערכים האופטימליים באמצע ובאלכסון, וניתן לראות את העליה עד ה־$Overfitting$ (כאשר מגדילים כל פרמטר בנפרד) ולאחר מכן ירידה, כלומר ככל שמתרחקים מהאמצע, מקבלים ירידה בביצועים.

- $Multi-layer\ Perceptron$:

```
MLP
Best parameters set found on development set:
{'alpha': 0.00015, 'hidden_layer_sizes': (500, 500)}

Grid scores on development set:
0.934 (+/-0.012) for {'alpha': 0.0001, 'hidden_layer_sizes': (15,)}
0.940 (+/-0.008) for {'alpha': 0.0001, 'hidden_layer_sizes': (15, 15)}
0.950 (+/-0.008) for {'alpha': 0.0001, 'hidden_layer_sizes': (100, 100)}
0.949 (+/-0.010) for {'alpha': 0.0001, 'hidden_layer_sizes': (500, 500)}
0.937 (+/-0.010) for {'alpha': 0.00015, 'hidden_layer_sizes': (15,)}
0.939 (+/-0.009) for {'alpha': 0.00015, 'hidden_layer_sizes': (15, 15)}
0.947 (+/-0.010) for {'alpha': 0.00015, 'hidden_layer_sizes': (100, 100)}
0.954 (+/-0.006) for {'alpha': 0.00015, 'hidden_layer_sizes': (500, 500)}
```

- **פקטור רגולריזציה:**

אנו לומדים את סט האימון במטרה לייצר מודל בעל יכולות הכללה טובות. כלומר ביצועים (במקרה זה דיוק) על מידע חדש (שלא ראינו בסט האימון).
הכללה חשובה מכיוון שסט האימון שלנו הוא בסה"כ סט סופי של דגימות - הוא אינו מכיל את כל המידע, אלא רק מהווה חלון שדרכו אנו מסתכלים על ההתפלגות האמיתית. בנוסף יתכן ומכיל רעש.
2 סיבות אלה מסבירות למה אנחנו רוצים להמנע מ־$Overfitting$, כלומר למידת סט האימון שלנו כל כך טוב, שאנחנו נותנים משקל גבוה לדגימות שראינו וגם לומדים את הרעש, וכך פוגעים בביצועים על דגימות חדשות (כלומר פוגעים ביכולות ההכללה).
כמובן ש־$Underfitting$ (לא לומדים מספיק את סט האימון) גם פוגעת בנו, מכיוון שסט האימון מכיל המון מידע על ההתפלגות האמיתית.
אם לא נלמד את סט האימון מספיק טוב, לא נלמד מספיק על ההתפלגות האמיתית, ולכן לא נוכל לקוות לביצועים טובים על דגימות חדשות.

המטרה שלנו היא כמובן למצוא נקודה אופטימלית המפשרת בין שני מצבי הקיצון האלה.

בתהליך האימון, המטרה שלנו היא למזער את פונקצית ה־$loss$.
כאשר ערכי המשקלים גדלים, אנו באופן פוטנציאלי, נותנים משקל גדול לרעש, מה שיכול לפגוע ביכולות ההכללה שלנו.
לכן אנו מוסיפים פקטור רגולריזציה, בכדי באופן מלאכותי להגדיר מחיר גבוה יותר עבור משקולות גבוהים.
כי עתה עוצמת המשקולות היא חלק מפונקצית המטרה שאותה אנו ממזערים.
ולכן נכניס משקלים גדולים יותר, רק כאשר השגיאה יורדת בהרבה.

לא אכפת לנו קצת להגדיל את השגיאה (על סט האימון), אם אפשר לקבל משקולות קטנים יותר.
מכיוון שאנו מבצעים אופטימיזציה, המשקולות שיקטנו הם גם אלה שהיו מוסיפים לנו רעש, לו היו גדולים יותר.

וככה אנחנו מגדילים טיפה את השגיאה על סט האימון, אבל מרוויחים יכולות הכללה יותר טובות.

- **מבנה הארכיטקטורה (מימדים של השכבות הנסתרות):**

יתרונות וחסרונות עבור עומק, רוחב וכמות הנוירונים גבוהים:

  - **חסרונות:**
    - **זמן אימון:**
      ככל שיש יותר נוירונים, יש יותר חישובים. זמן אימון וגם זמן שלוקח לתייג דגימות, שניהם עולים.
    - **פקטור רגולריזציה:**
      ככל שיש יותר נוירונים, המשקל שלהם בפונקצית ה־$Loss$ גדל, עד שבשלב מסוים נהיה דומיננטי יותר מפונקצית ה־$Loss$ ללא הרגולריזציה.
      לכן דרוש פקטור רגולריזציה חזק יותר (כלומר קטן יותר) כדי למנוע $Overfitting$.
  - **יתרונות:**
    ככל שהרשת יותר רחבה, עמוקה, ובעלת כמות נוירונים גדולה יותר, הביצועים שלה טובים יותר.
    - **מרחב היפותזות גדול יותר:**
      אפשר לחשוב על רשת עם פחות נוירונים, או רשת פחות עמוקה או פחות רחבה, כמקרה פרטי של העמוקה\רחבה או בעלת יותר

נוירונים ממנה. ע"י קביעת 0 עבור חלק מהפרמטרים נוכל לקבל רשת המתנהגת באופן דומה.
כאשר אנחנו מגבילים את הרשת לכמות נוירונים קטנה, אנחנו יוצרים $Bias$ מסוים, כי במרחב ההיפתוזות הגדול יותר, המכיל את הקטן ממנו, יתכן ויש היפותזות טובות יותר, שלא נמצאות בקטן.

השוואת המודלים לאחר אופטימיזציית ה־$Hyperparameters$:

לאחר שביצענו אופטימיזצית פרמטרים לכל אחד מהמודלים, אנו מקבלים את רשימת המודלים כאשר כל מודל בעל פרמטרים אופטימליים (מבין אלה שבדקנו), ומאומן על **כל** סט ה־$Train$.
לכל אחד מ־6 המודלים שלנו, אנו מריצים $Predict$ על סט ה־$Validation$. לאחר מכן ממיינים את המודלים לפי $F1\ Score$ (שוב זו המטריקה שלנו, מאותן הסיבות מקודם) שקיבלנו על סט ה־$Validation$, ובוחרים את המקסימלי.

התוצאות שקיבלנו:

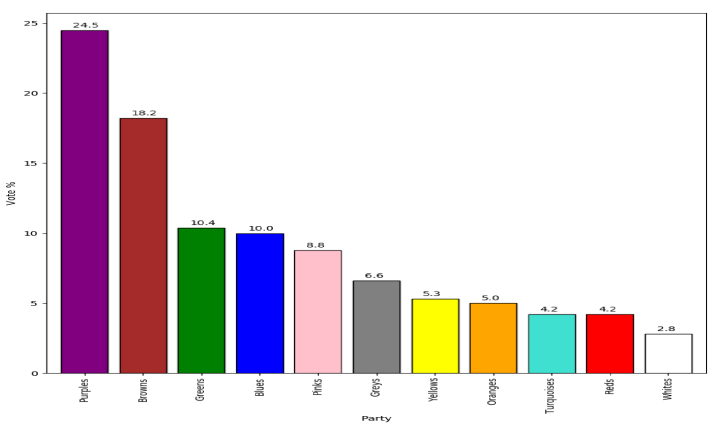
```
Models Evaluated F1 Score:
      Model Name            F1 Score
0            GBC  0.9669128064615438
1  RANDOM_FOREST  0.9558482944924336
2            MLP  0.9555412309840482
3            SVC  0.9520369037194467
4  DECISION_TREE  0.9362408051017167
5            KNN  0.9300284186768094

Best Model Is:
GBC 0.9669128064615438
```

המודל הטוב ביותר שקיבלנו הוא GBC.

הערכת המודל האופטימלי:

נריץ את המודל האופטימלי על סט ה־$Test$ (הישן).



Confusion Matrix

Actual \ Predicted	Blues	Browns	Greens	Grevs	Oranges	Pinks	Purples	Reds	Turquoises	Whites	Yellows
Blues	96	0	0	0	0	0	0	0	0	0	5
Browns	0	171	0	0	0	5	1	0	0	2	0
Greens	0	0	104	0	0	3	3	0	0	0	0
Grevs	0	0	0	65	6	0	0	2	0	0	0
Oranges	0	0	0	1	43	0	0	0	0	0	0
Pinks	0	7	0	0	0	79	4	0	0	1	0
Purples	0	1	0	0	0	0	235	0	0	0	0
Reds	0	0	0	0	1	0	0	40	0	0	0
Turquoises	1	0	0	0	0	0	1	0	42	0	1
Whites	0	3	0	0	0	1	1	0	0	25	0
Yellows	3	0	0	0	0	0	0	0	0	0	47

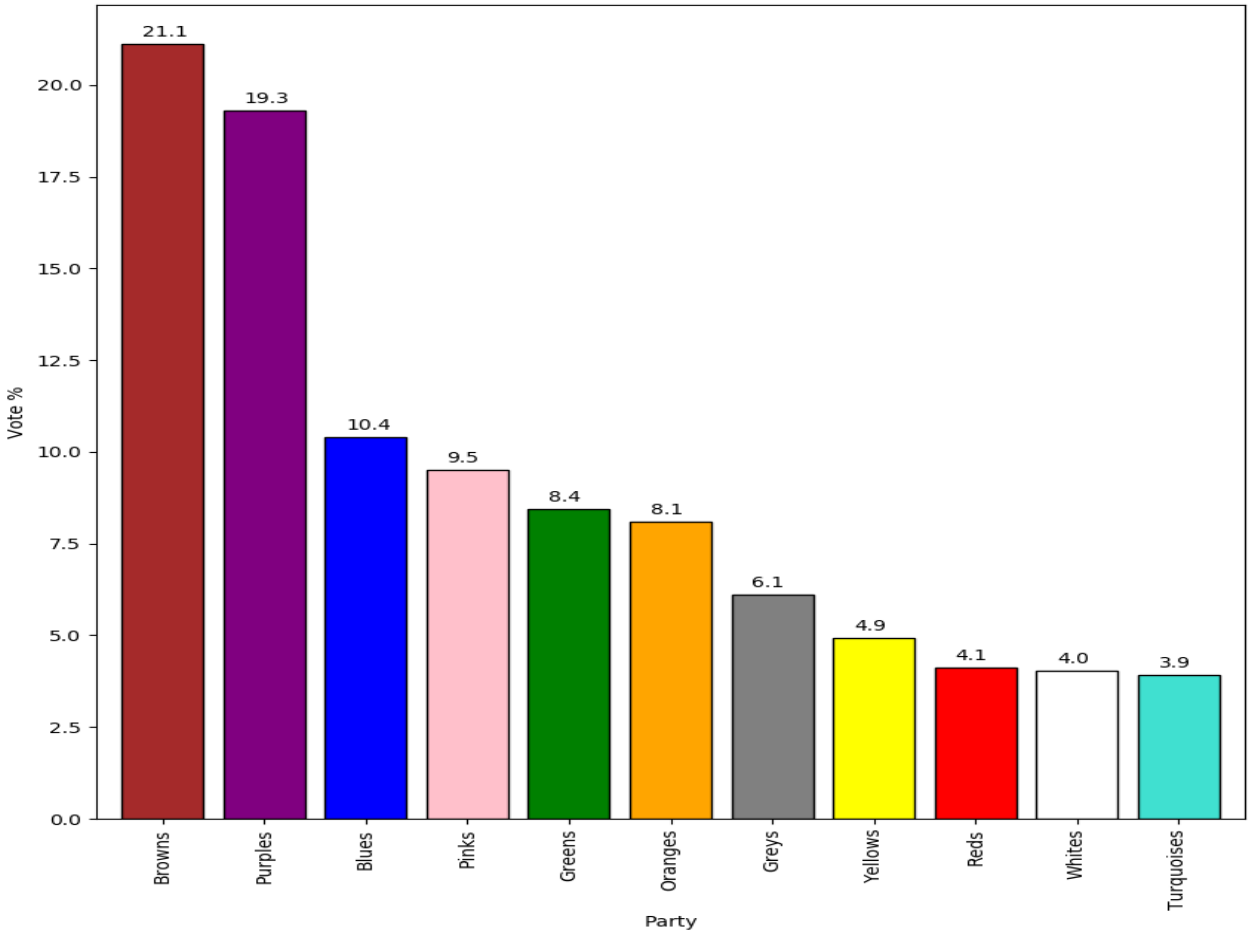
ה־$Test\ Error$ (כלומר $1 - Accuracy$) שלנו הוא 0.053, וה־$Accuracy$ היא 0.947.
כפי שניתן לראות, בזכות כך שהשתמשנו ב־$Weighted\ F1$ (מההסברים הקודמים שלנו), גם על הקלאסים שמופיעים פחות, יש אחוז דיוק לא רע, ולא קיבלנו הטיה גדולה מדי עבור קלאס גדול.
התוצאות די דומות לתוצאות הנכונות שפורסמו לתרגיל בית 3.

1-3. Final Results

עכשיו **נאמן** את מודל ה־GBC שלנו (עם הפרמטרים האופטימליים שמצאנו) על כל הסט הישן ($Train,\ Validate,\ Test$ ביחד).
התוצאות בקובץ $results.csv$ (עמודת $IdentityCard\_Num$ עבור ת"ז של המצביע, ועמודת $PredictVote$ בשביל ההצבעה הצפויה שלו).

נריץ את הקלאסיפייר המאומן הזה, על סט ה־$Test$ החדש.

מההסברים מההתחלה, נשתמש בתוצאות אלה על מנת לענות על משימות 1־3.



1. ניתן לראות שהמפלגה אשר תנצח היא מפלגת ה־**Browns**.
2. התפלגות ההצבעות לפי הגרף.
3. הצבעה לכל מצביע בסט החדש, נמצאת בקובץ $results.csv$.

4. Coalition

Clustering Model Hyperparameter Optimization

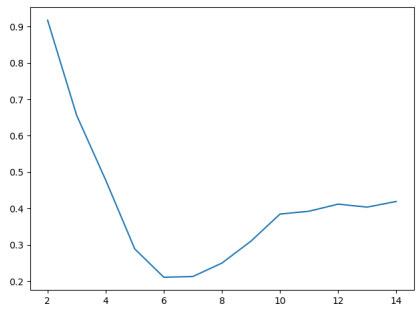
נתחיל מהסבר על המטריקות ששקלנו להשתמש בהן בשביל מציאת הקואליציה. לחלק מהמטריקות שכן עניינו אותנו, הרצנו $\boldsymbol{KMeans}$ ב־CV, עם ערכי $k \in [2, 100]_{\mathbb{N}}$. נציג את התוצאות כאן.

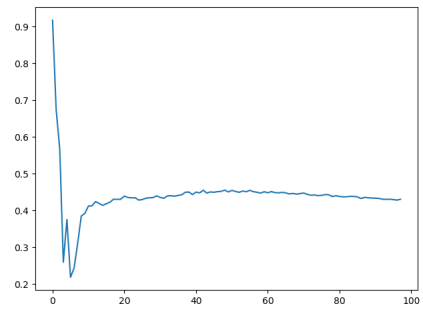
- $External\ Evaluation$:
  - $Purity$:

    פחות משקף את דרישות התרגיל. ה־"$Purity$" היא ביחס ללייבל (המפלגה) הנפוץ בקלאסטר. אין כאן שום ביטוי לדרישה שהמצביעים לקואליציה יהיו דומים מאוד, ושונים מאוד מהאופוזיציה.
  - $Completeness$:

    פחות מבטא את דרישות הדמיון בין המצביעים, אך כן יש הגיון מסוים בשימוש במטריקה הזאת, כי אולי כן נרצה לשאוף שמפלגה תוכל בשלמותה בקלאסטר (שאמור לייצג לנו חלק מהקואליציה). מצד שני, המטריקה הזאת מאוד נוקשה (0 או 1).

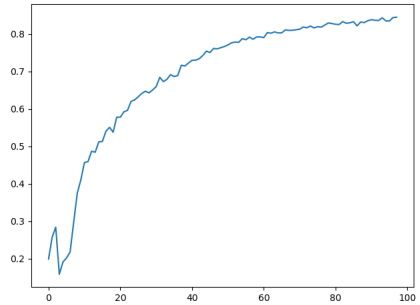
    נציג את התוצאות (מצד ימין עבור $k \in [2, 100]_{\mathbb{N}}$, ומצד שמאל עשינו זום עבור ערכי $k \in [2, 14]_{\mathbb{N}}$).





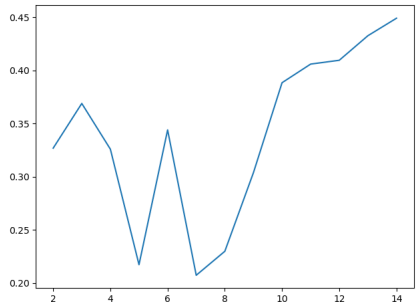
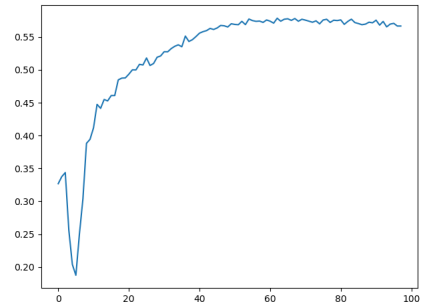
  - $Homogeneity$:

    מטריקה לא טובה למטרות התרגיל. לא מבטא את הדרישה שהמצביעים יהיו דומים. ככל שיהיו יותר קלאסטרים, נקבל קלאסטרים הומוגניים (מכילים לייבל אחד). ואכן התוצאות מראות עליה, ככל שה־k גדל:



    כמו כן, המטריקה מאוד נוקשה (0 או 1). אך עדיין יתכן שנרצה שבכל קלאסטר יהיו כמה שפחות לייבלים שונים. אפשר לשלב את $Completeness$ עם $Homogeneity$.
  - $V - Measure$:

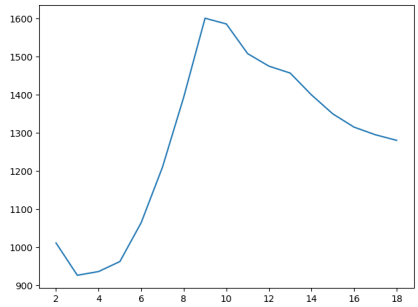
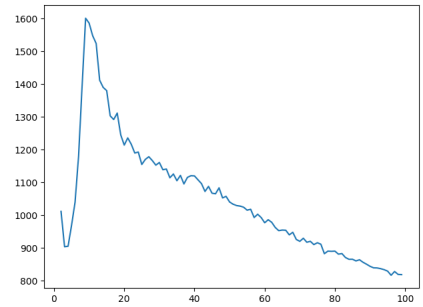
    ממוצע הרמוני בין $Completeness$ ו־$Homogeneity$.

מעבר ל־14 ה־$Completeness$ לא משתנה, ו־$Homogeneity$ רק גדל, אז כמובן שהוא משתלט על ה־$V-Measure$. ערכים מעל 14 פחות יעניינו אותנו (לפחות מבחינת המדדים האלה, $Completeness,\ Homogeneity,\ V-Measure$).
לפי מדד זה, נרצה לבדוק לעומק את הערכים $k \in \{2,3,4,6,9,10,11,12\}$ (טובים יחסית).

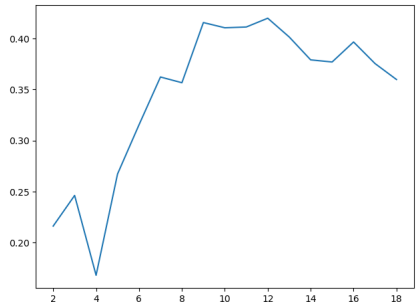
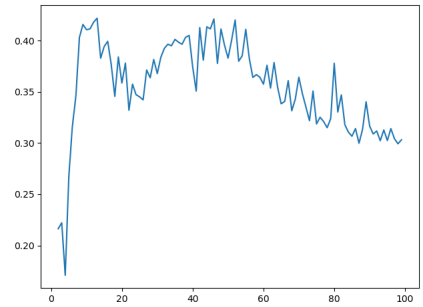
- $Internal\ Evaluation$:
  - $Calinski\ Harabaz$:

התוצאה גבוהה יותר, ככל שהקלאסטרים צפופים ומופרדים יותר טוב. זה בדיוק מה שאנחנו מחפשים.

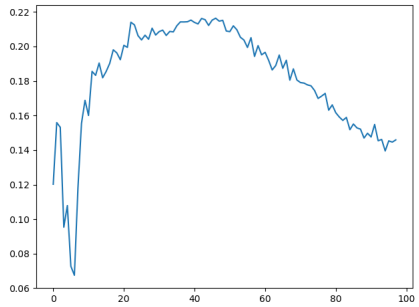
  - $Silhouette$:

בדומה למדד הקודם, התוצאה גבוהה יותר ככל שהקלאסטרים צפופים ומופרדים יותר טוב, ואנו מקבלים תוצאות דומות למדד הקודם.

לפי מדדים אלה, נרצה לבדוק את ערכים $k \in [6,12]_{\mathbb{N}}$. מכיוון שלפי המטריקות הקודמות שעניינו אותנו רצינו לבדוק את ערכים $k \in \{2,3,4,6,9,10,11,12\}$, סה"כ נבדוק לעומק את הערכים (החיתוך) ־ $k \in \{6,9,10,11,12\}$.

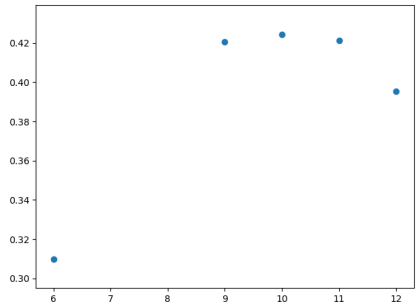
- $Relative/Stability\ Evaluation$:
  - $Adjusted\ Rand\ Index$:

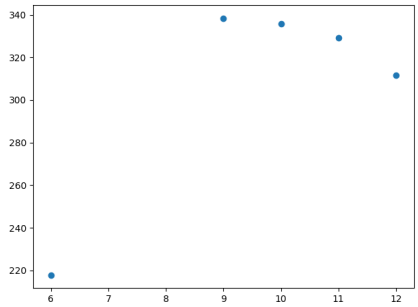


מדד זה פחות עניין אותנו, כי הוא פחות מבטא את דרישות הקואליציה היציבה, והערכים הטובים בו הם ערכים לא ריאליים במדדים האחרים שכן בחרנו בסוף.

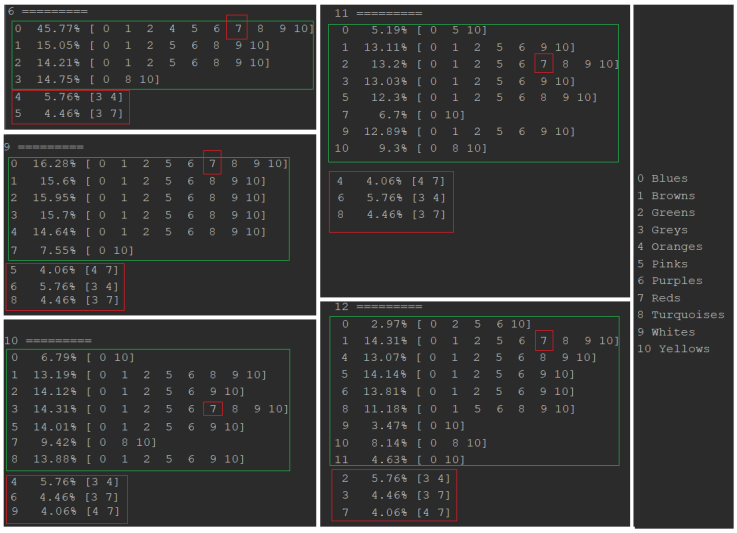
נרצה לבדוק לעומק את ערכי $k \in \{6, 9, 10, 11, 12\}$. ערכים אלה נבחרו לפי $V - Measure$, שהוא שילוב בין $Completeness$ ו־$Homogeneity$. כלומר, ערכי ה־k שלנו מהווים פשרה סבירה בין שאיפה שכל קלאסטר יכיל בשלמות את המפלגה, לבין השאיפה שכל קלאסטר יכיל כמה שפחות מפלגות. בנוסף, בחרנו ערכים אלה לפי מדדי $Internal\ Evaluation$ יחסית גבוהים, כלומר, הקלאסטרים שנוצרים כאן הם יחסית צפופים ומופרדים טוב.

נריץ עם $k \in \{6, 9, 10, 11, 12\}$, על ה־$Validate$, ונבחר את הערכים הטובים ביותר (מימין $Calinski\ Harabaz$, משמאל $Silhouette$).





נראה ש־ $9 - 12$ נותנים את התוצאות הטובות ביותר. לכן נתרכז ב־12־9, אבל גם נתבונן ב־6 כדי לראות איך מתבטא הציון הנמוך יותר. נריץ $K - Means$ עם ערכי $k \in \{6, 9, 10, 11, 12\}$ על **כל** ה־$data$ (כלומר חיבור של $Train, Validate, Test$), ונציע קואליציה לפיו.



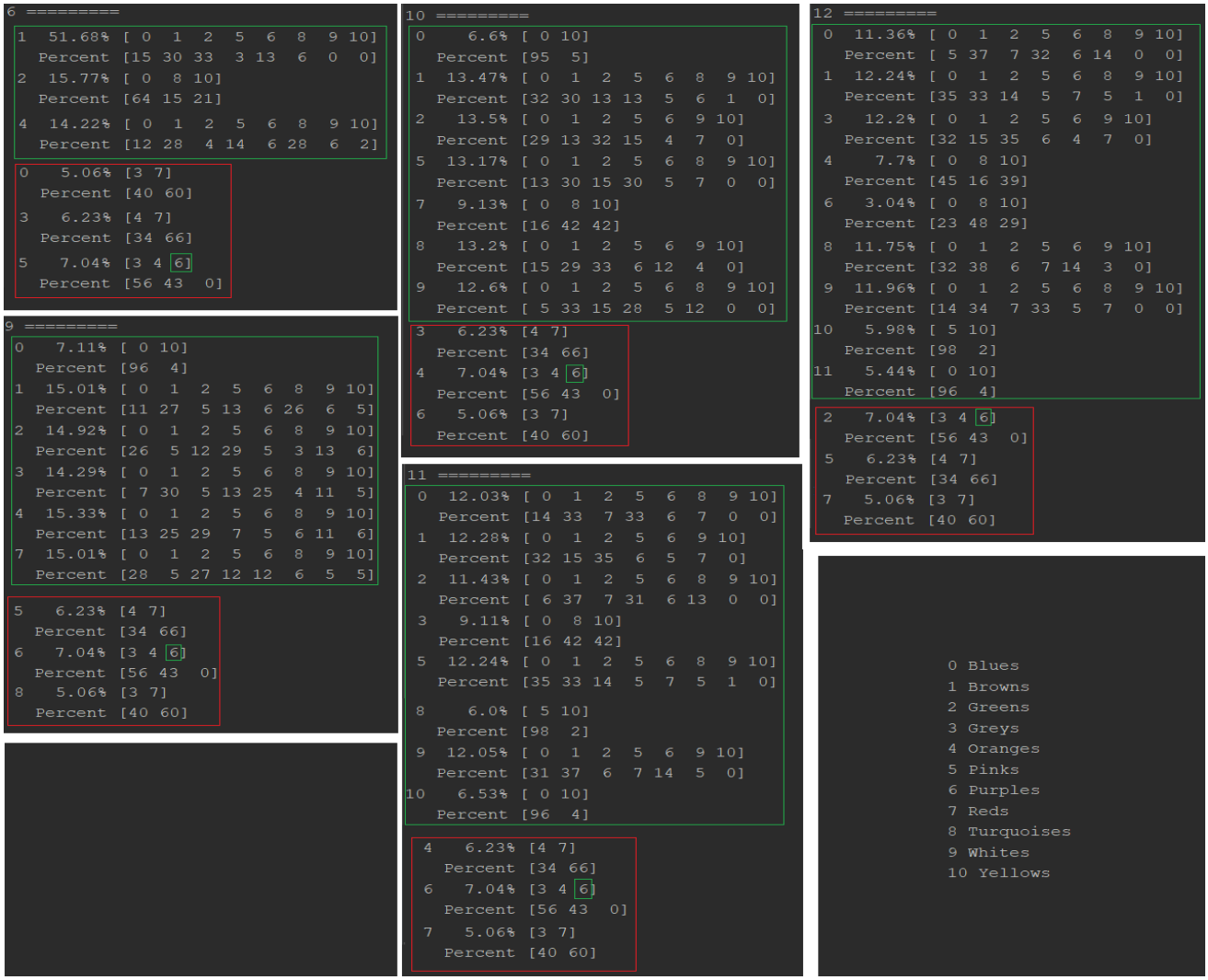
הסבר קטן על הצגת הנתונים (המדהימה) - כל k נמצא במרובע משלו, בראש רשום את מספר ה־k. לכל k יש 3 עמודות, מספר הקלאסטר, אחוז המצביעים שנמצא בקלאסטר, ובעמודה האחרונה - ערכי ההצבעות השונות בקלאסטר.

ניתן לראות מוטיב חוזר. המפלגות $3,4$ ($Greys,\ Oranges$) נוטות להכיל מצביעים דומים מאוד, וכן להיות מופרדות משאר המפלגות. כמו כן, מפלגה 7 ($Reds$) גם מקיימת זאת איתן, רק שחלק ממצביעיה דומים לשאר המפלגות.
זאת בעיה מבחינתינו, אך כמובן שלא ניתן לצפות (גם בחיים האמיתיים) שתהיה חלוקה **מושלמת**, ודמיון כלשהו בין מצביעים הוא צפוי. כמובן שאנו שואפים כמה שיותר להפריד לאופוזיציה "צפופה" ו־"שונה מאוד" מהקואליציה. בכל מקרה, אחוז מצביעי 7 מהקלאסטר הבעייתי הוא מאוד קטן (בערך 5% מהקלאסטר), ולכן נצפה שלא תהיה השפעה רבה לכך שלא כללנו את 7 ($Reds$) בקואליציה (במיוחד כאשר יש דמיון כה רב בין 3,4,7).

Coalition - New Test Set

כל מה שתיארנו בחלק הקודם, בוצע על הסט הישן (המתויג) כדי להשוות ערכי k שונים ולנתח את התוצאות.

עתה, כפי שהסברנו בהתחלה, נריץ את המודל שלנו עם ה־kים שהתעניינו בהם, ונראה האם נוכל למצוא מגמה דומה למה שמצאנו בסט הישן. כדי להחליט על אילו מפלגות מדובר, נשתמש בתיוגים שלנו ממשימות 1־3.
התוצאות, בדומה לקודם:



הצגנו את הנתונים בדומה לחלק הקודם, רק הפעם הוספנו לכל קלאסטר שורה אשר מראה לכל לייבל בקלאסטר, איזה אחוז הלייבל מהווה מהקלאסטר. לדוגמא, ניתן לראות שמפלגה 6 (ה־**Purples**) התגנבה לנו לאופוזיציה, אבל בפועל היא מהווה מכל קלאסטר אופוזיציה כזה, מספר מאוד קטן, שמעוגל ל־0.
ניתן לראות שהתוצאות דומות מאוד למה שקיבלנו על הטסט הישן.

מחלק זה נסיק שעבור מפלגות $3,4,7$ (**Greys**, **Oranges**, **Reds**) המצביעים מאוד דומים, ושונים מאוד מכל שאר המפלגות. מכיוון שמפלגות

אלה מהוות אחוז קטן יחסית (בערך 18% ביחד), לא נוכל ליצור מהן קואליציה (מעל 51% מההצבעות), כי נצטרך להוסיף להם מפלגות אחרות, שכפי שניתן לראות, מצביעיהן מאוד שונים מהם.

לכן ניקח את כל שאר המפלגות (0־01, חוץ מ־7,4,3), ונבדוק את כל האפשרויות לקואליציה המורכבת ממפלגות אלה. כלומר ננסה את כל הקומבינציות. מבין אלה אשר מהוות יותר מ־51% מהמצביעים, נדרג אותם ע"י *Internal Evaluation*. נשתמש ב־*Calinski Harabaz*, כי היא מהווה מדד אשר מעודד דמיון בתוך קלאסטרים, והפרדה ביניהם. לכל קומבינציה, נתייחס בתור קלאסטר קואליציה וקלאסטר אופוזיציה.

נציג את 3 התוצאות הטובות ביותר שקיבלנו (מימין ציון *Calinski Harabaz*):

```
Best coalitions:
((1, 2, 5, 6, 9), 1176.9134528409932)
((0, 1, 2, 5, 6, 8, 9, 10), 1127.8886275275393)
((1, 2, 5, 6, 9, 10), 1096.9348596720301)
```

הקואליציה השניה, לא רחוקה מהראשונה, והיא **כל** המפלגות, פרט למפלגות $3, 4, 7$ (**Greys**, **Oranges**, **Reds**).
עם זאת, נבחר בקואליציה הראשונה שקיבלנו. הקואליציה שהצענו תהיה יציבה יותר מקואליציות אחרות, בגלל הדמיון החזק בין מצביעיה שנובע ממדדי ה־*Internal Evaluation* הגבוהים יחסית. כל קלאסטר שלנו צפוף, ומופרד מקלאסטרים אחרים.

4.Coalition - Final Results

התוצאות הסופיות הן מפלגות:

- **Browns** (**21.1%**)
- **Purples** (**19.3%**)
- **Pinks** (**9.5%**)
- **Greens** (**8.4%**)
- **Whites** (**4%**)

סה"כ 62.3% מהמצביעים.
כמובן שגם ניתן לבחור את כל המפלגות פרט למפלגות $3, 4, 7$ (**Greys**, **Oranges**, **Reds**), ולקבל קואליציה גדולה יותר, אך במחיר קטן של דמיון המצביעים בקואליציה, ושוני בין הקואליציה ולאופוזיציה.