**פרויקט גמר – למידת מכונה**  
תכנון הפרויקט **חשוב! בעת הרצת קובץ ה – main יש להוריד מההערה את המודל הרלוונטי בו נרצה להשתמש. זאת על מנת להפחית את זמן הריצה כי הדאטה סט מורכב וכבד.  
אם בחרת בהרצת Adaboost עליך "להדליק" את מודל ה – KNN בנוסף.**

**הData Base בו נשתמש:** <https://www.kaggle.com/datasets/chrisfilo/urbansound8k/data>

ה-Dataset UrbanSound8K מכיל 8,732 קטעי קול עירוניים, כל אחד באורך של עד 4 שניות, הממוינים ל-10 קטגוריות:

* מזגן (air\_conditioner)
* צופר רכב (car\_horn)
* ילדים משחקים (children\_playing)
* נביחת כלב (dog\_bark)
* קידוח (drilling)
* מנוע רדום (engine\_idling)
* יריית רובה (gun\_shot)
* פטיש אוויר (jackhammer)
* סירנה (siren)
* מוזיקה רחוב (street\_music)

**המודלים בהם נשתמש:**

**חיפוש שכן קרוב (K-Nearest Neighbors, KNN) :**  
 KNN משמש לסיווג סוג הרעש (לדוגמה: צופר מכונית, דיבור, צליל שבירה וכו') על בסיס מרחק בין תכונות שנחצבו מנתוני הקול.

סד"פ:

1. תחילה יש לחלץ תכונות (Features) מקטעי הקול, כמו  
    MFCCs (Mel Frequency Cepstral Coefficients Spectrograms) או תכונות סטטיסטיות של התדר והעוצמה.
2. את התכונות הללו נכניס כקלט לאלגוריתם KNN .
3. עבור דגימה חדשה KNN ,יחשב את המרחק בינה לבין דגימות אחרות במרחב התכונות (למשל, על בסיס מרחק אוקלידי).
4. הדגימה החדשה תסווג לפי רוב הקטגוריות של ה-K השכנים הקרובים ביותר.

דוגמה ליישום:

* 1. קטע קול נבדק, והתכונות שלו מחושבות למשל: ממוצע התדר, שונות.
  2. האלגוריתם שואל: "מהם K הקטעים הקרובים ביותר לדגימה זו?"
  3. אם רוב השכנים הקרובים מסומנים כ"דיבור", הדגימה החדשה תסווג כ"דיבור".

**מכונות וקטורים תומכים (SVM) :**

משתמשים ב-SVM כדי לבצע סיווג דו-מעמדי או רב-מעמדי (Multi-Class Classification).

איך משתמשים?

* לאחר חילוץ התכונות, SVM יכולה לזהות גבולות בין קטגוריות קול שונות (כמו צופר לעומת רעש רחוב).
* SVM מתאים במיוחד כאשר יש קבוצות עם גבולות מורכבים אך מעט נתונים.

דוגמה ליישום:

שימוש בגרעינים (Kernels) לא ליניאריים כדי להפריד בין קטגוריות עם תכונות קול חופפות.

**מקבוץ (Clustering) :**

מחלקים את קטעי הקול לקבוצות לפי מאפיינים משותפים, מבלי להשתמש בתוויות.

איך משתמשים?

1. חישוב תכונות כמו Spectral Centroid ו-Zero-Crossing Rate.
2. שימוש בשיטות מקבוץ כמו K-Means לחלוקה של הצלילים לקבוצות.

דוגמה ליישום:  
חלוקה של צלילים לפי טון גבוה/נמוך או משך קצר/ארוך.

**רגרסיה ליניארית:**

אפשר להשתמש ברגרסיה ליניארית אם יש משתנה רציף, כמו ניבוי עוצמת הקול (dB) או תדר ממוצע.

איך משתמשים?

1. תכונות כמו עוצמת הקול הממוצעת ותדר מקסימלי משמשות כמשתנים מסבירים.
2. המודל מנבא ערך רציף כמו העוצמה או משך הצליל.

דוגמה ליישום:  
 ניבוי האם רעש מסוים חורג מרמת רעש חוקית בעיר.

**הגברה (Adaboost) :**

מחזקים ביצועים של מודלים בסיסיים (כמו מסווגים פשוטים או KNN) על ידי יצירת מודל מאוחד שמשיג תוצאות טובות יותר.

איך משתמשים?

1. ניתן ליישם Adaboost על סיווג קטעי הקול בעזרת מסווגים פשוטים כמו SVM.
2. המודל לומד להתמודד עם דוגמאות קשות יותר באמצעות משקלות דינמיות.

שלב מקדים:

1. חילוץ תכונות מקטעי הקול, כמו MFCCs או Spectrograms, כדי להמיר את הנתונים הקוליים למאפיינים נומריים.
2. להשתמש במאפיינים כקלט ל-Adaboost.

דוגמה ליישום:   
שיפור הדיוק בזיהוי צלילים עם רעשי רקע.

**תהליך לדוגמה:**

1. חילוץ תכונות: נשתמש בספריות כמו librosa (בפייתון) כדי לחלץ תכונות אודיו מתקדמות.
2. עיבוד נתונים: ננרמל את התכונות ונפריד בין קבוצות אימון ובדיקה.
3. ביצוע אנליזה: נחבר בין הכלים ונבצע ניתוח של התוצאות.
4. אינטראקציה ויזואלית: ניצור דיאגרמות שיראו את תוצאות המודלים.

**שאלות:**

1. מהו המודל שהצליח הכי טוב? (מהו המודל עם אחוז ההצלחה הגבוה ביותר? למה הוא הצליח?)
2. מהו המודל שהצליח הכי פחות? (מהו המודל עם אחוז ההצלחה הנמוך ביותר? למה זה קרה?)
3. האם המודלים עבדו טוב יותר על רעשים מסוימים? (למשל, האם המודל זיהה צופר מכונית בצורה טובה יותר מאשר רעש של דיבור? חיזוי ערך לעומת קלסיפיקציה?)
4. מהם היתרונות והחסרונות של כל מודל? (מה היתרון של KNN מול SVM? מה החיסרון של Adaboost?)
5. האם יש סוגי רעש שמיוצרים בתדירות גבוהה יותר, והאם זה משפיע על ביצועי המודל? (האם המודל זיהה רעשים שכיחים יותר בצורה טובה יותר?)

ביצוע הפרויקט

**חיפוש שכן קרוב (K-Nearest Neighbors, KNN) :**:  
הקוד:

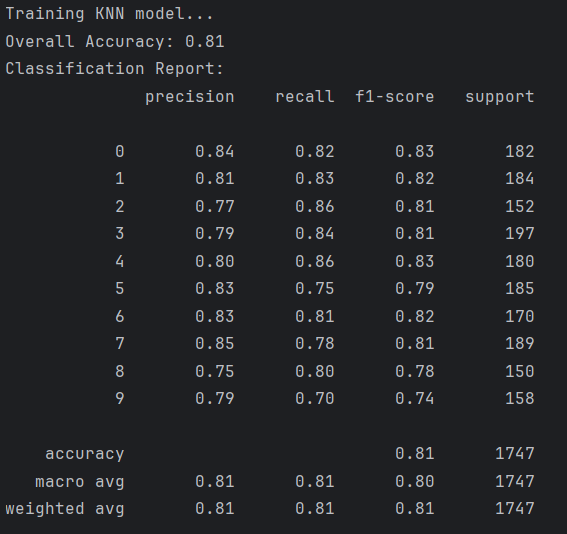
* 1. תחילה מתרחש **חילוץ התכונות** על ידי הפונקציה **extract\_features(self, audio\_path)** .  
     על מנת לטעון את קובץ השמע יש שימוש בספריית librosa ספרייה לעיבוד שמע, ניתוח תכונות קול, חישוב ספקטרוגרמות, חילוץ מאפיינים ומניפולציות על אותות קוליים.

הפונקציה מחשבת את המאפיינים MFCC, שהם תיאור סטטיסטי של התדרים בשמע וכדי לפשט את הייצוג, אנו מחזירים את הממוצע של כל מקדם לאורך הזמן.  
\* מדוע השימוש הוא דווקא ב - MFCC?

כי מדובר בתיאור מספרי קומפקטי המייצג את התוכן התדרי של האודיו, והוא משמש רבות בזיהוי קול ומאוד נוח לשימוש וניתוח.

* 1. לאחר מכן, מתבצע תהליך של **הכנת הנתונים ללמידה** בפונקציה  
     **preprocess\_data(self, X, y)**. כאן מתבצעת חלוקת הדאטה ל – 80% אימון ו – 20% בדיקה. ומאחר ש-MFCCs יכולים להיות במספרים מאוד שונים בטווחים שלהם, מבצעים נרמול (Standardization) כדי להבטיח שהתכונות יהיו בסקאלה דומה.
  2. בשלב זה מתבצע האימון בפונקציה **train(self, X\_train, y\_train)** . כלומר, שומרים את הדוגמאות המסווגות — כי KNN לא באמת "לומד" כמו מודלים אחרים, אלא משווה לדוגמאות קיימות בזמן חיזוי.
  3. לאחר האימון קורה תהליך **החיזוי** בפונקציה **predict(self, X\_test)** בו:
* מחושב המרחק בין כל דגימה לבדיקה לבין כל דגימה באימון.
* נבחרים K השכנים הקרובים ביותר (לפי מרחק אוקלידי).
* מחפשים מהי הקטגוריה הדומיננטית אצל השכנים.
* קובעים את התווית לדגימה החדשה לפי הרוב.
  1. לאחר מכן, מתרחשת הערכת הביצועים שהתקבלו בפונקציה  
     **evaluate(self, X\_test, y\_test)** בה:
* מחשבים דיוק (Accuracy) – אחוז הדוגמאות שסווגו נכון.
* מציגים דו"ח סיווג (Classification Report) שמראה סטטיסטיקות כמו Precision ו-Recall.
* מציגים מטריצת בלבול (Confusion Matrix) – מציגה היכן היו טעויות הסיווג.
  1. לבסוף מתבצעת ויזואליזציה בפונקציה **plot\_results(self, y\_test, y\_pred)** על מנת להמחיש את תוצאותיו של כל מודל.  
     כל דגימה מסווגת מיוצגת בנקודה בגרף. הצבע מציין את התווית האמיתית, ואם סווגה לא נכון היא צבעה יהיה אדום. וכן לבסוף, הקוד מוסיף מקרא (Legend) כדי להראות למה כל צבע שייך.

התוצאות:

* 1. דיוק כללי (Accuracy):  
     accuracy = 0.81 (81%). כלומר, המודל מסווג נכון 81% מהדוגמאות (דיוק טוב יחסית).
  2. דו"ח סיווג (Classification Report) : דו"ח הסיווג מציג דיוק ,(Precision) שליפה (Recall) ו-F1-Score לכל אחת מהקטגוריות.
* **ה - Precision נע בין 75% ל-85%**  
  הקטגוריות החזקות ביותר : מחלקות 0, 7, 4 (דיוק של 84%-85%)  
  הקטגוריות החלשות ביותר: מחלקות 8, 9 (דיוק של 75%-79%)
* **ה - Recall גם יחסית מאוזנת, בין 70% ל-86%**  
  מחלקות עם שליפה גבוהה במיוחד: מחלקות 2,4 (דיוק של 86%)  
   מחלקות עם שליפה נמוכה יותר: מחלקה 9 (דיוק של 70%)
* **ממוצע מאקרו**  
  (Macro Avg): Precision = 0.81, Recall = 0.81, F1-Score = 0.80  
   כל המדדים מעל 80%, שזה סימן טוב למודל מאוזן.
  1. מטריצת בלבול: (Confusion Matrix)

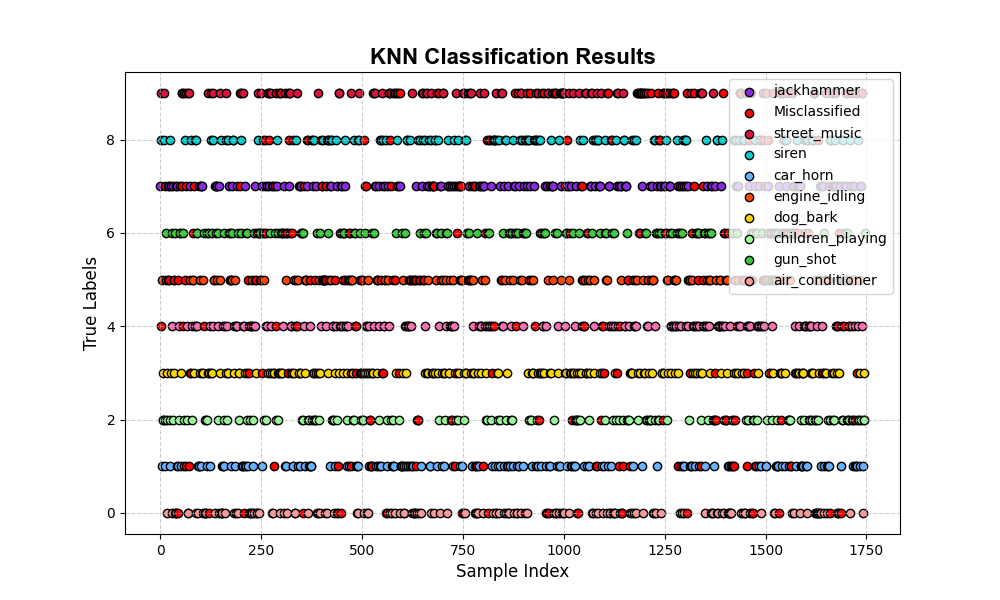
מטריצת הבלבול מראה כמה דוגמאות מכל מחלקה זוהו נכון (האלכסון) לעומת כמה דוגמאות זוהו בטעות כמחלקות אחרות. נשים לב שרוב הערכים הגבוהים נמצאים על האלכסון – כלומר, המודל מסווג נכון את רוב הדוגמאות. אך יחד עם זאת, יש בלבול מסוים בין מחלקות קרובות, למשל:

* מחלקה  **5** זוהתה לפעמים כ **3, 9**.
* מחלקה **9** זוהתה בטעות כ **1, 4**.

תמונה שמכילה טקסט, צילום מסך, גופן, טיפוגרפיה

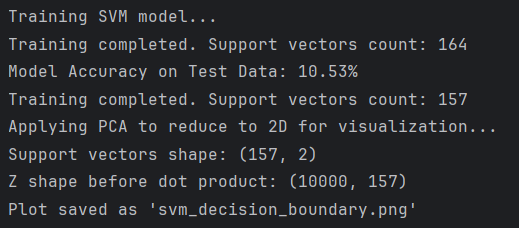
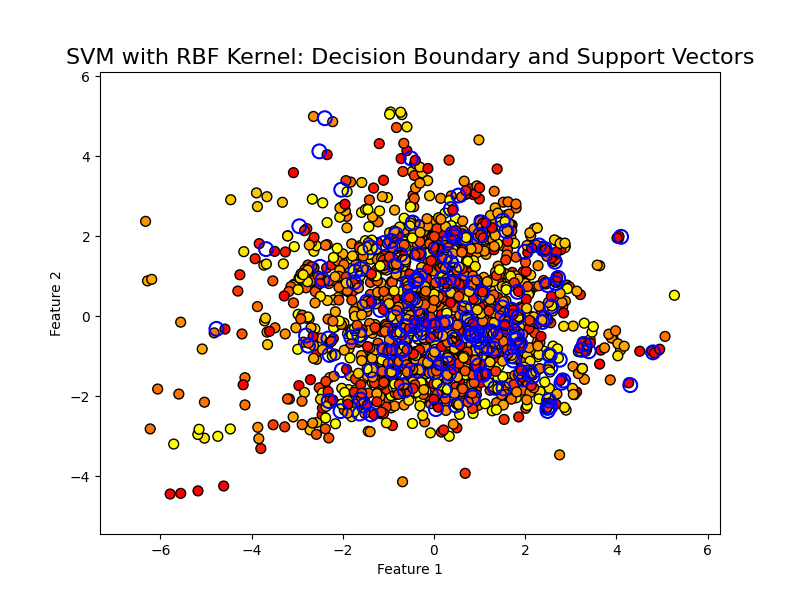
התיאור נוצר באופן אוטומטי

* 1. תוצר ויזואלי:  
     בגרף מוצגים תוצאות הסיווג של המודל עבור דגימות השמע. כל נקודה מייצגת דגימה מסוימת, כאשר הציר האופקי מציין את מספר הדגימה והציר האנכי מציין את התווית האמיתית שלה.

צבעי הנקודות משקפים את הקלאסים השונים, כפי שהוגדרו במודל, כאשר לכל קטגוריה מוקצה צבע ייחודי. נקודות שזוהו נכון מופיעות בצבע המתאים לקלאס שלהן, בעוד שנקודות שסווגו שגוי מסומנות באדום. בנוסף, המקרא (Legend) מציג את שמות הקטגוריות השונות כך שכל תווית מופיעה פעם אחת בלבד, מה שמאפשר קריאה ברורה של הנתונים. מטרת הגרף היא להמחיש את ביצועי המודל ולזהות תבניות אפשריות בטעויות הסיווג.

* 1. מסקנה: התוצאות נראות טובות. דיוק של 81% זה מכובד, במיוחד אם מדובר בנתונים לא פשוטים כמו סאונד. המטריקות מאוזנות בין הקטגוריות, בלי קטגוריה שסובלת מתת-ביצוע חריף. בנוסף אפשר לראות בתוצר הוויזואלי כי יחסית רוב הדאטה סווג נכון.

**מכונות וקטורים תומכים (SVM) :**

* 1. נקודת ההתחלה שהצריכה שיפור משמעותי:  
     כרגע פיתחתי והפעלתי מודל SVM עם קרנל RBF , אך תוצאותיו אינן מספקות. המודל אומן על סט נתונים עם 13 תכונות, אך לצורך ויזואליזציה בוצעה הפחתת ממדים באמצעות PCA לשני ממדים. לאחר האימון, התקבלו 157 וקטורי תמיכה, אך דיוק המודל על סט הבדיקה נמוך מאוד (10.53%), והתמונה המתקבלת אינה מציגה קו החלטה ברור אלא פיזור מעורבב של הנתונים. נדרשת התאמה נוספת של הפרמטרים C ו - gamma, בדיקה אם הנתונים מאוזנים, ובחינת ההשפעה של בחירת תכונות רלוונטיות יותר.
  2. התוצאות כעת :

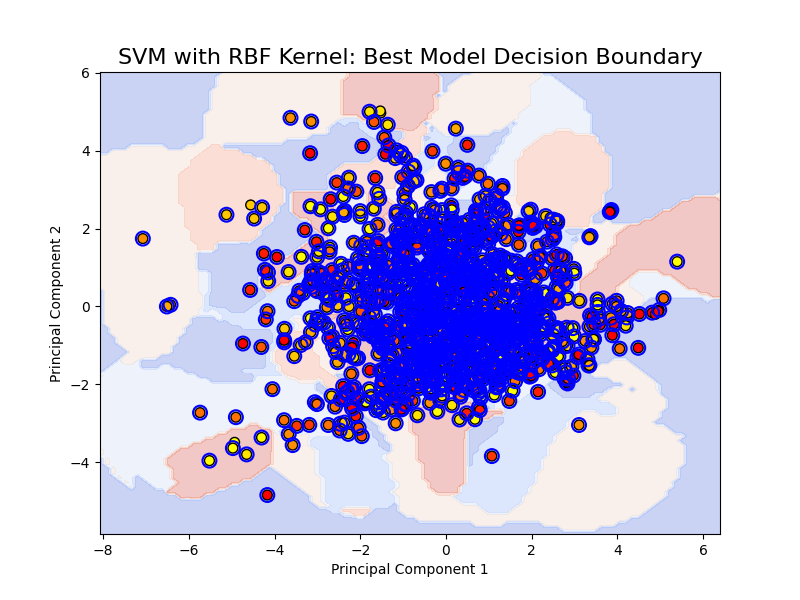
1. שלבי השיפור שהובילו לתוצאות טובות יותר:

* אופטימיזציה של C ו-𝛾 - ביצוע GridSearchCV כדי למצוא את השילוב האופטימלי.
* שימוש באיזון משקלים – הוספת class\_weight='balanced' פתרה בעיות שנגרמו מחוסר איזון בין הקלאסים.
* בחינת תכונות משמעותיות יותר – במקום להשתמש ב-PCA ל-2 ממדים, נבדקו המאפיינים בעלי ההשפעה הגדולה ביותר על ההחלטה.
* שיפור עיבוד הנתונים – נורמליזציה עם StandardScaler הבטיחה שהתכונות יהיו בקנה מידה אחיד.

תמונה שמכילה טקסט, צילום מסך, גופן

תוכן שנוצר על-ידי בינה מלאכותית עשוי להיות שגוי.לאחר ההתאמות, הדיוק השתפר משמעותית, וקו ההחלטה הפך ברור יותר, מה שמעיד על כך   
שה-SVM למד להפריד טוב יותר בין הקלאסים.

1. התוצאות הסופיות:



תוצאות האימון והכוונון של מודל SVM עם קרנל RBF מצביעות על שיפור מסוים בביצועים, אך הדיוק עדיין נמוך מאוד.

* דיוק ראשוני (10.53%): המודל שהוגדר עם הפרמטרים ההתחלתיים נכשל בהפרדת הנתונים בצורה טובה, כפי שניתן לראות מהדיוק הנמוך ומהפיזור הלא ברור של קו ההחלטה.
* שיפור לאחר GridSearch (23.14%): לאחר כוונון הפרמטרים, נמצאו הערכים האופטימליים:  
  C = 10: רגולריזציה חזקה יותר, שמובילה להחלטות חדות יותר.  
  gamma = 0.5: השפעה מתונה יותר של כל דגימה על המרחב, מה שמאזן בין גמישות להכללה.  
  class\_weight = None: ללא איזון אוטומטי בין הקטגוריות, ייתכן שהנתונים אינם מאוזנים, מה שעלול לפגוע בביצועים.

1. מסקנות:

השיפור ב-GridSearch אכן תרם לעלייה בדיוק, אך הערך הסופי (23.14%) עדיין נמוך מאוד, מה שמעיד על צורך בשיפורים נוספים אך מבחינת הפרויקט נעצור כאן.

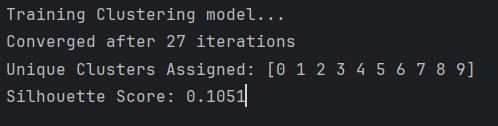
**מקבוץ (Clustering) :**

הקוד:

הקוד מגדיר אלגוריתם קלאסטרינג מותאם אישית שמבצע אשכולות (Clustering) של קבצי אודיו לפי מאפייני MFCC. המטרה היא לקבץ צלילים דומים (כמו אזעקות, קולות רחוב, קולות ילדים וכו'). באופן כללי, קלאסטרינג הוא למידה לא מפוקחת (Unsupervised Learning) שמטרתו לקבץ נקודות דומות יחד. כאן האלגוריתם משתמש בגרסה מותאמת אישית של K-Means, שבה:

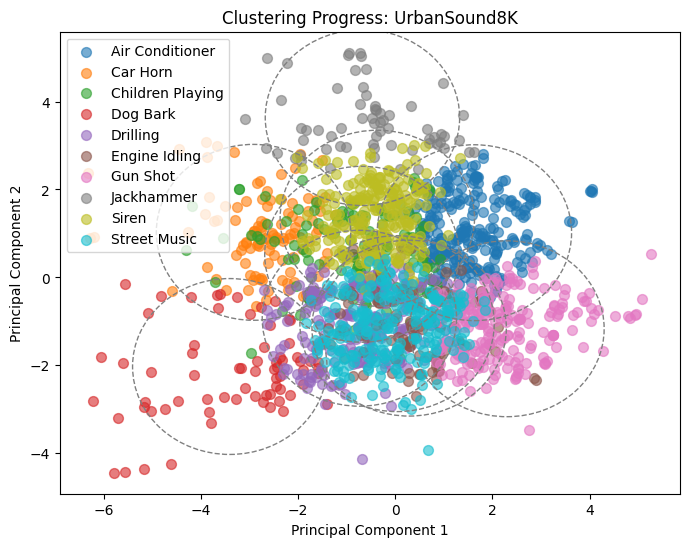
* מאתחלים k צנטרואידים (המרכזים של האשכולות) אקראיים.
* לכל דגימה מוצאים את הצנטרואיד הקרוב ביותר.
* מעדכנים את מיקום הצנטרואידים לפי ממוצע הנקודות ששויכו אליהם.
* חוזרים על התהליך עד שהמרכזים לא משתנים כמעט או שמגיעים למספר איטרציות מקסימלי.

התוצאות:



* 1. ניתוח תוצאות המודל:
* Converged after 27 iterations – האלגוריתם התכנס לאחר 27 איטרציות, כלומר, מצא פתרון יציב יחסית מהר.
* Unique Clusters Assigned: [0 1 2 3 4 5 6 7 8 9] – הנתונים התחלקו ל-10 אשכולות כפי שנדרש, מה שמעיד שהאלגוריתם הצליח לשמר את המבנה הרצוי.
* Silhouette Score: 0.1051 – מדד זה נע בין -1 ל-1.
  + ערכים קרובים ל-1 מעידים על הפרדה ברורה בין אשכולות.
  + ערכים קרובים ל-0 מעידים על חפיפה משמעותית בין האשכולות.

0.1051 הוא ערך נמוך, מה שאומר שהאשכולות עדיין לא נפרדים בצורה טובה, וייתכן שהנתונים חופפים או שהשיטה לא אידיאלית. אצלינו, קיימת חפיפה משמעותית בין הנתונים מה שעלול להיות הסיבה העיקרית לכך (נראה זאת בהמחשה הויזואלית).

2. המחשה ויזואלית:

למרות המסקנה והתוצאה הלא מאוד גבוהה, ניתן לראות באופן ויזואלי שאכן נוצרה הפרדה בין הקטגוריות ואף יש חלוקה יחסית ברורה בין הקטגוריות השונות. כמו כן, אפשר באמת לראות כי קיימת חפיפה לא ריקה בין סוגי הרעשים השונים מה שיכול להיות סיבה טובה לקושי של המודל לייצר הפרדה כמו שצריך.  
יתרה מכך, במהלך כתיבת הקוד היה ניסיון לשפר את התוצאה על ידי שימוש ב – GMM במקום ב – K-Means אך התוצאות הפכו רעות יותר, ולכן, למרות ש – GMM מותאם יותר לאודיו דווקא בחרתי להשתמש במבנה המוכר יותר.  
3. מסקנה:  
המודל התכנס במהירות ושמר על 10 אשכולות, אך האיכות של ההפרדה בין האשכולות עדיין נמוכה ככל הנראה עקב החפיפה הרבה בין נתוני הקטגוריות השונות.

**רגרסיה לינארית (Linear Regression) :**

הקוד:

השתמשתי ברגרסיה ליניארית על מנת לחזות את עוצמת הקול (dB). הרגרסיה הליניארית שימשה כדי לנבא ערך רציף תוך שימוש בתכונות של האודיו, כמו RMS (עוצמת קול ממוצעת) ותדרים.  
1. תהליך המימוש:

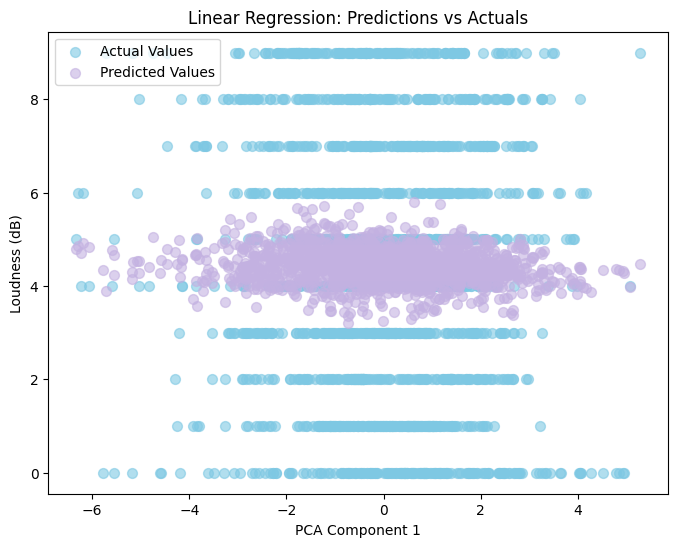
1. תכונות כמנבאות (Features):  
הוצאתי תכונות כמו RMSמתוך קבצי האודיו. RMS הוא מדד שמייצג את עוצמת הצליל הממוצעת, וזו הייתה התכונה העיקרית לשם הניבוי. בנוסף, אפשר היה להשתמש גם בתדרים אחרים, אך בחרתי להתמקד ב-RMS כמדד עיקרי.

2. המודל הליניארי:  
המודל הליניארי השתמש בתכונות האודיו (כמו RMS) כדי לחזות את ערך ה-dB (עוצמת הקול). במהלך שלב האימון, המודל למד את הקשר הליניארי בין התכונות וערך ה-dB, באמצעות חישוב משקלים וביסוס מודל ליניארי.

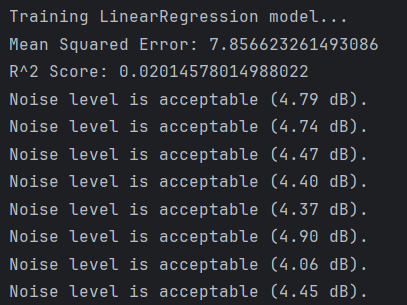
3. הדרישה – ניבוי חריגה מרמת רעש חוקית:  
המטרה הייתה לנבא אם רעש מסוים חורג מהמגבלות החוקיות של רעש בעיר, על פי התקנות שנקבעו בישראל. כלומר, אחרי תחזית עוצמת הקול (dB), יכולתי לבדוק אם הרעש חורג מהגבולות החוקיים (למשל, 55-65 dB במהלך היום או 45-55 dB בלילה).

התוצאות:

* 1. המחשה ויזואלית:

  
  
  
  
  
  
  
  
  
  
  
  
  
  
בגרף שמתי את תחזיות המודל מול הערכים האמיתיים של עוצמת הקול.

* על ציר ה-X יש את הערכים של התכונות שנבחנו, כשכל נקודה מייצגת את עוצמת הקול בפועל.
* על ציר ה-Y יש את תחזיות המודל המקוריות, והגרף מאפשר להשוות בין תחזיות המודל לבין הערכים האמיתיים.
* הנקודות הכחולות מייצגות את רמת הדציבלים שנמדדה בפועל (ground truth).
* הנקודות הסגולות מייצגות את התחזיות של המודל עבור אותם קלטים.

ניתן לראות שבאמת המודל מצליח להפריד בין סוגי הקטגוריות מבחינת רמת ה – db בכך שיש עקביות ורציפות יחסית גבוהה בנקודות הכחולות. בנוסף, אפשר לראות שאכן הנקודות הסגולות, המייצגות את הערכים שנמדדו, צפופות יחסית וקרובות לקווים הכחולים אך כן קשה ליצור באזור זה הפרדה ברורה ומוחלטת לחלוטין.  
  
בנוסף, ניתן לראות בהדפסות את הערכים הבאים:  


* Mean Squared Error (MSE): ערך ה-MSE של 7.86 מציין שישנן טעויות ממוצעות לא קטנות בין התחזיות לערכים האמיתיים של רמות הדציבלים. ככל ש-MSE גבוה יותר, כך המודל לא מצליח לחזות בצורה מדויקת את הערכים.
* R^2 Score: ערך ה-R^2 של 0.02 מציין שהמודל מצליח להסביר רק 2% מהשונות בנתונים, כלומר המודל לא מצליח להתאים היטב לנתונים.
* Noise Level Predictions: כל הערכים שהמודל חזה הם נמוכים מאוד, בסביבות 4-5 dB, שזה בהחלט הרבה מתחת לרמות הרעש החוקיות הרלוונטיות. לפיכך, לפי התחזיות האלו, כל הרמות תקינות ועומדות בתקן של הטווח החוקי עבור רעש יומי (55-65 dB).

3. מסקנה:

המודל אינו מצליח לייצר הפרדה ווודאות מוחלטת בין העוצמות השונות של הנתונים שנבדקים, אך כן בין נתוני המקור. יתרה מכך, ניתן לראות בגרף שרמות הדציבלים שנבדקו עומדות בתקן הנדרש כפי שרצינו לוודא.  
אך יחד עם כל זאת, אכן הפלט מעיד על דיוק נמוך ולא מספק.

**הגברה (Adaboost) :**

הקוד:

יצרתי יישום של אלגוריתם AdaBoost המיועד לסיווג רב-מחלקתי באמצעות SAMME (וריאציה של Adaboost המיועד לסיווג רב מחלקתי) בשילוב עם מסווג KNN חלש.

במסגרת הקוד, אתחול המערכת מתבצע על ידי הגדרת מספר המודלים החלשים (n\_estimators) וכמות השכנים ב – KNN (k).  
במהלך האימון, אני מחשבת משקלים התחלתיים לכל דוגמה ומעדכנת אותם באופן דינמי – דוגמאות שבהן המודל טועה מקבלות משקל גבוה יותר, מה שמדגיש את חשיבותן באיטרציות הבאות. כל מודל חלש מאומן, נמדדת השגיאה המשוקללת שלו, וממנו מחושבת ערך האלפא (α) לפי נוסחת SAMME (הערך הזה נחתך לטווח מסוים כדי למנוע בעיות מספריות).

בתהליך הניבוי, אני מצרפת ניקוד לכל מחלקה על בסיס התוצאות של כל המודלים החלשים, והניבוי הסופי נקבע על ידי המחלקה עם הניקוד הגבוה ביותר.

התוצאות:

תמונה שמכילה טקסט, גופן, צילום מסך, טיפוגרפיה

תוכן שנוצר על-ידי בינה מלאכותית עשוי להיות שגוי.

באמצעות גישה זו, הצלחתי לשפר את דיוק הסיווג מ-**81% ל-83%,** מה שמדגים את היכולת של המערכת להתמודד בצורה יעילה עם דוגמאות קשות ולהגביר את ביצועי הסיווג. זאת כמובן בהתחשב בכך שה – data set כאן הוא קטעי קול, כלומר, יחסית מורכב ועדיין ראינו שיפור.

**תשובות לשאלות**

* 1. המודל שהצליח הכי טוב היה מודל ה – KNN בשילוב עם Adaboost בדיוק של 83%.  
     נשים לב ש – KNN הוא מודל שיודע לעבוד טוב כאשר סוג הדאטה שלנו מחולק באופן כזה שכל קטגוריה "לא מפוזרת מדי" וכל הנקודות אכן קרובות יחסית זו לזו כפי שקורה בדאטה עליו עבדנו. זאת בניגוד לשאר המודלים, אשר עקב החפיפה בנתונים והמורכבות בנתוני קול, הפיקו תוצאות פחות טובות. זאת למרות קרבת הנקודות השייכות לאותה קבוצת סיווג.  
     בנוסף, נשים לב שאין רעש בנתונים. כלומר, אין נתוני סרק מה שהיה יכול מאוד להעיק ולהפריע ל – KNN.  
     כמו כן, השילוב שלו עם Adaboost שמוודא שהמודל מתמקד בדוגמאות שקשות לסיווג ע"י שינוי המשקלים של הנתונים, גורם למודל להתמקד בתיקון "נקודות החולשה" שלו ובכך מעלה את התוצאות ורמת הדיוק.
  2. המודל שהצליח הכי פחות היה מודל ה – SVM עם רמת דיוק של 23.14%. מה שעלולה להיות הסיבה העיקרית לכך היא שקיימת חפיפה רבה מאוד בין סוגי הנתונים השונים. כלומר, גבולות הסיווג אינם ברורים מספיק וזוהי תכונה שאיתה מודל זה מתקשה לעבוד ולספק דיוק גבוהה.  
     בנוסף, משום שהדאטה אותו בחרנו מאוד גדול, SVM איטי יותר לעומת מודלים פשוטים יותר כמו KNN , שכפי שציינו קודם, גם הפיק תוצאות טובות יותר משמעותית.
  3. כן, עבור קלסטרינג המודלים עבדו בצורה חלקה יותר (כמובן כל מודל עם תוצאות שונות כפי שגם הוסבר לעיל על כל אחד באופן מפורט). וכן עבור קבוצות רעשים שהיו יחסית מבודדות, כלומר, לא חפפו באופן חריף עם קבוצות אחרות. אי חפיפה זו סייעה למודל להבחין בין הקטגוריות ולסווג אותם באופן מדויק יותר. ניתן כמובן להבחין בכך גם בהמחשה הוויזואלית של מודל ה – Clustering הנ"ל.  
     כמו כן, תהליך חיזוי הערך אותו ראינו במודל של הרגרסיה הלינארית היה קשה יותר לעומת סיווג הנתונים. זאת משום החפיפה הנ"ל וכן עקב כך שהדאטה מכיל מספר גדול מאוד של דוגמאות ונתונים.
  4. היתרונות והחסרונות של כל מודל:

**KNN (K-Nearest Neighbors)**

יתרונות:

* פשטות: קל להבנה וליישום, ללא צורך במודל מובהק מראש.
* אדפטיביות: מתפקד היטב עם נתונים בלתי ליניאריים ומורכבים.

חסרונות:

* ביצועים עם נתונים גדולים: איטי יחסית עבור קבוצות נתונים גדולות, כיוון שהוא מצריך חיפוש עבור כל נקודה.
* רגישות לרעש: יכול להיות מושפע מאוד מנתונים רעשיים או לא מאוזנים.

**SVM (Support Vector Machine)**

יתרונות:

* ביצועים גבוהים: יכול להתמודד היטב עם בעיות קווניות לא ליניאריות באמצעות גרעינים שונים.
* מקסימיזציה של מרווח: ממקסם את המרווח בין הקטגוריות, דבר שמוביל לתוצאות טובות יותר במקרים רבים.

חסרונות:

* דרישות חישוביות: עלול להיות איטי מאוד עם קבוצות נתונים גדולות, במיוחד במצבים עם מספר גבוה של תכונות.
* רגישות לבחירת הגרעין: בחירת גרעין לא מתאימה עלולה להוביל לתוצאות לא טובות.

**Clustering (כמו K-Means)**

יתרונות:

* מיפוי נתונים בלתי מפוקחים: מאפשר להבין את מבנה הנתונים מבלי לדעת מראש את הקטגוריות.
* יישומים מגוונים: מתאים למגוון רחב של בעיות, כמו דיפרנציאציה בין קבוצות.

חסרונות:

* בחירת מספר הקלאסטרים: בחירת מספר הקלאסטרים הנכון יכולה להיות קשה ומאתגרת.
* רגישות לחפיפות: במקרה של קבוצות חופפות, התוצאות יכולות להיות לא מדויקות.

**Linear Regression**

יתרונות:

* פשטות וקלות הבנה: קל ליישום ולפרש את התוצאות.
* ביצועים טובים במקרים ליניאריים: עובד טוב כאשר הקשרים בין התכונות הם ליניאריים.

חסרונות:

* מגבלות על הקשרים: לא מתאים לבעיות עם קשרים לא ליניאריים.
* רגישות לרעש: יכול להיות מושפע מרעש בנתונים, מה שמוביל לתוצאות לא מדויקות.

**AdaBoost (Adaptive Boosting)**

יתרונות:

* שיפור ביצועים: משפר את הביצועים של מודלים חלשים על ידי שילובם.
* התמודדות עם דינמיות: מתמודד היטב עם בעיות של חוסר איזון בין הקטגוריות.

חסרונות:

* רגישות לרעש: עלול להתרכז ברעש ולהתאמן על דוגמאות לא מייצגות.
* דרישות חישוביות: עשוי להיות כבד בחישובים, במיוחד עם מספר גדול של דוגמאות או תכונות.
  1. ב – UrbanSound8K לא כל סוגי רעש בעלי כמות דוגמאות זהה. מכאן, שיש קטגוריות המכילות יותר או פחות דוגמאות מהשאר. ההשפעה על המודלים היא בכך שככל שיש למודל יותר דוגמאות על קטגוריה מסוימת (הוא שכיח יותר), כך יש לו יותר מידע עליה והוא ידע לסווג אליה מידע בצורה מדויקת ונכונה יותר.   
     קל לראות זאת בהמחשה הוויזואלית של מודל ה – Clustering בו כל קטגוריה בעלת צבע שונה וניתן ממש לראות את הנקודות בקטגוריות השונות מסווגות לעיגולים הרלוונטיים. ככל שיש יותר נקודות מצבע מסוים כך ניתן לראות שהעיגול הממרכז אותן מכיל הרבה יותר מהן.