

## **פרויקט למדת מכונה**

**נושא הפרויקט:** סיווג כל נגינה

**מגישים:**

- יותם דפנה – 205890908
- מתי שטרן – 311603641

GitHub -

[Yotam124/Machine-Learning-Project: Classification of musical instruments \(github.com\)](https://github.com/Yotam124/Machine-Learning-Project)

Drive for data-sets -

<https://drive.google.com/drive/folders/1RGBI1blAheFubafPZ3Im2DFtKNozyo6k?usp=sharing>

## **תיאור המאגר:**

המאגר שלנו כולל בתוכו קטעי שמע (סוג wav) של ניגון בכליים שונים. כאשר השם של הקובץ הוא הכליל המתנגן.

לקחנו שני מאגרים משנה מקומות שונים כאשר האחד (בשם data-set) כולל בתוכו קטעי מוזיקה בהם מתנגנים כל יחיד בלבד, השני (בשם IRMAS-TrainingData) כולל בתוכו קטעי מוזיקה בהם מתנגנים כמה כלים שונים כאשר השם של הקובץ מייצג את הכלים הללו לפי סדר הדומיננטיות שלהם (משמעות לימים) ולבסוף מצין גם דיאגרה המוזיקה.

מעבר לפיזול data ל-train ו-test, הבנו עוד מאגר (בשם IRMAS-TestingData) המכיל מאגר של קטעי שירים, אשר לכל קובץ wav. קיימים בנוסף קובץ txt מקביל המכיל את הכלים המתנגנים שם. ביצענו שימוש במאגר זה על מנת לנסות לחזות איזה כלים מתנגנים בשיר.

את טיענת הקבצים ביצענו דרך parser שכתבנו (בהתאם למאגר שרוצים לטען)

## **בעיות/אתגרים ופתרונות:**

### **• בעיה 1: בחירת מאגר**

במהלך בחירת המאגר היינו המון הבעיות של מגיר אחד או יותר אינחנו רוצים. האם לבחור מאגר המכיל קטעי שמע של כלים יחידים, או שילוב של כלים, האם קטעי שמע קצרים של שניות בודדות מספיקות או האם ציריך קטעי שמע ארוכים יותר, האם בחירה של אחד מהם עלולה להקשות علينا בהמשך כאשר נרצה לבצע סיווגים מורכבים יותר וכדומה.

### **פתרונות:**

לבסוף בחרנו להשתמש ב-3 מאגרים שונים ולבצע בהם שילובים על מנת לאפשר סיווג בנסיבות שונות, ובפרט לענות לעצמנו על השאלות שהאלנו בהתחלה.

### **• בעיה 2: המרת audio ל- vector מספרי.**

על מנת למדוד ולסwoג את המידע באמצעות האלגוריתמים השונים, היינו צריכים להמיר את קבצי השמע לווקטור מספרי בעל משמעויות. על מנת לעשות זאת היינו צריכים למדוד ולהבין איך שמע עובד, איך הוא מיוצג לפי דיאגרמות שונות ואילו פרמטרים קיימים בו.

### **פתרונות:**

לאחר שקרהנו והבנו איך קבצי השמע עובדים, ניגשנו לספריה בשם librosa המאפשרת לטען קובץ אודיו ולחצץ ממנו מידע שבאמצעותו ניתן להרכיב את הווקטור. בפרויקט בנוינו 2 פונקציות ליצירת ווקטורים אשר אחת ממירה כל קובץ אודיו לווקטור בגודל 13 כאשר הוא כולל את המידע שראינו כרלוונטי ביותר (משמעות הכי טוב על סיווג), והשנייה ממירה

לוקטור בגודל 20 (מכיל את ה13 ועוד 7 פרמטרים נוספים של המידע על אופי הקובץ שמע).  
הסיבה ליצירת 2 פונקציות שונות היא שרצינו לבדוק האם ניתן לשפר את אחוזי הצלחה  
באמצעות הוספה של מידע נוסף לוקטור.  
\*) ניתן לראות תוצאות של הבדיקה בסוף (\*)

• **בעיה 3: אופן תיוג הדגימות**  
כאשר ניגשנו לעבד על מאגר של כלים משלבים, נתקלו בעוויות סיוג (אחוזי הצלחה נמוכים,  
וכמות גדולה של מחלקות שונות). לכן התחלנו לנסות ולהבין מהי הדרך הטובה ביותר לתיוג  
דגם חז. (כזכור, שם הקובץ של כל דגימה בניי לפי סדר הדומיננטיות של הכלים ולבסוף מצוין הז'אנר)

**פתרונות:**  
ניסינו לסוג במספר שיטות שונות, והשוונו בין התוצאות שקיבלנו.  
- שיטה 1: התיוג עבור קובץ יהיה לפי הכללי הדומיננטי ביותר (הראשון משמאלי בלבד).  
- שיטה 2: התיוג עבור קובץ יהיה שמות כל הכלים, בלי התחשבות בז'אנר.  
- שיטה 3: התיוג עבור קובץ יהיה הכללי הדומיננטי ביותר ווגhz'אנר  
- שיטה 4: סיוג אך ורק לפי ז'אנר.

• **בעיה 4: סיוג כל הנגינה בשיר בודד קלט**  
המטרה שלנו כאן הייתה לנסות לזהות את כל הנגינה בשיר בודד המתקבל כקלט.  
הקלט שלנו הוא קטע שמע (wav) וקובץ (txt). המתאר את כל הנגינה המתנהגים בשיר.  
הבעיה העיקרית הייתה שכאשר אנו מנסים לחזות את כל הנגינה בשיר בעזרת אלגוריתמי  
הלמידה נקבל סיוג בודד של כל.  
גם השימוש במאגר של כלים משלבים לא עזר כיון שלא כל השילובים קיימים.

**פתרונות:**  
לכן חשבנו על רעיון ובנו אלגוריתם הפועל כך:  
1. חתוך את השיר למקטעים (chunks) באורך של שנייה.  
2. בצע predict על המקטעים.  
3. סדר את המערך לפי כמות המופיעים (כל הסוג הכלי הרבה יהיה במקום הראשון וכן  
הלאה...)  
4. בצע *sequence* על המערך.  
5. החזר את כלים הראשונים כאשר  $k =$ מספר הכלים של השיר המקורי שהתקבל כקלט.

## טכניות:

### **אלגוריתמים:**

4 אלגוריתמי הלמידה בהם השתמשנו הם:

- 1. KNN
- 2. SVM
- 3. Random Forest
- 4. Adaboost (\*חושבים לשנות לאלגוריתם אחר\*)

ספרייה: sklearn

### **מחלקות סיווג:**

פתחנו מספר מחלקות כאשר כל אחת מביצעת פעולה סיווג שונה. המחלקות נמצאות תחת התיקייה .Classifications

- `on_train_classification.py`
- במחלקה זו אנו מבצעים סיווג פשוט על המאגר "data-set" המכיל כלים בודדים.
- test size = 0.25
- (הדפסות של התוצאות עם confusion matrix נמצאות בעמוד הבא)

נשים לב כי לפי התוצאות שקיבלנו SVM נתן לנו את התוצאה הטובה ביותר בידוק של 0.97  
אחריו KNN בדיק של 0.95, Random forest בדיק של 0.93 ולבסוף Adaboost בדיק נמוך  
מאוד של 0.13.  
החלק המעניין ב Adaboost הוא ה-confusion matrix, ניתן לראות כי רוב הסיווגים מתרכזים  
בחלק הימני, כאשר הרוב המוחלט סוג כינור (violin), ואת ה- tuba הצלחנו לסווג יפה.

KNN

```

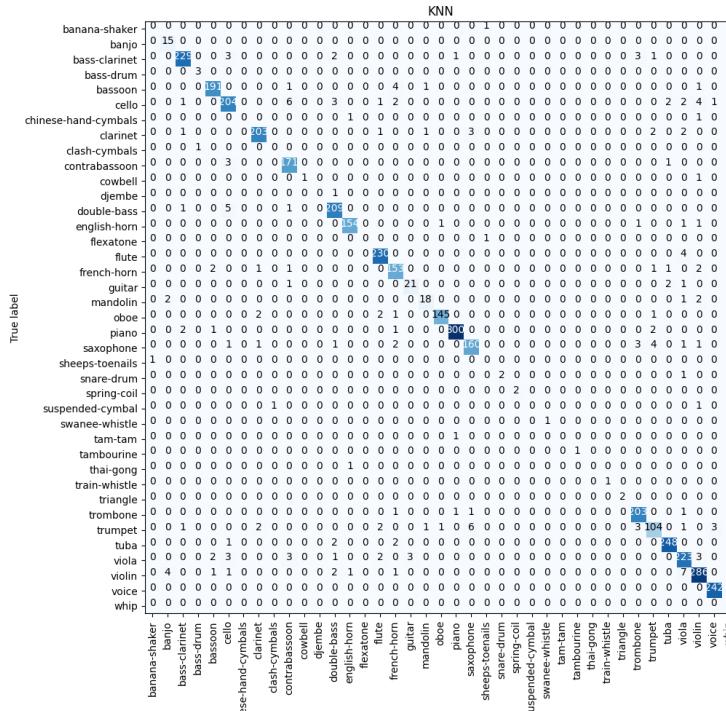
KNN - Evaluation ----

Recall: [0.          1.          0.958159  1.          0.96464646 0.90265487
0.          0.95305164 0.          0.97714286 0.5         0.
0.96759259 0.97468354 0.          0.98290598 0.95031056 0.84
0.7826087  0.9602649  0.98039216 0.91954023 0.          0.66666667
1.          0.          1.          0.          1.          0.
1.          1.          0.98067633 0.83870968 0.98023715 0.92916667
0.94389439 1.          1.          ]]

Precision: [0.          0.71428571 0.97446809 0.75        0.96954315 0.92307692
1.          0.97129187 0.          0.92934783 1.          1.
0.94570136 0.98089172 1.          0.96638655 0.91616766 0.875
0.85714286 0.98639456 0.99009901 0.94117647 0.          1.
1.          1.          1.          1.          1.          1.
1.          1.          0.95305164 0.90434783 0.97637795 0.91020408
0.94389439 0.98373984 0.66666667]

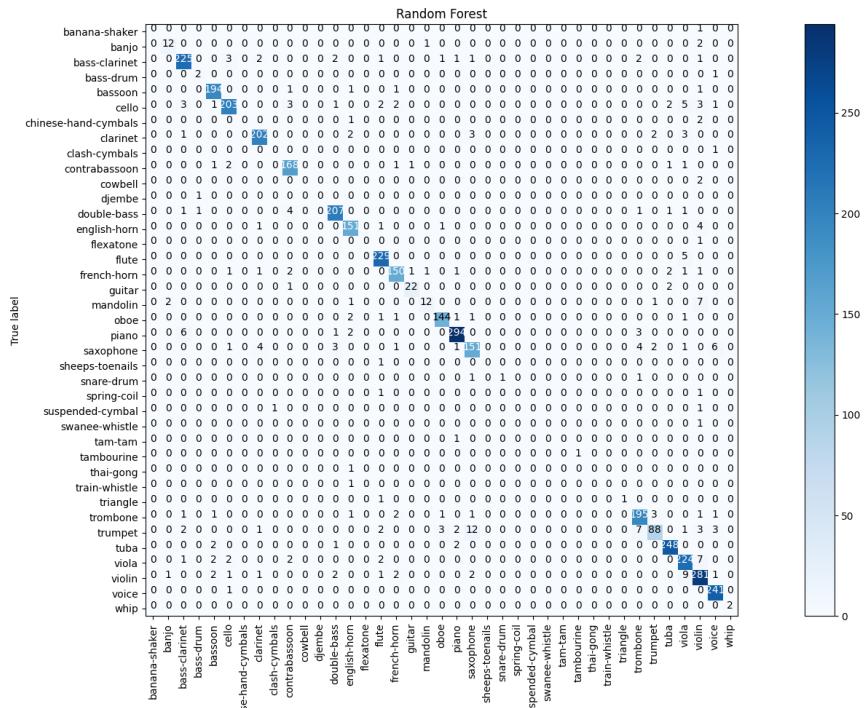
F1-Score: [0.          0.83333333 0.96624473 0.85714286 0.96708861 0.91275168
0.          0.96208531 0.          0.95264624 0.66666667 0.
0.95652174 0.97777778 0.          0.97457627 0.93292683 0.85714286
0.81818182 0.97315436 0.98522167 0.93023256 0.          0.8
1.          0.          1.          0.          1.          0.
1.          1.          0.96666667 0.87029289 0.97830375 0.91958763
0.94389439 0.99180328 0.8       ]

```

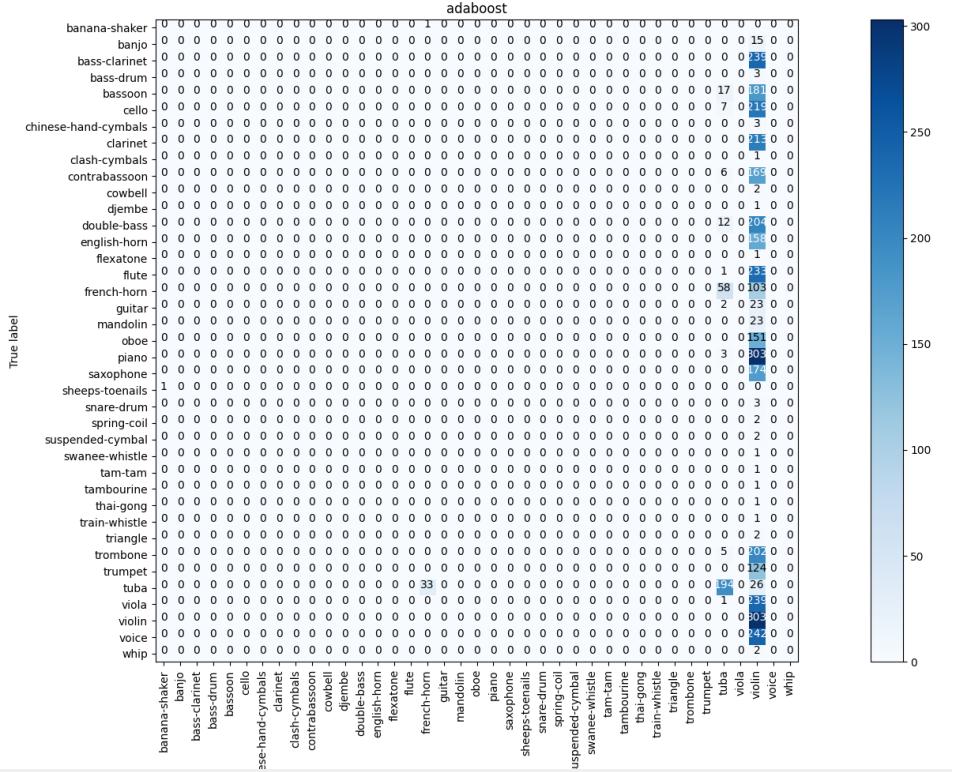


SVM

## Random Forest



## Adaboost



## external\_test\_classification.py •

במחלקה זו אנו מבצעים למידה על המאגר "data-set" המכיל כלים בודדים. ומשתמשים ב"IRMAS-TrainingData" (כלים משלבים) כ-test, כאשר אנחנו מתyiגים לפי הכל' הדומיננטי.

המטרה שלנו כאן היא לנסות לראות האם באמצעות כלים בודדים, ניתן לחזות את הכל' הדומיננטי וביצם להתעלם מרעש רקע.

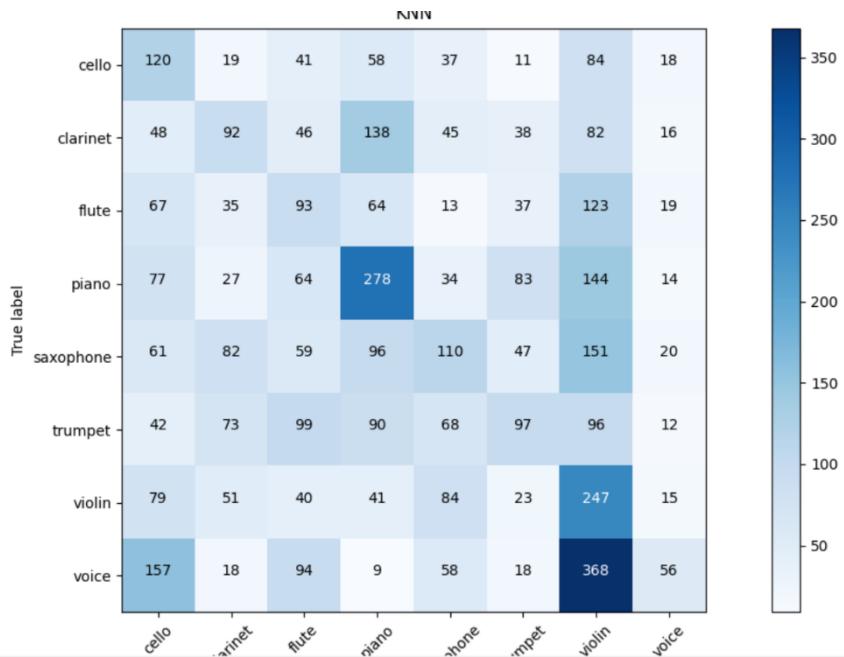
לפי התוצאות, ניתן לראות שרעשי הרקע מאוד משפיעים על הסיווג.

(הדפסות של התוצאות עם confusion matrix נמצאות בעמוד הבא)

נשים לב כי לפי התוצאות שקיבלו SVM, KNN, Random forest בדיק של בערך 0.22-0.24 ולבסוף Adaboost בדיק של 0.19.

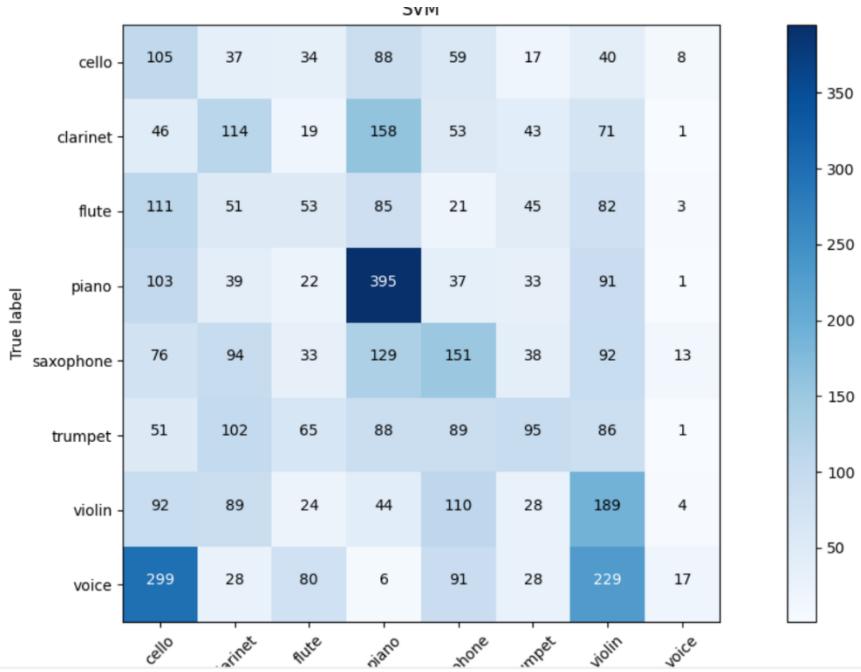
הפעם Adaboost הרבה יותר מואזן. אנחנו מאמינים שבגלל שהtrain רחוק מהtest האופן שבו adaboost עובד הוא מצליח לדיק יותר (אמנם הוא לא כל כך טוב אבל הוא יותר טוב מפעם קודמת).

### KNN



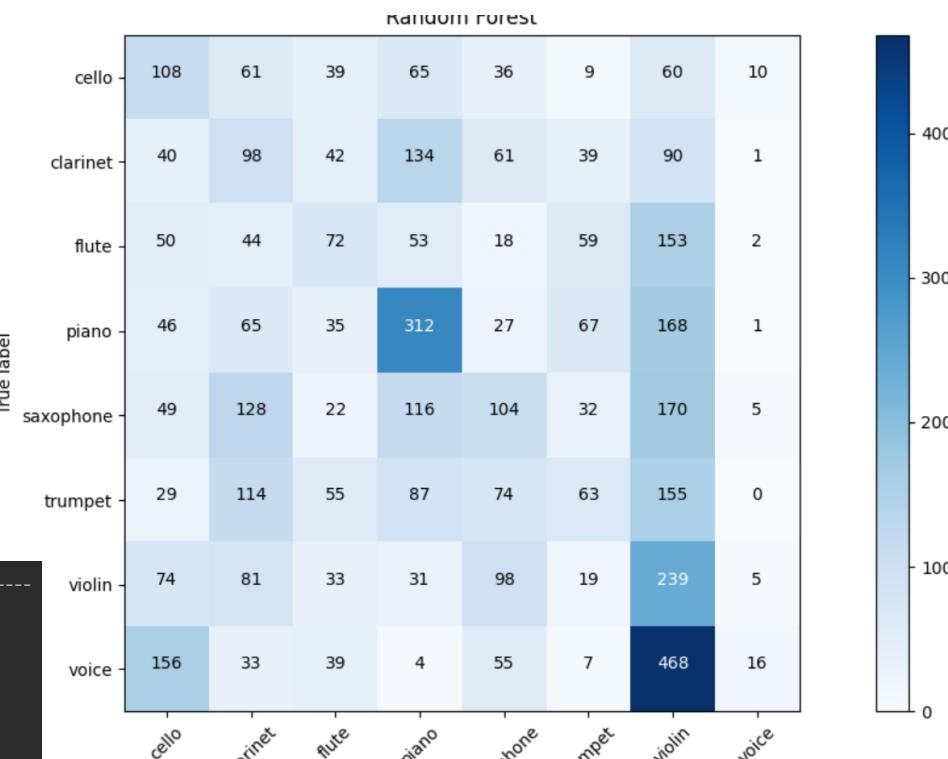
```
----- KNN - Evaluation -----
Recall: [0.30927835 0.18217822 0.20620843 0.38557559 0.17571885 0.16811092
0.42586207 0.07197943]
Precision: [0.1843318 0.23173804 0.17350746 0.35917313 0.24498886 0.2740113
0.19073359 0.32941176]
F1-Score: [0.23099134 0.20399113 0.18844985 0.37190635 0.20465116 0.20837809
0.26346667 0.11814346]
Accuracy: 0.24 , 1093
Number of samples: 4626
```

## SVM



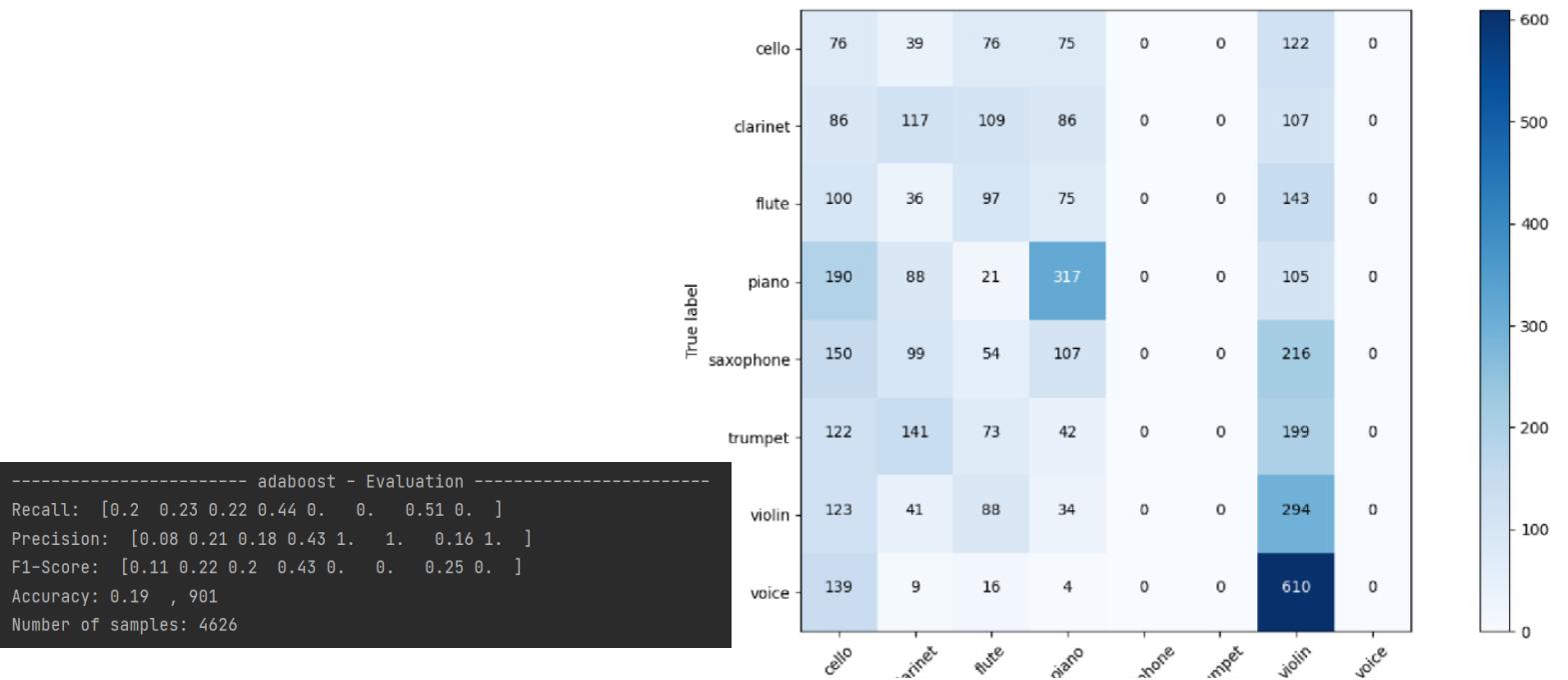
```
----- SVM - Evaluation -----
Recall: [0.27 0.23 0.12 0.55 0.24 0.16 0.33 0.02]
Precision: [0.12 0.21 0.16 0.4 0.25 0.29 0.21 0.35]
F1-Score: [0.17 0.22 0.14 0.46 0.24 0.21 0.26 0.04]
Accuracy: 0.24 , 1119
Number of samples: 4626
```

## Random Forest



```
----- Random Forest - Evaluation -----
Recall: [0.28 0.19 0.16 0.43 0.17 0.11 0.41 0.02]
Precision: [0.2 0.16 0.21 0.39 0.22 0.21 0.16 0.4 ]
F1-Score: [0.23 0.17 0.18 0.41 0.19 0.14 0.23 0.04]
Accuracy: 0.22 , 1012
Number of samples: 4626
```

## Adaboost



## complex\_classification.py - part 1 •

במחלקה זו אנו מביצים למידה על המאגר "IRMAS-TrainingData" המכיל כלים משולבים עם כל מרכז. ומחלקים אותו ל 25% test.

המטרה שלנו כאן היא לנסות לראות האם באמצעות כלים משולבים, ניתן לחזות את הכלים הדומיננטי ובעצם להתעלם מרעש רקע. כמו שעשינו פעם קודמת אבל הפעם לומדים על כלים משולבים באמצעות כלים משולבים.

כאן השארנו עוד תוצאות חוץ מהכללי המרכזי ולכן אפשר לראות שהסתוווג יותר טוב ממה קודם אך עדין סיבוב ה 50% אחוז.

(הדפסות של התוצאות עם confusion matrix נמצאות בעמוד הבא)

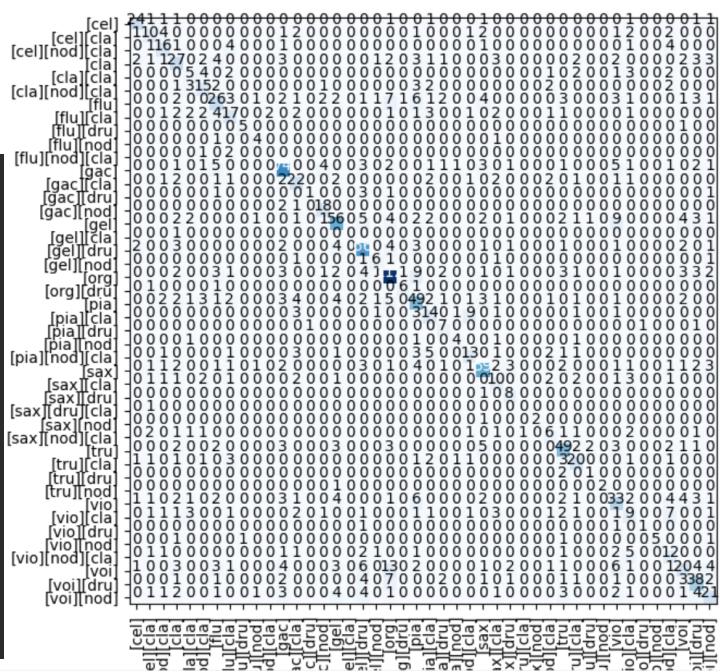
KNN

```

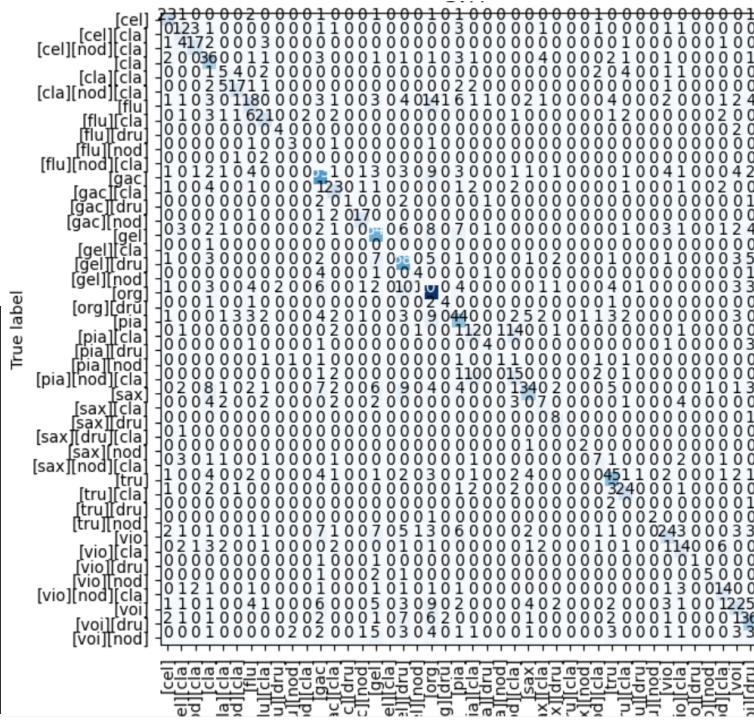
Precision: [ 0.75      0.41666667 0.5      0.421875   0.26315789 0.46875
 0.44067797 0.425      0.83333333 0.5      0.        0.66666667
 0.47826087 0.5      0.64285714 0.64367816 1.        0.62264151
 0.5      0.6746988 0.75      0.49      0.37837838 0.41176471
 0.57142857 0.43333333 0.67045455 0.33333333 0.44444444 0.
 1.        0.31578947 0.6125      0.51282051 0.25      1.
 0.42857143 0.25      0.5      1.        0.3      0.44444444
 0.55072464 0.46666667]

F1-Score: [ 0.73846154 0.39215686 0.52459016 0.42519685 0.25641026 0.47619046
 0.39097744 0.40963855 0.83333333 0.57142857 0.        0.67579909
 0.50574713 0.22222222 0.73469388 0.59893048 0.        0.67005076
 0.52173913 0.69565217 0.70588235 0.51041667 0.3943662 0.51851852
 0.53333333 0.41935484 0.64835165 0.34482759 0.59259259 0.
 0.8        0.31578947 0.62025316 0.52631579 0.28571429 0.8
 0.43708609 0.24      0.4      0.71428571 0.35294118 0.32786885

```

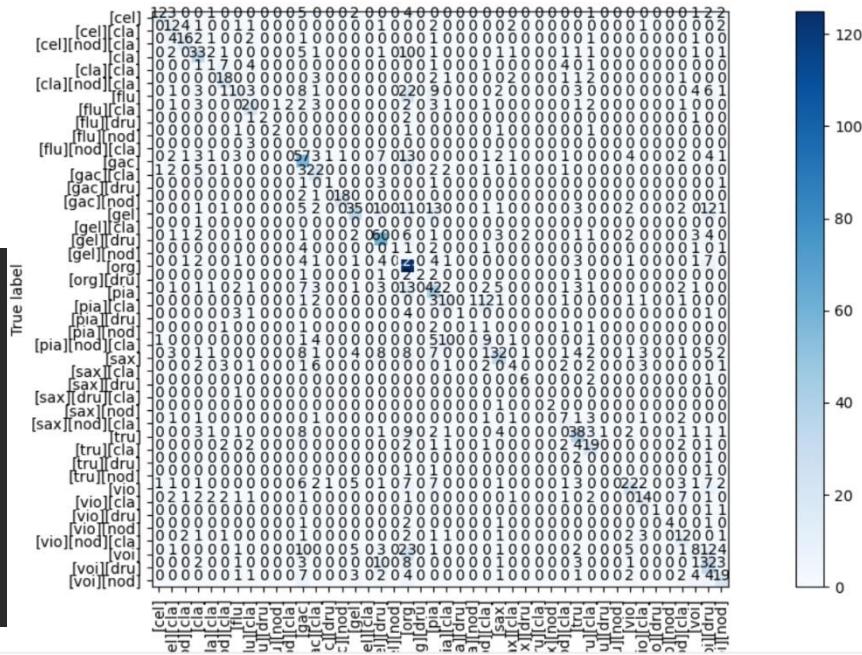


SVM

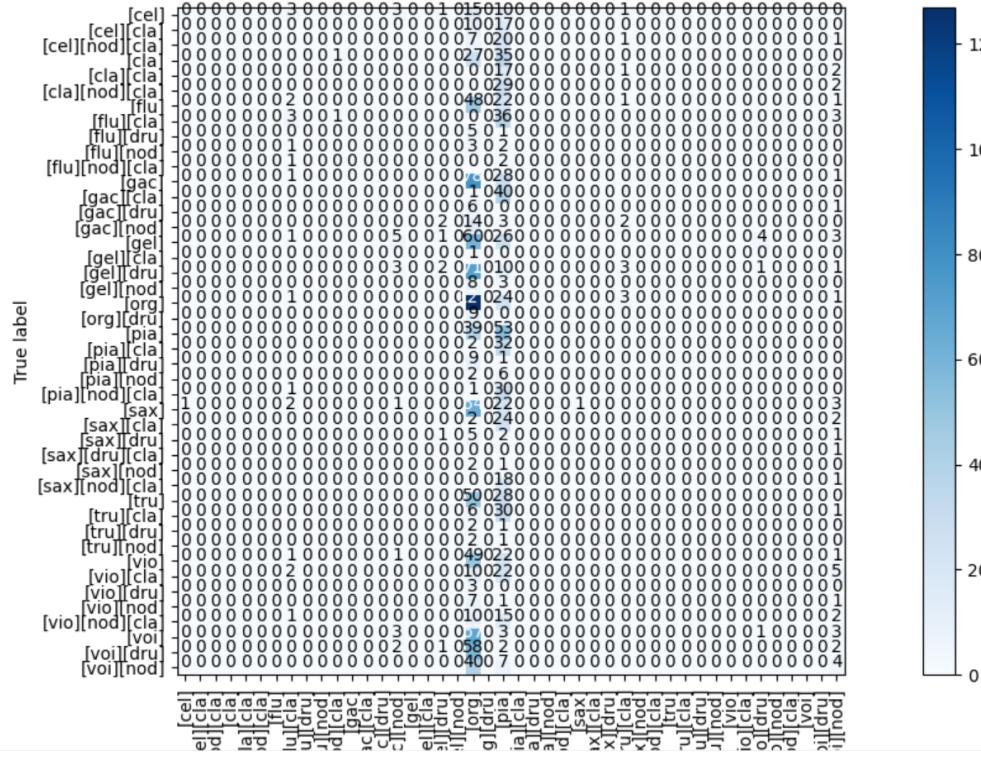


# Random Forest

```
-- Random Forest - Evaluation --
Recall: [0.36 0.44 0.55 0.52 0.05 0.58 0.14 0.47 0.33 0.33 0. 0.53 0.54 0.14
         0.86 0.35 0. 0.66 0.09 0.8 0.22 0.46 0.29 0.1 0.12 0.28 0.34 0.14
         0.67 0. 0.67 0.37 0.49 0.51 0. 0. 0.3 0.36 0.33 0.44 0.43 0.1
         0.49 0.37]
Precision: [0.8 0.32 0.62 0.46 0.09 0.47 0.36 0.49 1. 0.67 0. 0.37 0.39 0.33
            0.95 0.6 1. 0.52 1. 0.44 1. 0.36 0.32 1. 0.5 0.26 0.54 0.29
            0.67 1. 1. 0.28 0.51 0.41 0. 1. 0.49 0.5 1. 1. 0.3 0.28
            0.31 0.43]
F1-Score: [0.5 0.38 0.58 0.49 0.06 0.52 0.2 0.48 0.5 0.44 0. 0.43 0.45 0.2
            0.9 0.44 0. 0.58 0.17 0.57 0.36 0.4 0.31 0.18 0.2 0.27 0.42 0.19
            0.67 0. 0.8 0.32 0.5 0.46 0. 0. 0.37 0.42 0.5 0.62 0.35 0.15
            0.38 0.4 ]
Accuracy: 0.44 , 730
Number of samples: 1677
```



## Adaboost



## complex\_classification.py - part 2 •

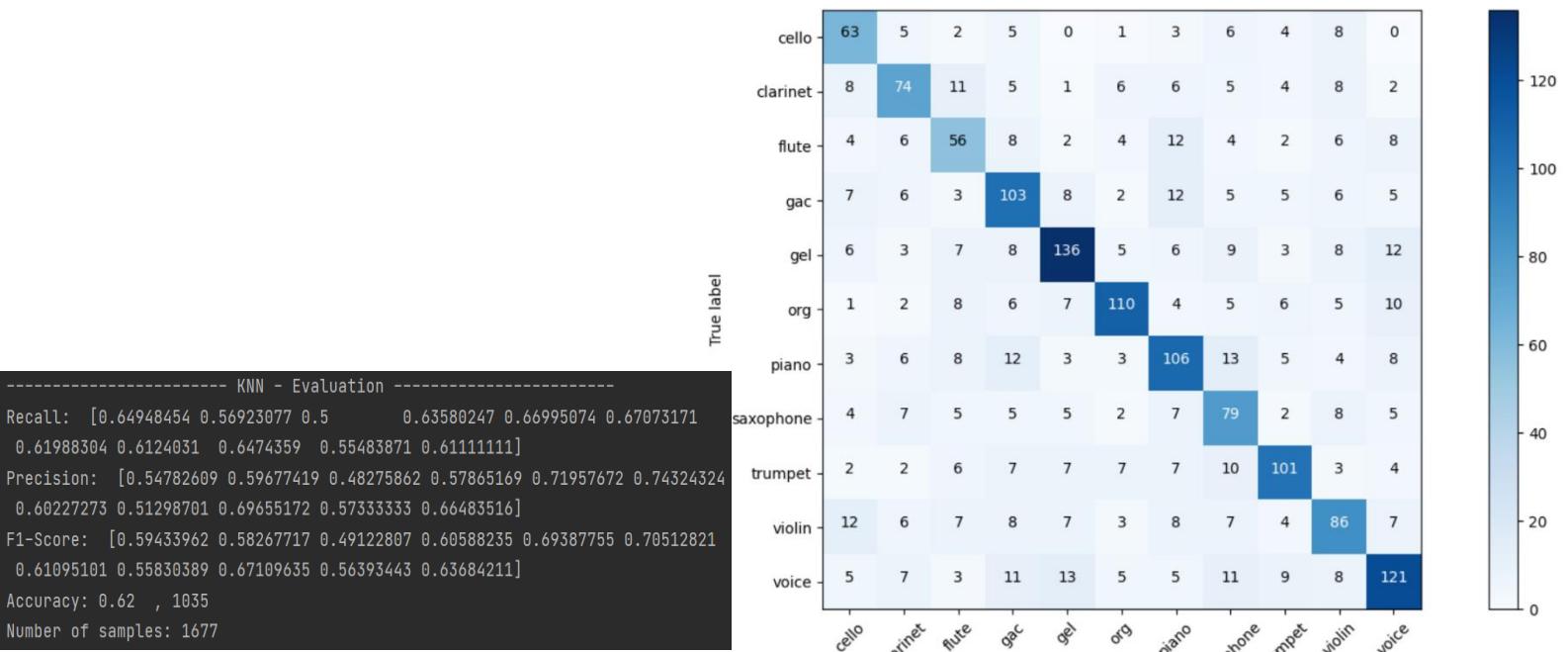
במחלקה זו אנו מבצעים למידה על המאגר "IRMAS-TrainingData" המכיל כלים משולבים עם כל מרכז. ומחלקים אותו ל 0.25 אחוז test.

ההרצאה השנייה הייתה באותו class אבל טעינת המידע הייתה בצורה שונה, ניסיון זה היה יותר כללי, התיאוג היה כלי בודד ולא כלי וגם רעש הרקע. אפשר לראות שהרצאה זאת היא יותר טובה מהחלק הראשון.

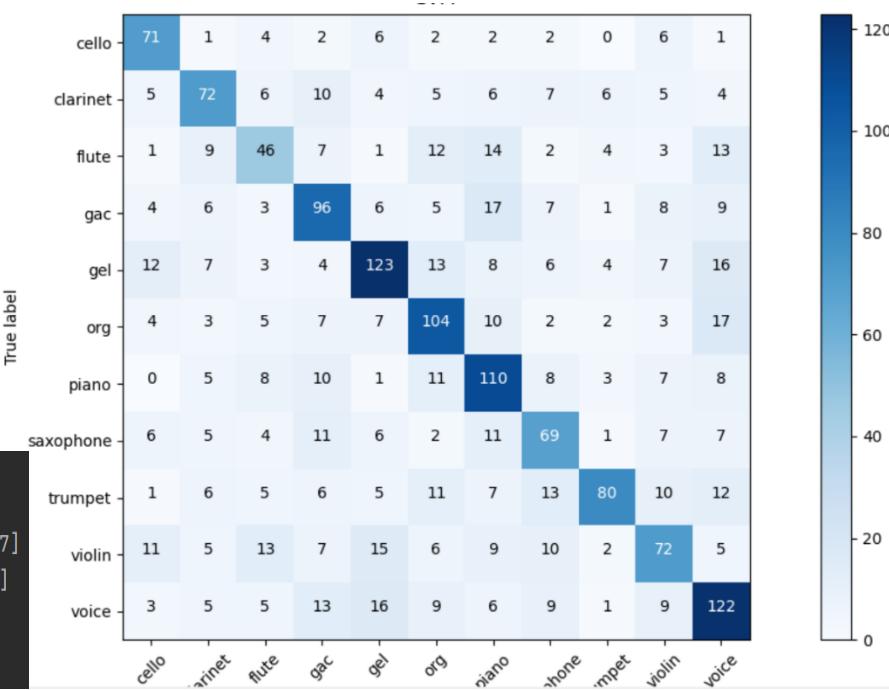
העליה באחוזים היא בכל האלגוריתמים, אפשר גם לראות שמדובר שהוא לא טוב כמו כולם גם הוא עולה בהתאם.

(הדפסות של התוצאות עם confusion matrix נמצאות בעמוד הבא)

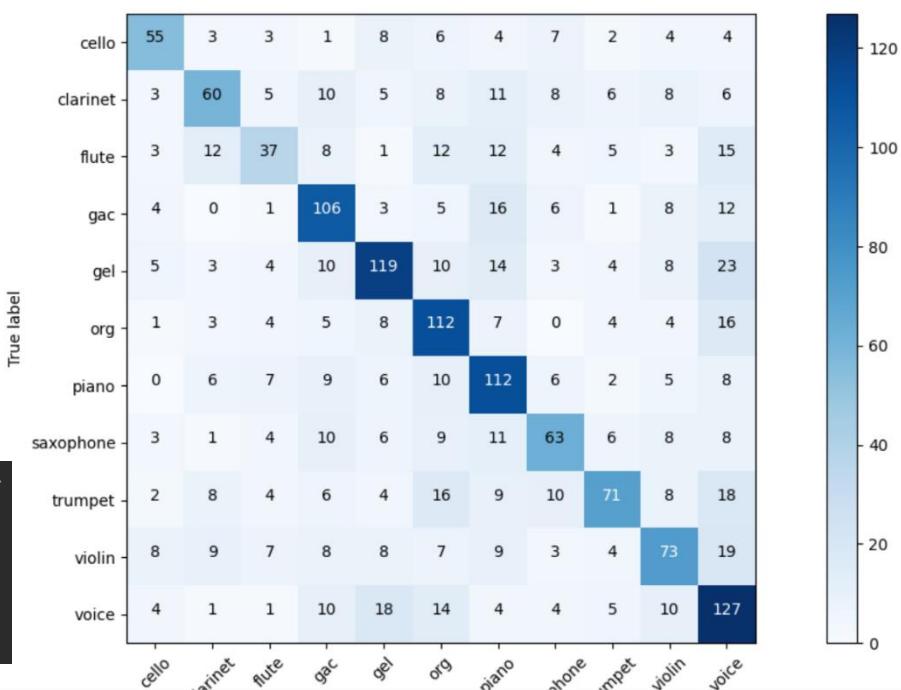
### KNN



### svm

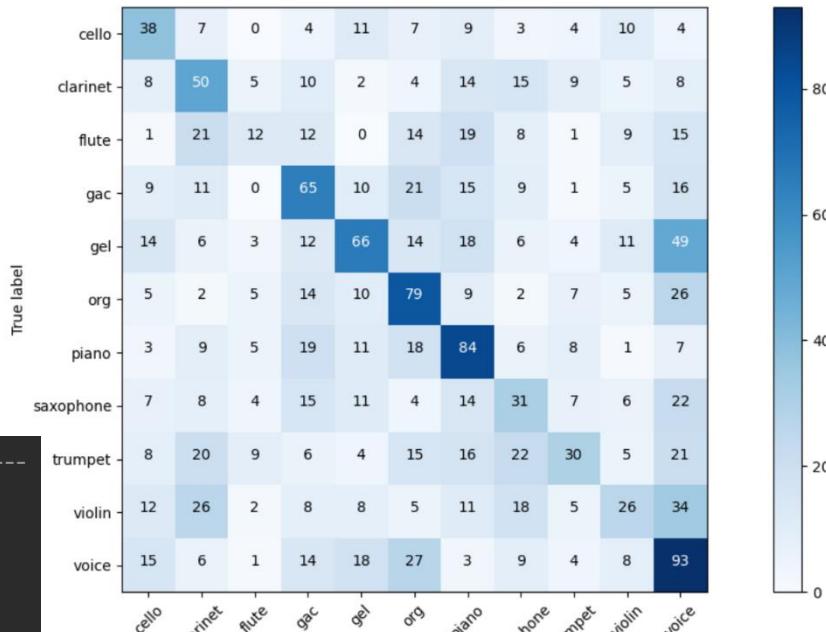


### Random Forest



## Adaboost

```
----- adaboost - Evaluation -----
Recall: [0.39 0.38 0.11 0.4 0.33 0.48 0.49 0.24 0.19 0.17 0.47]
Precision: [0.32 0.3 0.26 0.36 0.44 0.38 0.4 0.24 0.38 0.29 0.32]
F1-Score: [0.35 0.34 0.15 0.38 0.37 0.42 0.44 0.24 0.25 0.21 0.38]
Accuracy: 0.34 , 574
Number of samples: 1677
```



## bigger\_vector\_clf.py •

במחלקה זו אנו מבצעים למידה על המאגר "IRMAS-TrainingData" המכיל כלים משולבים עם כל מרכז. ומחלקים אותו ל 0.25 אחוז test.

בharaza זאת בינוי וקטור של מידע גדול יותר עם יותר פיצרים. (מ 13 ל 20)

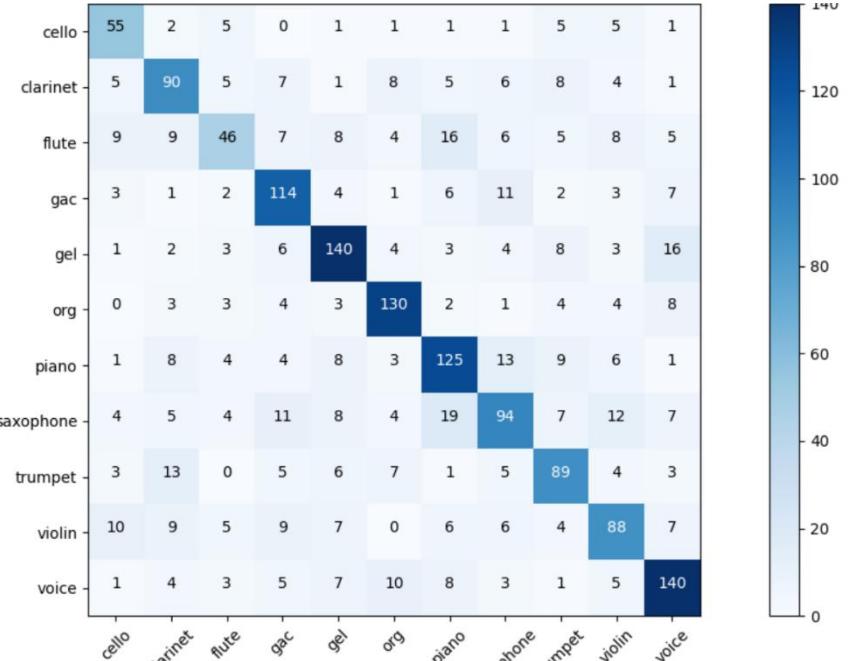
אפשר לראות שהharaza זאת היא יותר טובה בקצת.

כאן SVM יותר טוב עם 0.68 ו KNN עם 0.66

0.59 עם Random Forest

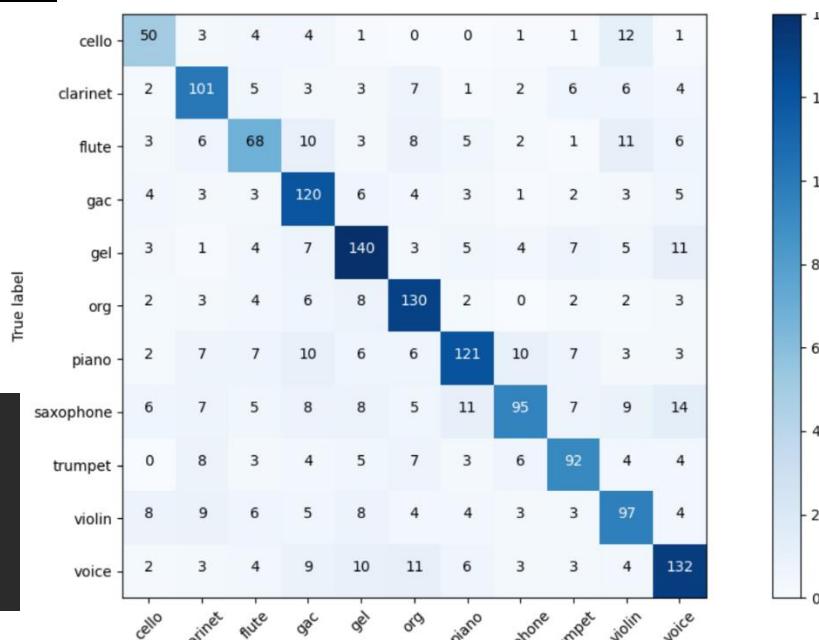
(הdfsotot של התוצאות עם confusion matrix נמצאות בעמוד הבא)

## KNN



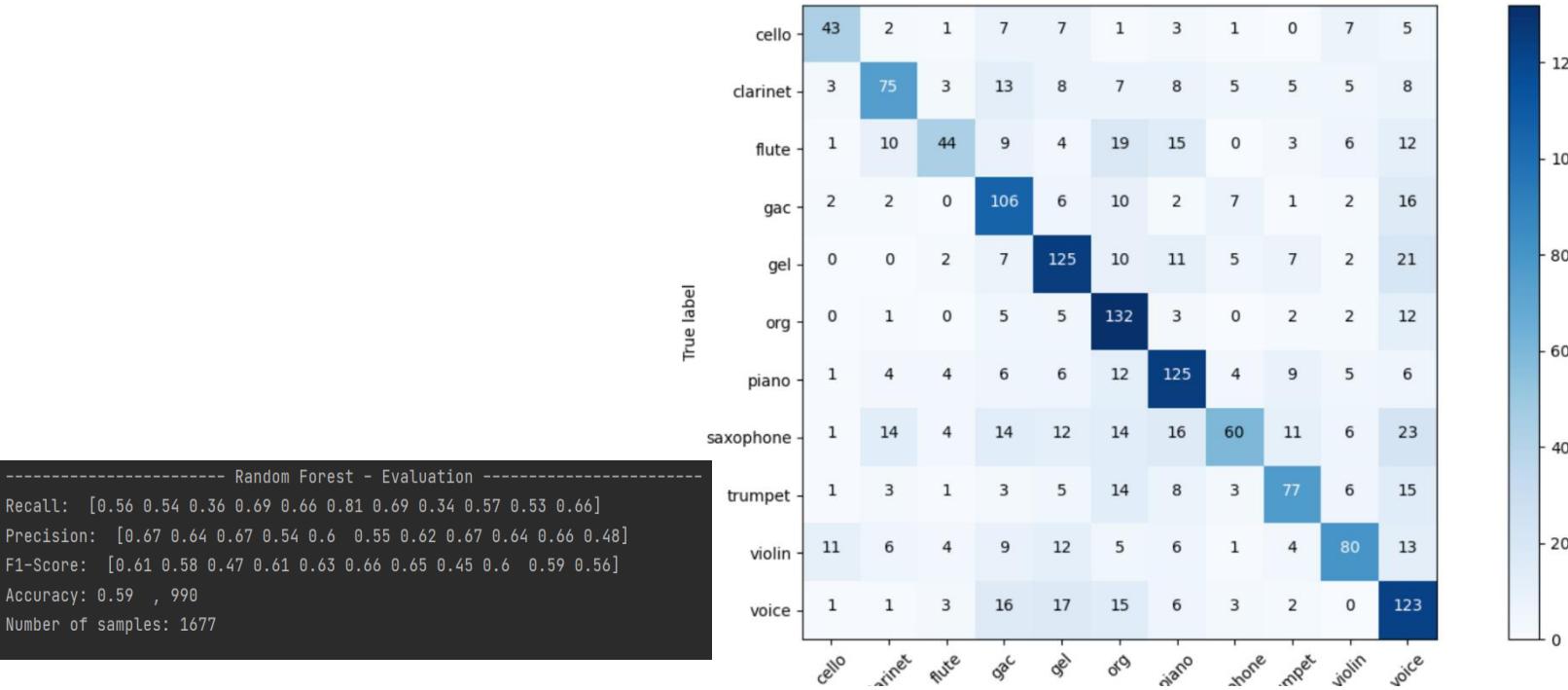
```
----- KNN - Evaluation -----
Recall: [0.71428571 0.64285714 0.37398374 0.74025974 0.73684211 0.80246914
0.68681319 0.53714286 0.65441176 0.58278146 0.7486631]
Precision: [0.59782609 0.61643836 0.575 0.6627907 0.7253886 0.75581395
0.65104167 0.62666667 0.62676056 0.61971831 0.71428571]
F1-Score: [0.65088757 0.62937063 0.45320197 0.6993865 0.7310705 0.77844311
0.6684492 0.57846154 0.64028777 0.60068259 0.7310705]
Accuracy: 0.66 , 1111
Number of samples: 1677
```

## SVM



```
----- SVM - Evaluation -----
Recall: [0.65 0.72 0.55 0.78 0.74 0.8 0.66 0.54 0.68 0.64 0.71]
Precision: [0.61 0.67 0.6 0.65 0.71 0.7 0.75 0.75 0.7 0.62 0.71]
F1-Score: [0.63 0.69 0.58 0.71 0.72 0.75 0.71 0.63 0.69 0.63 0.71]
Accuracy: 0.68 , 1146
Number of samples: 1677
```

## Random Forest



## joined\_data\_sets\_classification.py •

במחלקה זו אנו מבצעים למידה על המאגר "IRMAS-TrainingData" המכיל כלים משולבים עם כל מרכז וגם במאגר "data-set" המכיל כלים בודדים. ומחלקים אותו לאחוז 0.25 test.

רצינו לבדוק האם זה יעזר לסווג את הכלים המשולבים בצורה משמעותית או יגרע מס'ווג הכלים הבודדים בצורה משמעותית. הוי לנו ציפיות מהערכת זאת שהרי מאגר הכלים הבודדים גדול כפול 3 ממאגר הכלים המשולבים ולכן באופן טבעי חשבנו שנתקבל סיווג גובה יחסית למאגר הכלים המשולבים לבד. כמו שאפשר לראות בכך באמת מה שהיא כאשר הסיווג עולה ביחס לבדיקה על הכלים המשולבים באותו יחס שהם קיימים במאגר (בערך רביע).

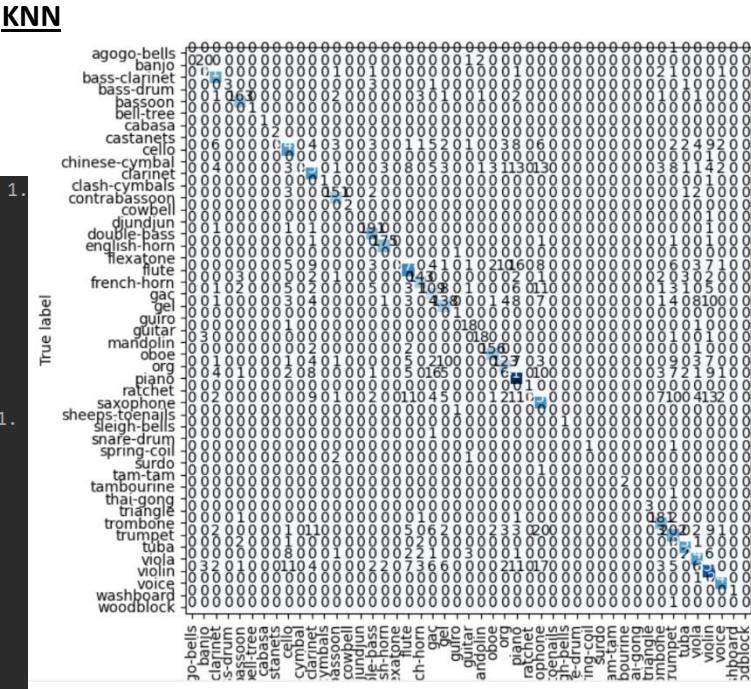
מסקנה: אין משמעות להערכת זו והיא סתם מבלבלת, או שבוחרים לסווג כלים בודדים או משולבים.

0.81 RandomForest , 0.85 SVM

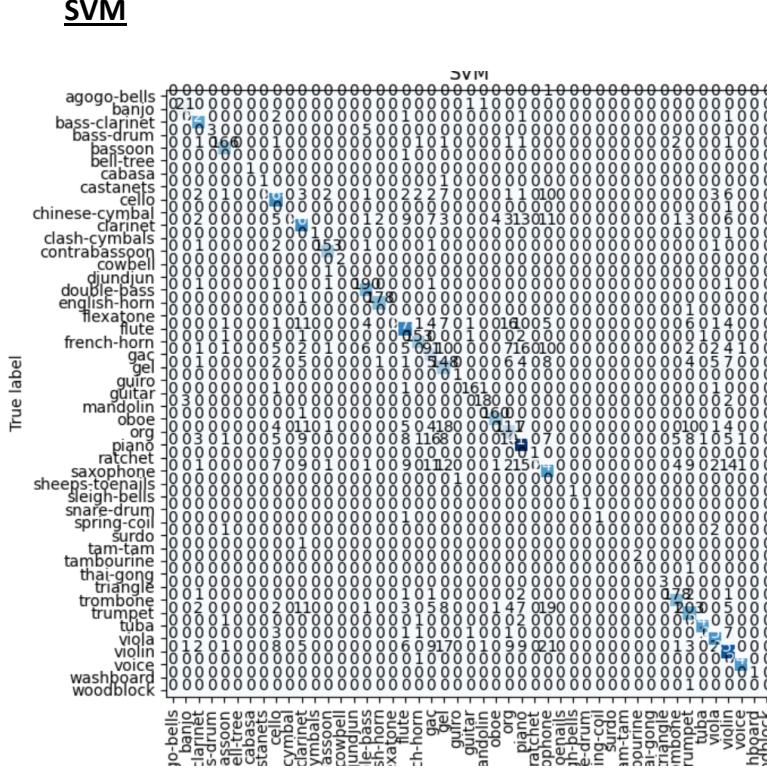
```

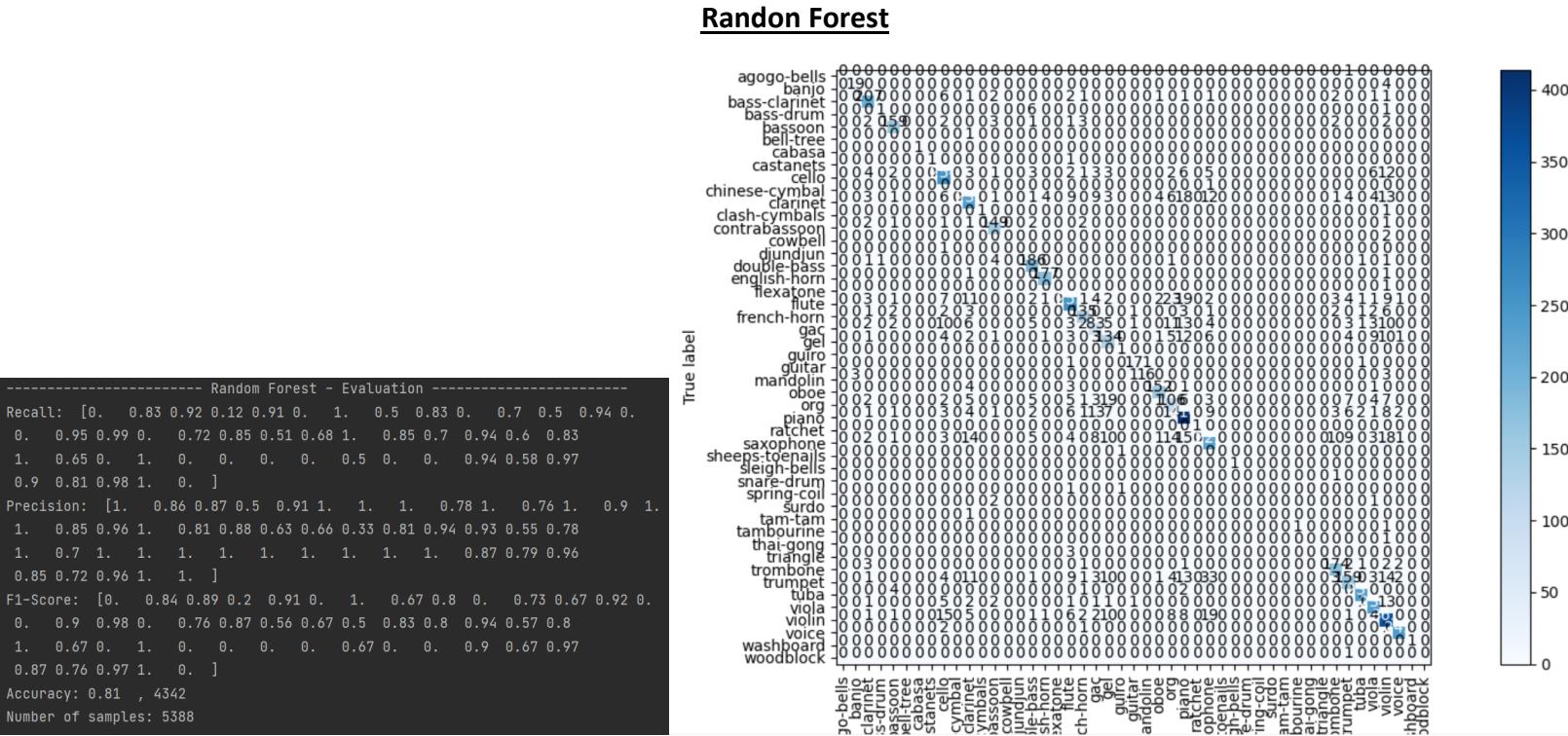
Precision: [1.          0.76923077 0.89754098 1.          0.94219653 1.
1.          1.          0.84482759 1.          0.80877743 1.
0.92073171 1.          1.          0.89671362 0.96685083 1.
0.8404908  0.91666667 0.66463415 0.76243094 0.33333333 0.69230769
0.81818182 0.94545455 0.7987013  0.82248521 1.          0.72316384
1.          1.          1.          1.          1.          1.
1.          1.          1.          0.87439614 0.76226415 0.94820717
0.86071429 0.80361174 0.96078431 1.          1.          ]
F1-Score: [0.          0.81632653 0.93191489 0.54545455 0.93678161 1.
1.          1.          0.82077052 0.          0.78899083 0.66666667
0.93498452 1.          0.          0.93627451 0.97222222 0.
0.81065089 0.90793651 0.66463415 0.73015873 0.5        0.7826087
0.8          0.95705521 0.74545455 0.82985075 1.          0.73775216
0.          1.          0.          0.66666667 0.          0.
1.          0.          1.          0.92111959 0.75232775 0.96161616
0.88117002 0.8          0.97804391 1.          0.          ]
Accuracy: 0.85 , 4557
Number of samples: 5388

```



```
SVM - Evaluation  
Recall: [0. 0.91 0.98 0.38 0.95 0. 1. 0.5 0.86 0. 0.79 0.5 0.96 1.  
0. 0.97 0.99 0. 0.79 0.96 0.55 0.75 1. 0.8 0.78 0.99 0.63 0.82  
1. 0.71 0. 1. 1. 0.5 0. 0. 1. 0. 1. 0.96 0.75 0.98  
0.94 0.79 1. 1. 0. ]  
Precision: [1. 0.84 0.92 1. 0.95 1. 1. 1. 0.84 1. 0.79 1. 0.96 1.  
1. 0.9 0.98 1. 0.84 0.95 0.58 0.62 0.5 0.8 0.86 0.96 0.65 0.82  
1. 0.72 1. 1. 1. 1. 1. 1. 1. 1. 0.93 0.8 0.99  
0.93 0.83 0.99 1. 1. ]  
F1-Score: [0. 0.87 0.95 0.55 0.95 0. 1. 0.67 0.85 0. 0.79 0.67 0.96 1.  
0. 0.94 0.99 0. 0.82 0.96 0.57 0.68 0.67 0.8 0.82 0.98 0.64 0.82  
1. 0.72 0. 1. 1. 0.67 0. 0. 1. 0. 1. 0.94 0.77 0.99  
0.94 0.81 0.99 1. 0. ]  
Accuracy: 0.85 , 4573  
Number of samples: 5388
```



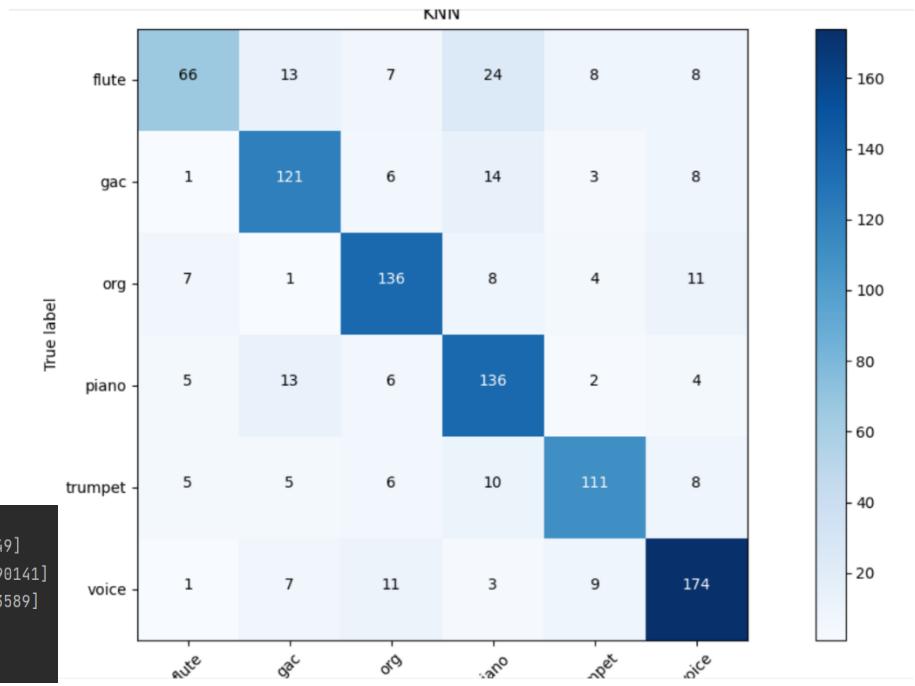


bigger\_vector\_less\_labels.py •

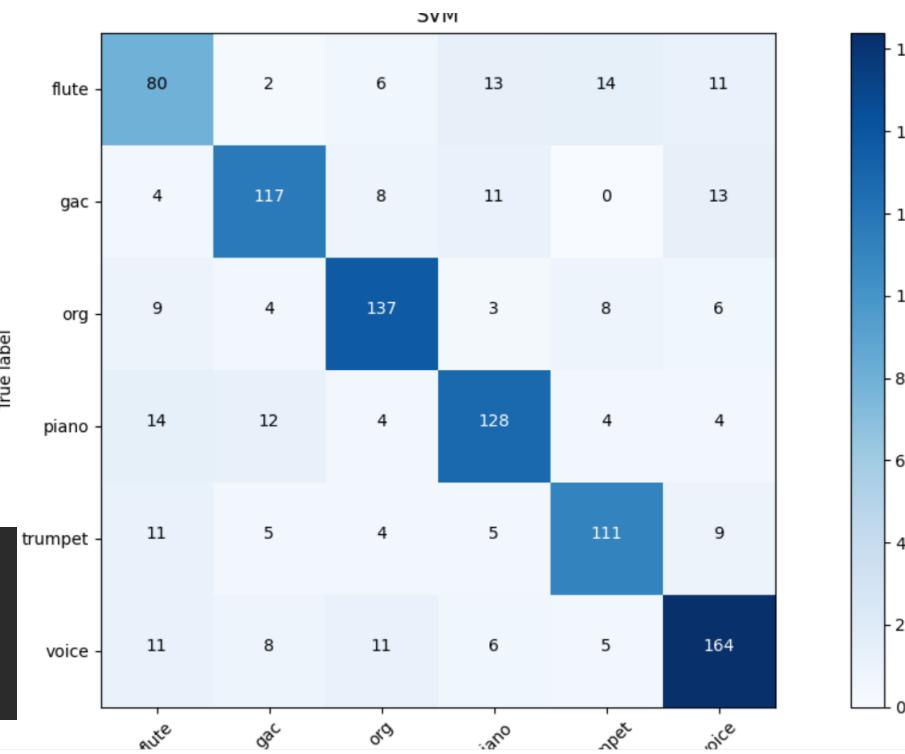
במחלקה זו אנו מבצעים למידה על המאגר "IRMAS-TrainingData" המכיל כלים מושלבים ומחלקים אותו ל-0.25 אוחז.test.  
עשינו ניסוי עם שימוש בPCA ובל', ולא היה שינוי. יותר מזה, כאשר ביצענו שימוש עם PCA (components=2) התוצאה ירדה מאד. אנחנו חושבים שהסתיבה לכך היא דומה לשיבת בה adaboost לא עבד. כאשר העלנו את כמות הוקטור התוצאה עלה ונכ枉 בהתאם כאשר הורדנו את כמות הפיצרים ל-2, התוצאה ירדה.

בברכה זו השתמשנו בחצי מכמומיות התויוגים, כאשר התויוגים שבחרנו לקחת הם התויוגים שלאינו באחד המחקרים שקרו לנו שם יביאו את התוצאות הטובות ביותר. נשים לב שככל ההרצאות קיבלו תוצאות טובות יותר כאשר הן משתמשים בוקטור גדול יותר ביצוג של כל דגימה.

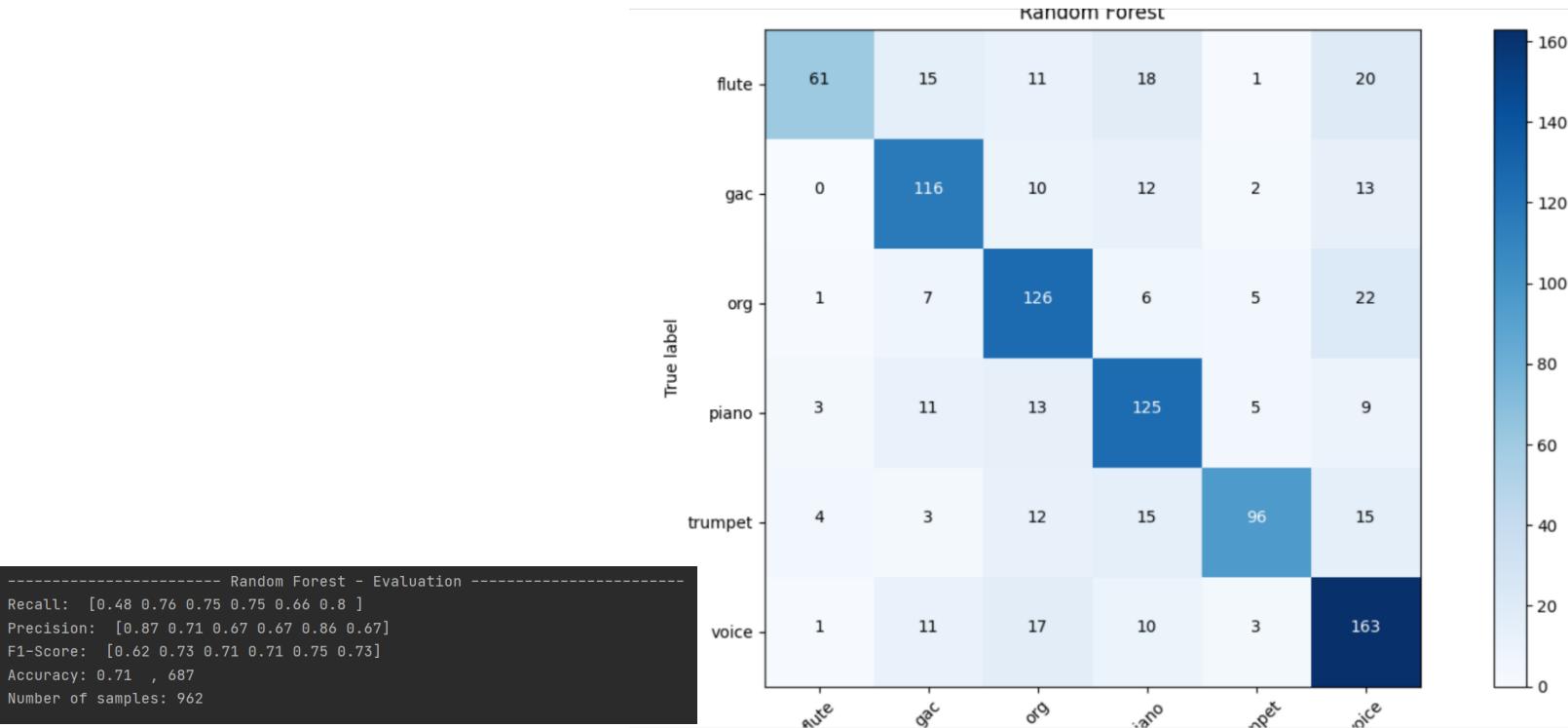
## KNN



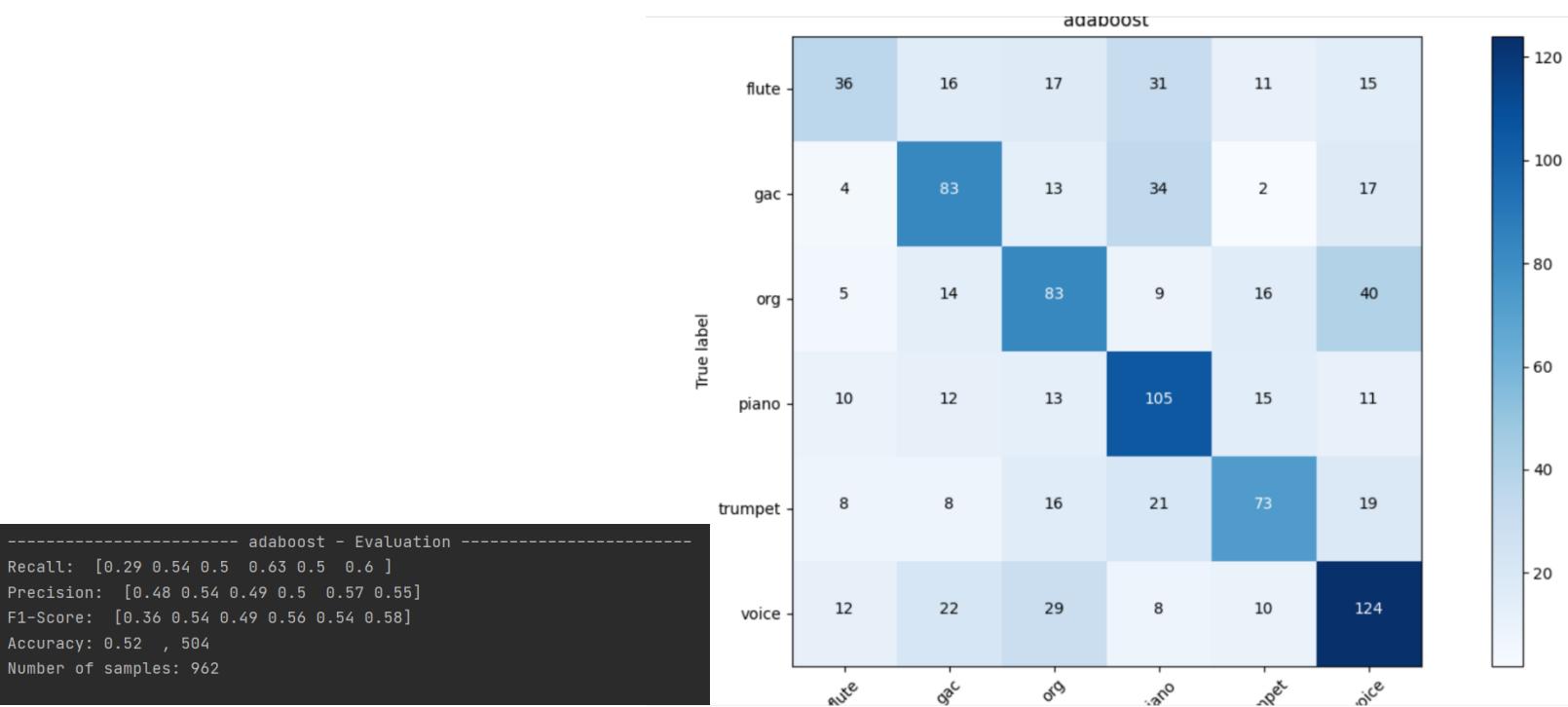
## SVM



## Random Forest



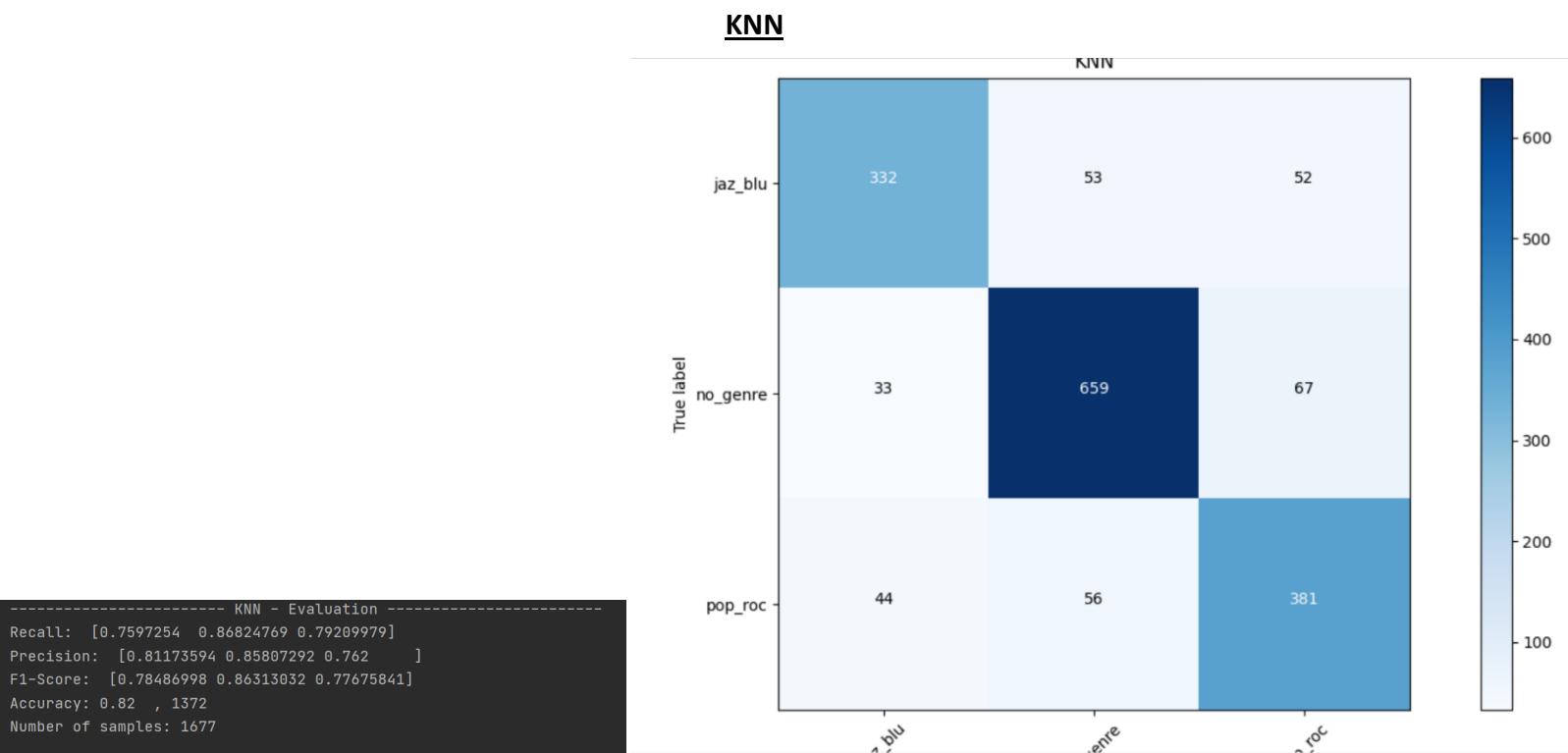
## Adaboost



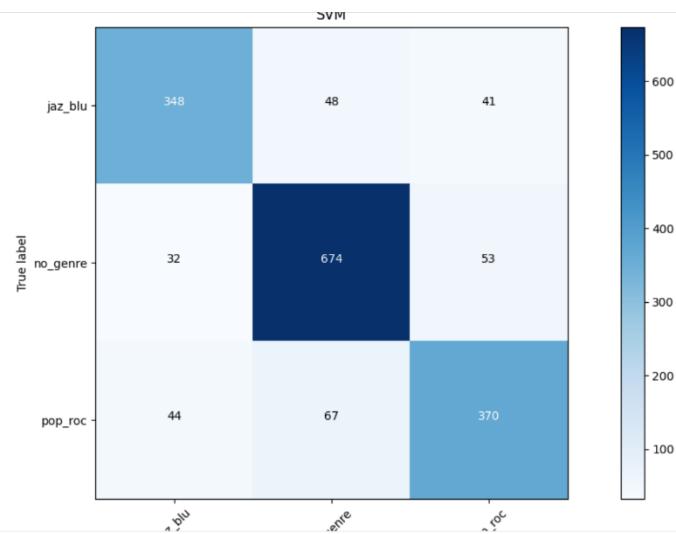
במחלקה זו אנו מבצעים למידה על המאגר "IRMAS-TrainingData" המכיל כלים משולבים עם כלי מרכזי. כאן אנחנו מבצעים סיווג אך ורק לפי הז'אנר בלבד התיחסות בכלים המתנגנים. מחלוקת ל-0.250 אוחז test.

ניתן לראות שאט הז'אנרים אנחנו מצליחים לסיווג בצורה טובה מאוד. תוצאה זו מעניינת וחשובה מכיוון שהוא מדגיש את חשיבותו של הzinser על הסיווג, שכן יש לו השפעה גדולה בלבד התיחסות בכלים המתנגנים בו.

(הדפסות של התוצאות עם confusion matrix נמצאות בעמוד הבא)

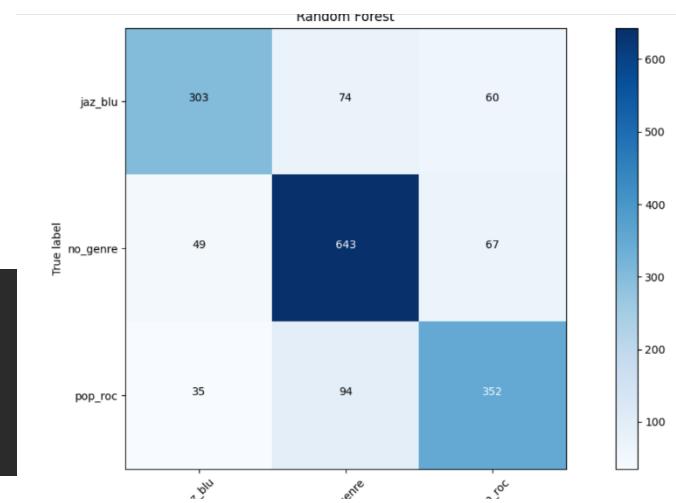


## SVM



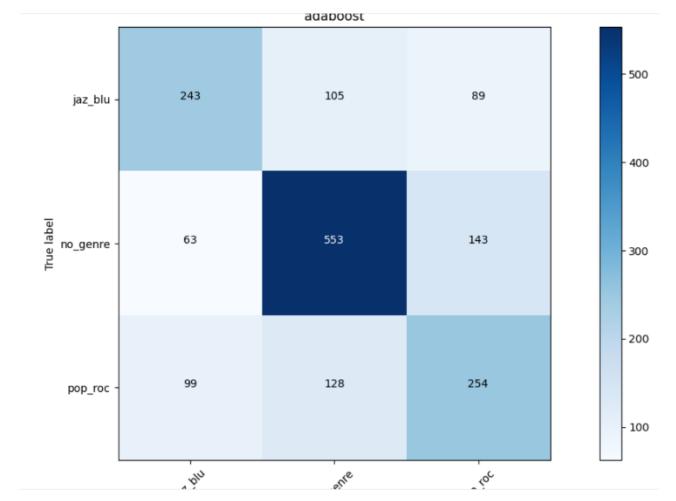
```
----- SVM - Evaluation -----
Recall: [0.8  0.89 0.77]
Precision: [0.82 0.85 0.8 ]
F1-Score: [0.81 0.87 0.78]
Accuracy: 0.83 , 1392
Number of samples: 1677
```

## Random Forest



```
----- Random Forest - Evaluation -----
Recall: [0.69 0.85 0.73]
Precision: [0.78 0.79 0.73]
F1-Score: [0.74 0.82 0.73]
Accuracy: 0.77 , 1298
Number of samples: 1677
```

## Adaboost



```
----- adaboost - Evaluation -----
Recall: [0.56 0.73 0.53]
Precision: [0.6 0.7 0.52]
F1-Score: [0.58 0.72 0.53]
Accuracy: 0.63 , 1050
Number of samples: 1677
```

## song\_instruments\_classification.py •

במחלקה זו אנו מנסים לסווג את כל הנגינה בשיר שלם המתקבל כקלט לפי האלגוריתם שתיארנו (בעיה 4).

ניסינו לעשות זאת במספר דרכיים כאשר השתמשנו ב2 המאגרים שיש לנו, כל אחד לחוד וגם באופן משולב.

לצערנו התוצאות שקיבלנו לא מרשימות. לעיתים הצלחנו לסווג (כל אחד או יותר מתוך התוצאות שקיבלנו) ולעתים לא הצלחנו לסווג אפילו כל אחד נכון.

לפי הבדיקות וההסתכלות שלנו נראה שרushi הרקע גם כאן הם הגורם לחוסר הדיקוק.