

פרוייקט גמר

מגמת הנדסת תוכנה

מכללת הרב תחומי ב'

נושא הפרוייקט: מערכת לזיהוי מוזיקה מתשדירי רדיו

שם: אביב בכור (214201600)

שם מנחה: פליקס גרייצר

תאריך הגשה: 29.5.2023 שנה"ל תשפ"ג

תוכן עניינים

[1. הצעת פרוייקט 4](#_Toc134795440)

[2. תקציר 6](#_Toc134795441)

[2.1 רקע לפרוייקט 6](#_Toc134795442)

[2.2 תהליך המחקר 7](#_Toc134795443)

[2.3 סקירה ספרותית 8](#_Toc134795444)

[2.4 אתגרים מרכזיים 9](#_Toc134795445)

[2.4.1 הבעיה איתה התמודד התלמיד 9](#_Toc134795446)

[2.4.2 הסיבות לבחירת הנושא 9](#_Toc134795447)

[2.4.3 מוטיבציה לעבודה 9](#_Toc134795448)

[2.4.4 הצורך עליו עונה הפרוייקט 9](#_Toc134795449)

[2.4.5 הצגת פתרון לבעיה 9](#_Toc134795450)

[3. מטרות ויעדים 10](#_Toc134795451)

[4. אתגרים 10](#_Toc134795452)

[5. מדדי הצלחה למערכת 10](#_Toc134795453)

[6. רקע תאוריטי 11](#_Toc134795454)

[7. תיאור המצב הקיים 23](#_Toc134795455)

[8. ניתוח חלופות מערכתי 23](#_Toc134795456)

[9. תיאור החלופה הנבחרת 23](#_Toc134795457)

[10. אפיון המערכת שהוגדרה 24](#_Toc134795458)

[10.1 ניתוח דרישות המערכת: 24](#_Toc134795459)

[10.2 מודול המערכת: 24](#_Toc134795460)

[10.3 אפיון פונקציונלי: 24](#_Toc134795461)

[10.4 ביצועים עיקריים: 26](#_Toc134795462)

[10.5 אילוצים: 27](#_Toc134795463)

[11. תיאור הארכיטקטורה 27](#_Toc134795464)

[11.1 הסבר בלוקים מפורט: 28](#_Toc134795465)

[11.4 ארכיטקטורת רשת: 29](#_Toc134795466)

[13. למידת מכונה 29](#_Toc134795467)

[13.1 תהליך איסוף הנתונים: 29](#_Toc134795468)

[13.2 סוג הלמידה שנבחר: 29](#_Toc134795469)

[13.3 נוסחאות: 30](#_Toc134795470)

[14. ניתוח ותרשים UML 31](#_Toc134795471)

[14.1 אלגוריתמים ראשיים: 31](#_Toc134795472)

[14.2 ניתוח תרשים וUse Cases-: 32](#_Toc134795473)

[14.3 מבני נתונים: 33](#_Toc134795474)

[תרשים UML: 33](#_Toc134795475)

[15. רכיבי ממשק 34](#_Toc134795476)

[16. תיכון המערכת 34](#_Toc134795477)

[ארכיטורת המערכת: 34](#_Toc134795478)

[17. תיאור התוכנה 34](#_Toc134795479)

[17.1 סביבות עבודה: 34](#_Toc134795480)

[17.2 שפות תכנות: 34](#_Toc134795481)

[18. תיאור מסכים\ממשק משתמש 35](#_Toc134795482)

[19. קוד התוכנית 35](#_Toc134795483)

[20. מדריך למשתמש 41](#_Toc134795484)

[21. בדיקות והערכה 41](#_Toc134795485)

[22. מסקנות 42](#_Toc134795486)

[23. פיתוחים עתידיים 42](#_Toc134795487)

[24. ביבליוגרפיה 42](#_Toc134795488)

# 1. הצעת פרוייקט

הצעת פרוייקט – "פרסוף"

אביב בכור (214201600)

מכללת הרב תחומי ב'

הפרוייקט שלי נעשה יחד עם הצבא והמטרה שלו היא להכיר אותי עם העולם של עיבוד אותות דיגיטליים. כדי לעשות זאת הוחלט שהדרך הטובה ביותר היא לבצע מערכת התקלוט אותות מתשדיר רדיו, תעבד אותם ולאחר מכן תסיק עליהם איזו שהיא מסקנה. המערכת שהחלטנו לבצע היא מערכת התקלוט אותות מתחנות רדיו ותידע את המשתמש האם ברדיו משודר שיר או לא (למשל משודרת פרסומת או תוכנית) כדי לאפשר למשתמשים להימנע מהקשבה לפרסומות או תוכניות ולאפשר להם להקשיב רק למוזיקה.

תיאור הבעיה האלגוריתמית:

ישנן שני בעיות אלגוריתמיות עיקריות במערכת. הראשונה, קליטה ועיבוד אותות דיגיטליים (DSP). שרשרת העיבוד הדיגיטלית כוללת קליטה של אות רדיו FM והוצאת המידע המתאים ממנו. הבעיה השניה היא זיהוי וקטלוג האות – האם הוא מוזיקה או לא – הקיטלוג יכול להיות כללי (האם האות הוא מוזיקה או לא) או יותר ספציפי, חלוקה לז'אנרות.

רקע תיאורטי:

כמו שישנן שתי בעיות אלגוריתמיות שונות ישנם שני תחומים עיקריים הדורשים ידע תיאורטי בפרוייקת שלי; עיבוד אותות ולמידת מכונה.

עיבוד אותות:

שלב עיבוד האותות מחולק לשני שלבים, שרשרת הקליטה (rf chain) ושרשרת העיבוד (DSP chain).

שרשרת הקליטה:

תהליך קליטת האותות הוא בעיקר תהליך אלקטרוני המתרגם גלי רדיו לאותות חשמליים במתחים שונים המתורגמים לאפס ואחד ונקראים כביטים. תהליך זה מאפשר לנו לתרגם גלי רדיו לתוכן בינארי שאפשר לעבד במחשב. לצורך הפרוייקט שלי אין לי צורך לעבוד על תהליך התרגום עצמו אלא רק להשתמש במודול המאפר לי לקלוט גלי רדיו ולתרגם אותו ללא צורך לעבוד ברמה הפיזית. הקליטה נעשית כאוסף של דגימות IQ, כלומר אוסף של דגימות המורכבות כאשר I מהווה את החלק הממשי וQ מהווה את החלק המדומה. את הדגימות האלו ניתן לתרגם מאוד בקלות מערך מדומה לערך ממשי שניתן לשדר\לקלוט על ידי מציאת פאזה ואמפליטודה. דגימות IQ הן דרך טובה ופשוט לדגום אות מבלי לאבד הרבה מידע.

למידת מכונה:

בשביל להבדיל בין מוזיקה ללא מוזיקה בחרתי להשתמש במודל LOF (Local Outliar Factor) שנועד לאלגוריתמים של מציאת חריגים (anomaly detection) אך להשתמש בו בתור אלגוריתם המאפשר לי גם למצוא האם נקודה חדשה היא חריגה (novelty detection).

הליכים עיקריים בפתרון הבעיה:

1. איסוף מידע: מציאת מאגר מידע גדול ואמין ממנו אפשר לשלוף דגימות של מוזיקה

2. עיבוד אותות: יצירת מודולים המאפשרים קליטת אות רדיו והוצאת אות קול.

3. לימוד המכונה: שימוש במידע כדי ללמד את המכונה לזהות את התכונות של מוזיקה וכך גם לדעת מה הוא לא מוזיקה.

הליכים עיקריים בלמידת מכונה:

1. וקטורזציה של מידע: שליפת תכונות האודיו עבור כל אחד מהקבצים במאגר המידע ולהפוך אותם לוקטורים.

2. לימוד המודל (novelty detection using clustering): יצירת המודל (או שימוש בספריה) וללמד אותו לזהות מוזיקה בעזרת שיטת clustering. כלומר, יצירת קבוצה המתארת את התכונות של מוזיקה ועבור כל ערך חדש (novelty) לבדוק האם הוא שייך לקבוצה או לא.

תיאור טכנולוגית ההנדסה:

המערכת שלי תחולק לשלוה מודולים עיקריים: מודול ה-DSP, מודל ה-LOF והלולאה המרכזית אשר עוברת על כל ספקטרום ה-FM (88khz-108khz) וקולטת את התחנות.

שלבי המערכת יהיו:

קליטת תחנות --> עיבוד האותות (DSP) --> קטלוג בעזרת המודל (מוזיקה או לא) --> תצוגה למשתמש

פיתוחים עתידיים:

1. הוספת בחירה של תחנה ספציפית ולא כל הספקטרום

2. זיהוי ז'אנרות

לוחות זמנים:

אוגוסט: למידה תיאורתית על עיבוד אותות

ספטמבר: עבודה על שרשרת DSP

אוקטובר: למידה תיאורתית על למידת מכונה וfeature extraction לגלי קול

נובמבר: עבודה על למידת המכונה

דצמבר: מעבר אפקטיבי על כל ספקטרום הFM

סוף דצמבר – ינואר: חיבור כל המודולים יחד למערכת אחת

# 2. תקציר

הפרוייקט הינו תוכנה היודעת לקבל ולעבד אותות רדיו בספקטרום ה-FM ולדעת לזהות האם התוכן המשודר בכל תכנה הינו מוזיקה או משהו אחר (פרסומת, תשדיר חדשות וכולי).

מערכת זו מורכבת ממספר חלקים:

1. קליטת אותות
2. עיבוד אותות
3. חיזוי על ידי מכונה
4. קליטת אותות:

בעזרת ממשק של רדיו USRP ניתן לבצע קליטת דגימות הנשמרות לתוך זיכרון ה-RAM של המחשב ומוכנות לעיבוד.

1. עיבוד אותות:

המערכת משתמשת במספר פעולות המאפשרות לבצע מעבר מאפנון FM לאות קול שבני אדם יכולים לשמוע ולהבין.

1. חיזוי על ידי מכונה:

המערכת מוציאה את המידע המתאים מאותות הקול, הופכת אותם לוקטור ומעבירה אותו דרך מודל המנסה לחזות האם מדובר המוזיקה או משהו אחר.

מפרט טכני:  
שם המכשיר: Ettus USRP x310

הדרישות מהמבצע:

* תכנון ומימוש המודולים בשפת Python
* מודלי ניתוח ועיבוד אותות
* מודלי למידת מכונה
* התממשקות למכשיר ה-USRP

## 2.1 רקע לפרוייקט

אחת הבעיות המרגיזות יותר בהאזנה לרדיו היא שאין דרך לדעת היכן ניתן להקשיב למוזיקה והיכן נמצאות החדשות. כיום, כדי למצוא תחנה עם מוזיקה להקשיב לה בזמן הנהיגה יש צורך לזפזף בין תחנות עד שמוצאים את תחנה עם מוזיקה ולאחר כמה שירים, כהתשדיר מסתיים ומתחילות הפרסומות, יש צורך לעשות זאת שוב וחוזר חלילה.

ישנו צורך במערכת היכולה להציג למשתמשים באילו תחנות משודרת מוזיקה ובאילו תשדירי חדשות ופרסומות.

המערכת שלי תיתן מענה בזמן אמת לצורך זה על ידי קליטת ספקטרום ה-FM וחיזוי על ידי מכונה באילו מן התחנות ישנה מוזיקה משודרת.

## 2.2 תהליך המחקר

תהליך המחקר כלל ארבעה חלקים עיקריים:

1. מחקר תיאורתי על אותות רדיו
2. מחקר על ממשק ה-USRP (UHD)
3. מחקר על תכונות של אותות אודיו
4. מחקר על למידת מכונה

תחילה, הייתי צריך לחקור על אותות. הייתי צריך ללמוד על איך אות מתנהג, אילו תכונות יש לו, אילו נתונים לגביו חשובים לי לדעת וכדומה. בנוסף, הייתי צריך ללמוד על דרכים שונות לדגום אותות, בעיקר דגימות IQ. לאחר מכן התחלתי ללמוד את שרשרת ה-DSP.

לאחר קריאה מעמיקה על הנושא הבנתי שהדבר הראשון שיש עלי ללמוד הוא התרמת פורייה, שמאפשרת לי לפרק אות מורכב לרכיבים המחזוריים שלו – תדרים. למדתי על שני דרכים שונות לממש את ההתמרה – DFT ו-FFT ועל ההבדלים ביניהן ולבסוף החלטתי להשתמש ב-FFT משום הוא יעיל יותר – DFT הוא ביעילות של לעומת FFT שהוא ביעילות של . לאחר מכן המשכתי לקרוא על שרשרת ה-DSP ולמדתי שהשלב הבא הוא להפעיל פילטר דיגיטלי על האות, כדי לסנן רעש שנוצר מתחנות ואותות שונים אשר עולים אחד על השני. למדתי על סוגים שונים של פילטרים (או חלונות) ולבסוף בחרתי להשתמש בפילטר הנעזר בחלון המינג. את החלון אפשר להפעיל האחד משני דרכים; כפל במימד התדר או קונבולוציה במימד הזמן. לאחר הפעלת הפילטר למדתי כי יש צורך בביצוע דימודולציה, תהליך הורדת האפנון מאות הרדיו – הפעולה ההופכית למודולציה. לאחר הדימודולציה יש צורך לבצע דסימציה, הורדת התדר לאחד שבני אדם יכולים לשמוע, שכן אנחנו לא יכולים לשמוע אותות רדיו בתדרים גבוהים מאוד.

השלב השני היה למידה על ממשק ה-USRP. למדתי שקיים API לשפות התכנות Pythong ו- C++ הנקרא UHD המאפשר לבצע דרכו קליטה של אותות רדיו.

בשלב השלישי הייתי צריך ללמוד לעומק אותות אודיו ותכונות שלהם המאפשרים לי להבדיל בין דיבור למוזיקה. למדתי על תוכנות בשלושה רבדים שונים; זמן, תדר ותדר-זמן ועל איך כל אחד מהם משפיע על יכולות הזיהוי וההבדלה בין מוזיקה לדיבור.

ולבסוף למדתי על למידת מכונה. למדתי על סוגים שונים של מודלים ועל שימושים שונים שלהם. הבנתי שאני צריך להשתמש במודל מסוג Anomaly Detection ולמדתי על סוגים שונים של מודלים כאלו ועל ההבדלים ביניהם.

## 2.3 סקירה ספרותית

DFT – Discrete Fourie Transform

FFT – Fast Fourie Transform

DSP – Digital Signal Processing

FM – Frequency Modulation

API – Application Interface

USRP – Universal Software Radio Peripheral

SDR – Software Defined Radio

Hz (khz,mhz…) – (kilo, mega) Hertz

OCC – One Class Classification

SVM \ OCSVM – Support Vector Machine \ One Class Support Vector Machine

Db – Decibel. The ratio of two values of a power or root-power quantity on a logarithmic scale.

ספקטרום – סְקָלָה דו-מימדית שמבטאת הספק עבור תדר\אורך גל.

## 2.4 אתגרים מרכזיים

### 2.4.1 הבעיה איתה התמודד התלמיד

בכתיבת הפרוייקט היו שני סוגים עיקריים של בעיות, בעיות אלגוריתיות ובעיות לימודיות.

בעיות לימודיות:

הפרוייקט מתעסק בנושאים שלא היה לי ידע קודם בהם, אותות. היה לי בעיה בתחילת הדרך להבין את כל העולם של אותות אשר כלל הרבה פיזיקה ומתמטיקה ולתרגם אותו לעולם המחשבים.

בנוסף, לא היה לי הרבה ידע גם בתחום למידת מכונה. תחום למידת המכונה הוא מורכב מאוד וכולל הרבה תחומים מתמטיים מורכבים שלא היה לי ידע נרחב אליהם לפני כן.

בעיות אלגוריתמיות:

בפרוייקט היו שתי בעיות עיקריות שהייתי צריך לעבוד עליהם.

1. תרגום העולם המתמטתי של איבוד העותות לאלגוריתם שאני אוכל לכתוב.
2. לאמן מודל למידת מכונה שיוכל לזהות מוזיקה.

### 2.4.2 הסיבות לבחירת הנושא

הסיבה שבחרתי לעבוד על הנושא הזה היא משום שהתחומים שפרוייקט זה מלמד ועוסק בהם הם אותם נושאים שבהם אני אעסוק בזמן השירות שלי בצבא ובכך, תוך כדי העבודה על פרוייקט זה אני למדתי את התחומים שיידרשו לי במהלך השירות. בנוסף, הבעיה של עודף פרסומות בהאזנה לרדיו, בעיקר בזמן נסיעה, מפריעה לי מאוד ורציתי לעשות משהו בנוגע לזה.

### 2.4.3 מוטיבציה לעבודה

המוטיבציה שלי לעבודה היא ללמוד נושאים חדשים ומעניינים שיעזרו לי במהלך השירות הצבאי.

### 2.4.4 הצורך עליו עונה הפרוייקט

ישנו צורך למערכת היכולה להבדיק בין מוזיקה לפרסומות\חדשות כדי לעזור לאנשים לבחור את התחנות הרצויות עליהם ללא צורך בזיפזוף.

### 2.4.5 הצגת פתרון לבעיה

בעזרת טכנולוגיית למידת מכונה ותכונות פיזיולוגיות של גלי קול ניתן לייצר מודל היוכל להבדיל בין מוזיקה לדיבור. לאחר מכן, מתבצעת קליטה של כל תחנות הרדיו בספקטרום ה-FM והתשדיר המשודר בהן מועבר דרך המודל לצורכי חיזוי.

## 3. מטרות ויעדים

מטרות:

* לקלוט ולעבד אותות באיפנון FM.
* סריקת ספקטרום ה-FM ומציאת תחנות.
* זיהוי אותות האודיו.

יעדים:

* מודול המאפשרת תהליך DSP מלא עבור תחנות.
* מודל למידת מכונה שיוכל להבדיל בין מוזיקה לדיבור.
* מערכת התחבר בין תהליך ה-DSP למודל ותציג תוצאה למשתמש.

## 4. אתגרים

כדי לייצר מערכת שתוכל להבדיל בין מוזיקה לדיבור תחילה היה צורך לחקור על תהליך ה-DSP באופן כללי ועל איפנון FM באופן ספציפי. חקר זה דרש העמקה בתכונות של אותות ובהתנהגותם. כמו כן היה צורך בלמידה על אלגוריתמים שכיחים בתחום כגון FFT וסינון אותות.הלמידה הייתה מאתגרת משום שאלו תחומים שלא נחשפתי אליהם לפני כן.

אתגר נוסף היה לתרגם את עולם המתמטיקה והפיזיקה שאותו למדתי לעולם התוכנה, שכן לא כל אלגוריתם מתמטי ניתן לתרגום ישיר בשל מגבלות מערכת. אחד האתגרים העיקריים שהיו לי היה לממש FFT שכן האלגוריתם הינו רקורסיבי ודורש נתונים באורך שהוא חזקה של 2, אך לא כל קליטה נתנה לי סט של נתונים שתואם.

לצורך קליטת האותות עצמם היה לי צורך ללמוד את ה-API של מכשיר ה-SDR שנקרא UHD. ל-API יש תמיכה בשני שפות; Python ו-C++ אך הדוקומנטציה ל-Python, השפה שבה אני עושה את הפרוייקט, לא מפורטת מספיק ולכן תהליך הלמידה על ה-API היה גם כן אתגר.

ולבסוף הייתי צריך לעבוד על מודל למידת המכונה בו היו שני אתגרים עיקריים. הראשון, למצוא איזה סוג מודל אני צריך. לאחר למידה על מגוון רחב של סוגים שונים של מודלים הבנתי שאני צריך מודל מסוג novelty detection, המאפשר לזהות נקודת מידע חדשה ולדעת האם היא שייכת לקבוצה מסויימת או לא, אך בחירת מודל ספציפי היה אתגר. האתגר השני היה למצוא איזה מידע מאותות האודיו אני צריך לצורכי לימוד המכונה. לאחר מחקר רב, וכמה טעויות, הצלחתי למצוא את המידע הנדרש.

## 5. מדדי הצלחה למערכת

* מעבר נקי על כל ספקטרום ה-FM.
* זיהוי מספק (70% ומעלה) של מוזיקה.
* ביצוע דימודולציה והשמעת אודיו מקטע נקלט.

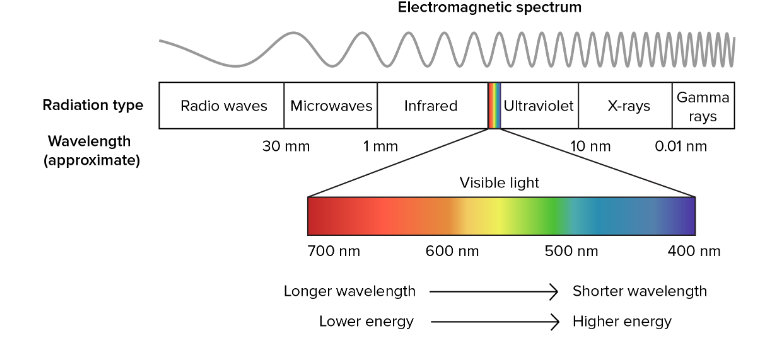
## 6. רקע תאוריטי

תקשורת אלחוטית:

תקשורת אלחוטית(Wireless Communication/"Over-the-Air" Communication) הוא מונח  
כללי להעברת מידע בין שתי נקודות בלי צורך במוליך חשמלי שמחבר ביניהם. כאשר רוצים לשדר מידע יש צורך בגל נושא חשמלי אשר "מרכיבים" עליו גל מידע כלשהו בתהליך הנקרא איפנון (Modulation). האות המאופנן מוגבר ומועבר לאנטנה אשר הופכת את האות החשמלי לאות אלקטרו-מגנטי. אות זה מתפשט (Propagates) דרך תווך, בדרך כלל אווירי, עד שהוא מגיע לאנטנת הרכיב אליה הוא מיועד. האות מגיע בעיכוב (Delay) ובהפחתה. ככל שהגל מתרחק מאנטנת המקור עוצמתו נחלשת. האנטנה שקולטת את האות הופכת אותו לאות חשמלי חלש מאוד. לאחר הקליטה האות עובר דרך שרשרת RF הכוללת מסננים המסננים אותות לא רצויים, מגברים אשר מגבירים את האות החלש לעוצמה שניתן לעבוד איתה וגלאים אשר מבצעים דימודולציה (פעולה הפוכה לאיפנון) המאפשרת לחלץ את גל המידע מהגל הנושא.

את הקרינה האלקטרו-מגנטית נהוג לחלק לתחומים; בעיקר סוגי קרינה, שמתוכם נמצאים גם גלי הרדיו. את החלוקה הזאת מבצעים לפי תדר (Frequancy) או אורך גל (Wavelength). תדר ואורך גל קשורים ביחס הפוך המיוצג על ידי הנוסחה כאשר: v=מהירות, f=תדר λ= אורך גל. בפרוייקט זה אתרכז בספקטרום ה-FM הנמצא בין mhz 88-108 .

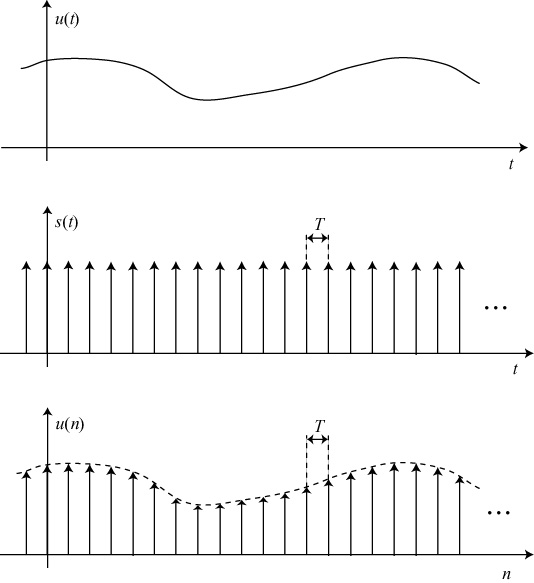
חלוקה זו מתבטאת בספקטרום האלקטרו מגנטי:

[[1]](#footnote-1)

(האור הנראה בעין האנושית הוא גם תחום ספציפי של תדרי גלים אלקטרו-מגנטיים.)

Quadrature Sampling:

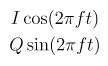
כאשר רוצים לקחת אות רציף ולהפוך אותו לאות דיגיטלי בדיד שהמחשב יכול לעבוד איתו צריך לדגום אותו. דגימת אות היא בעצם קליטת האות במרווחים אחידים אשר יתנו את הצורה הכללית של האות הרציף, ככל שיש יותר דגימות האות מדוייק יותר אך תופס יותר מקום בזיכרון.

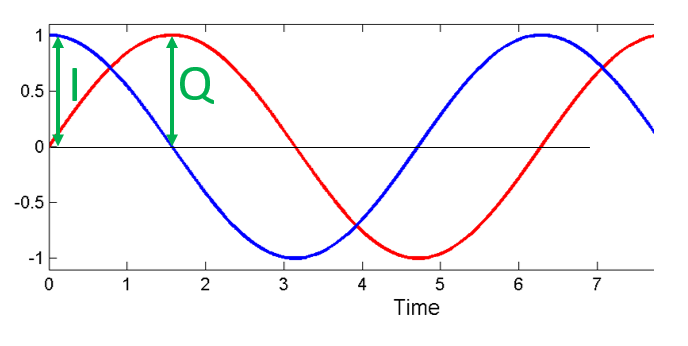
[[2]](#footnote-2)

(למעלה – האות הרציף, למטה – האות לאחר הדגימה, באמצע – האות הדוגם)

ישנן שיטות דגימה רבות אך שיטת הדגימה שאני השתמשתי בה נקראת Quadrature Sampling.

Quadrature Sampling, או בשם אחר דגימת IQ, זה שיטת דגימה המשתמשת בשני אותות, sin ו-cos הנמצאים מחוץ לפאזה ב-90 מעלות אחד מהשני, כאשר איבר ה-cos נמצא בתוך הפאזה (in phase) ולכן מיוצג על ידי האות I, ואיבר ה-sin נמצא בפאזה של 90 מעלות ממנו (quadrature) ולכן מיוצג על ידי האות Q. בעזרת הזהות הטריגונומטרית ניתן לשדר כל אות סינוס עם פאזה של אפס על ידי סכום של אות סינוס ואות קוסינוס. ()

[[3]](#footnote-3)

[[4]](#footnote-4)

כך שניתן לכתוב כל אות שאנחנו רוצים בצורה הבאה:

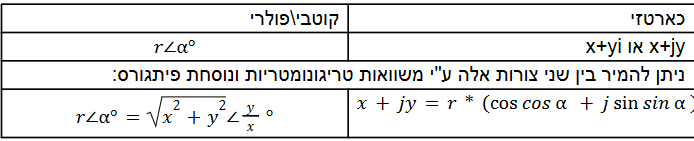
[[5]](#footnote-5)

כאשר x(t) הוא האות הרצוי, f הוא התדירות ו-t הוא הזמן.

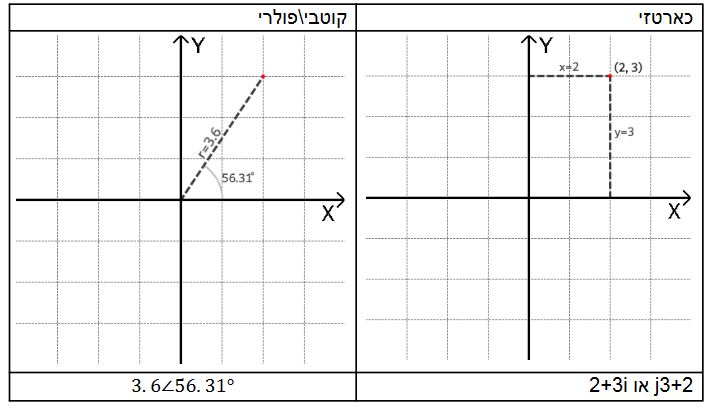
בזכות הזהות הזאת ניתן לייצר איזו אמפליטודה ופאזה שאנחנו רוצים רק על ידי שינוי ערכי ה-I וה-Q.

בעצם, שיטת ה-IQ מאפשרת לנו לייצג את הפאזה והאמפליטודה בצורה שונה, מספרים מרוכבים.

מספרים מורכבים:  
מספר מרוכב הוא מספר שמייצג ערך בעל רכיב ממשי ורכיב מדומה. למעשה מספר על מערכת צירים דו-מימדית, בשונה ממספרים ממשיים, שמיוצגים רק על מישור אחד (ממינוס אינסוף או אפס עד פלוס אינסוף). מספרים ממשים ניתן להציג שני צורות; כארטזית (קורדינטות – x ממשי ו-y מדומה) ופולארי\קוטבי (α – זווית\כיוון, r – רדיוס\מרחק)

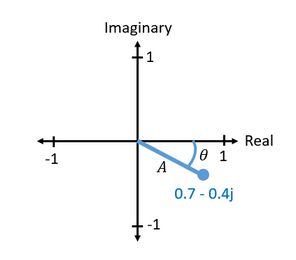


נניח ואנחנו רוצים לייצג את המספר המורכב (x=2, y=3), כך הוא יוצג בשני הצורות השונות:

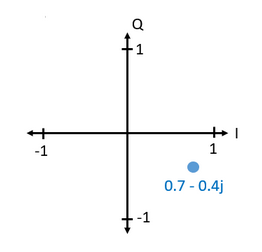
[[6]](#footnote-6)

כשקולטים או משדרים גל מאחסנים אותו בזיכרון כשני מספרים (שיטה כרטזית), אולם אין דרך לשדר ערכים מדומים ולכן יש צורך למהיר אותו לערך ממשי.

הזווית (α) והרדיוס (r) מהווים את הפאזה () ואת האמפליטודה (A) של הגל שלנו.

[[7]](#footnote-7)

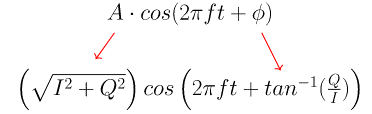
כמו שנאמר לפני כן ניתן להמיר מהייצוג הכרטזי לייצוג הקוטבי על ידי זהויות טריגונומטריות ומפשט פיטגורס. אם נתייחס לציר הממשי כערך ה-I ולציר המדומה כערך ה-Q ניתן להמיר ערכי I ו-Q לפאזה ואמפליטודה ובכך להמיר את הערכים מצורה שקל לשמור בזיכרון של המחשב (שיטה כרטזית) לצורה שניתן לשדר לאוויר.

 [[8]](#footnote-8)

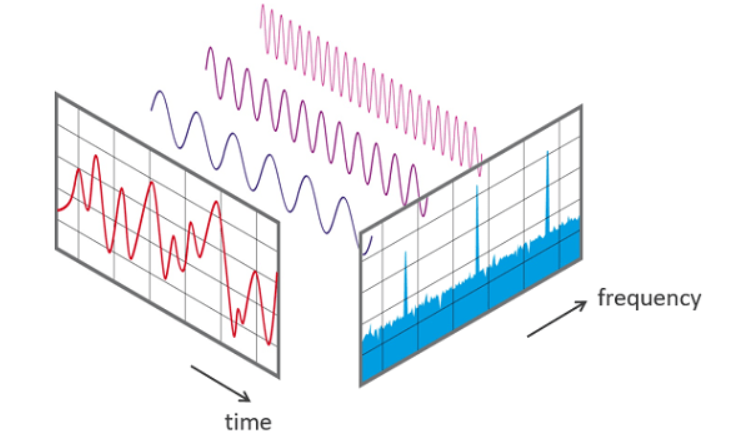


[[9]](#footnote-9)

5

[[10]](#footnote-10)

התמרת פורייה:

[[11]](#footnote-11)

פונקציה חשובה המשומשת בתהליך של עיבוד אותות היא fourier transform המאפשרת להפריד אות הנמצא במימד הזמן לתצוגה במימד התדר, דבר שמאפשר לנו לראות אילו תדירויות הן התדירויות העיקריות באות ובכך למצוא את האותות המרכיבים אותו. FFT (Fast Fourier Transform) או DFT (Discrete Fourier Transform) מאפשרים לנו לאתר תחנות בקלות (איזורים בגרף עם ריכוז גבוהה של תדרים) ומאפשר לנו לראות ולהבין אות בצורה יותר טובה מאשר אם הוא היה במרחב הזמן.

[[12]](#footnote-12)

כאשר N הוא כמות הדגימות באות במרחב הזמן.

DFT היא פונקציית התמרה בסיבוכיות O(), קיימת גם פונקציית FFT המבצעת התמרת פורייה ביעילות של O() אך מצריכה שכמות הדגימות תיהיה חזקה של 2. פונקציית ה-FFT משתמשת בשיטת "הפרד ומשול" (Devide and Conquer) כדי להעלות את היעילות של הפונקציה.

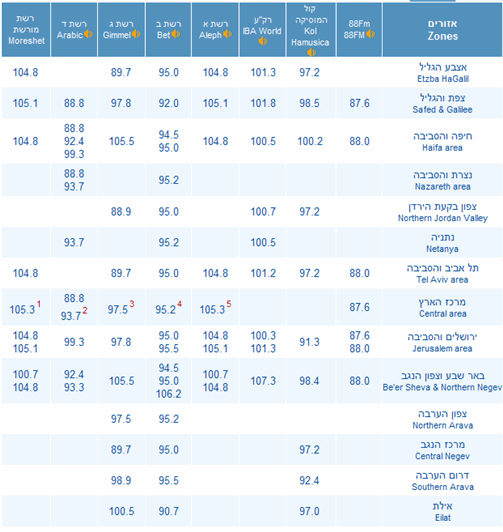
אפנון–Modulation:

אִפְנוּן הוא תהליך שינוי של גל מחזורי (הנקרא גל נושא), שהוא לרוב גל סינוסי בתדר הגבוה מזה של המידע המאופנן, באמצעות אות מידע.

אפנון משמעו שינוי, ויסות או כוונון והוא למעשה העברה של מידע על גל נושא: אנו משדרים מידע על גל נושא ואז מפענחים אותו. גל נושא יכול להיות גל אלקטרומגנטי כמו גלי רדיו – שמשמשים לדוגמא לשידורי טלוויזיה, רדיו ואף לאינטרנט אלחוטי. אך שימוש של גל נושא יכול להיות גם דרך סיבים אופטיים ואז אנו משתמשים 'בגלי אור' (שימוש טכנולוגי נפוץ בקווי טלפון).

הצורך באפנון:

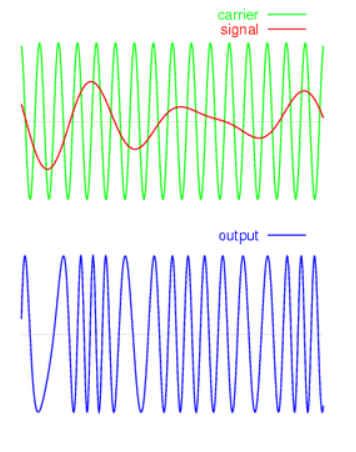
ישנן הרבה מאוד מערכות שכמעט כולן מעבירות את אותם התדרים ובכדי לנצל את כלל הספקטרום מאופן מלא אז "ממירים" את התדר שרוצים להעביר לתדרים אחרים שיהיה אפשר להעביר מבלי להפריע אחד לשני. אפנון נותן לנו גם להעביר את אות המידע למרחקים גדולים יותר, שכן האות הנושא הוא בעל תדר גבוה יותר.

[[13]](#footnote-13)

FM Modulation:

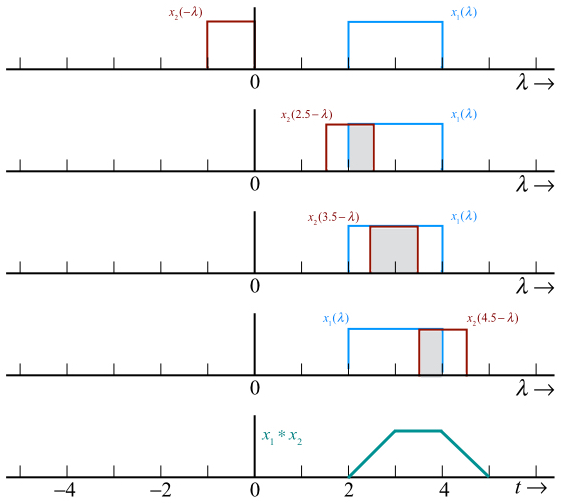
היא שיטת אפנון שבה אנו מייצגים את המידע ע"י שינויים בתדר של הגל הנושא. אפנון תדר הוא הוספת גודל אות המידע (מוכפל בקבוע ידוע) לתדר אות הנושא.

כאשר אנו רוצים לשמור מידע מסויים, הנקרא אות המידע, נשתמש בגל נושא (אות מחזורי בעל תדר גבוה) לתדר אות הנושא. כפי שניתן, לראות בשרטוט, הגל הירוק הוא גל הנושא, אות המידע הוא האדום, האות המאופנן (הכחול) אנו מחזורי – קצב השינוי שלו (התדר) לא אחיד (משתנה). ניתן לראות כי האזורים שבהם האות המאופנן בעל תדר גבוה יותר הם האזורים שבהם אות המידע (האדום) הוא בעל ערכים גבוהים – ולהפך.

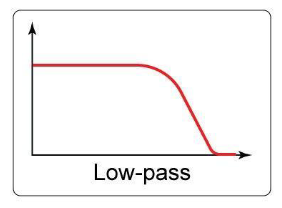
[[14]](#footnote-14)

מסנן ספרתי:

מסנן ספרתי מתאר מערכת המבצעת פעולה ליניארית על דגימות של אות. במוצא המסנן מתקבלות דגימות חדשות כאשר כל דגימה היא סכום משוקלל (קונבולוציה) של הדגימות שלפניה או אחריה. באמצעות המסנן ניתן להדגיש או להפחית תכונות של אות דגום.

[[15]](#footnote-15)

בפרוייקט זה השתמשתי במסנן מעביר תדרים נמוכים (Lowpass Filter) – מעביר רק אותות עד תדר מסוים, כל תדר גבוה יותר מסונן. משמש לסינון רעשים מאותות שמע.

[[16]](#footnote-16)

תהליך הפלטור עובד על עיקרון של חלונות. חלון הוא אוסף של דגימות אשר יכול, בעזרת פעולה מתמטית הנקראת קונבולוציה, להפריד בין תחנות שונות ובכך למנוע רעש הנוצר משני תחנות אשר נמצאות קרובות מידי. כדי לעשות את זה מבצעים קונבולוציה בין דגימות החלון (החלון צריך להיות בגודל שרוצים לסנן את האות אליו, במקרה של תחנות FM זה khz75) לבין דגימות האות.

[[17]](#footnote-17)

כאשר K הוא מספר הדגימות בחלון ו-h הוא החלון.

דמודוליזציה:

בשביל דמודוליזציה ישנם שני שלבים עיקריים; הדמודוליזציה עצמה ודסימציה.

תחילה צריך לחלץ את אות המידע מהאות, לשם כך משתמשים בדמודוליזציה. דמודוליזציה זה תהליך בו מכפילים את האות באות ההפוכי\* שלו בדיליי ובכך ניתן לחלץ את אות המידע על ידי מציאת הפאזה (בכדי להיפתר מהערך הממרוכב של האות).

[[18]](#footnote-18)

\*מספר הופכי הוא מספר שאם מכפילים אותו במספר המקורי מקבלים 1.

השלב השני הוא דסימציה. הרעיון של דסימציה הוא להקטין את התדר של האות כדי שיהיה אפשר לשמוע אותו (בדרך כלל 44.1khz). בשביל לעשות את זה, כל דגימה n באות החדש תיהיה שווה לדגימה ה-n\*m באות החדש (כאשר m הוא הגודל שרוצים לבצע דסימציה בו). לאחר מכן יש להעביר פילטר נוסף כדי למנוע רעש היכול להיווצר כתוצאה מהדסימציה.

[[19]](#footnote-19)

לאחר העברת האות בכל השרשרת מתקבל אות קול (דגימות ממשיות) אשר ניתן להשמיע או לעבד.

למידת מכונה:

המטרה המרכזית של למידת המכונה היא טיפול ממוחשב בנתונים מן העולם האמיתי עבור בעיה מסוימת, כאשר לא ניתן לכתוב תוכנת מחשב עבורה למשל, בעיית זיהוי שמומחה אנושי מסוגל לפתור, אך לא מסוגל לכתוב את הכללים לזיהוי בצורה מפורשת, או שהם משתנים עם הזמן ולא ניתנים לכתיבה מראש.

מטרת הלמידה יכולה להיות מידול, חיזוי או גילוי (דטקציה) של עובדות לגבי העולם האמיתי.

נהוג לחלק את אלגוריתמי למידת המכונה למספר סוגים:

למידה מונחית (supervised learning): כל דוגמה מגיעה עם תווית סיווג. מטרת האלגוריתם היא לחזות את הסיווג של דוגמאות חדשות שאותן לא פגש בתהליך הלמידה. אימון של רשת עצבית מלאכותית ("רשת נוירונים") מסתמך על אלגוריתמים מסוג זה.

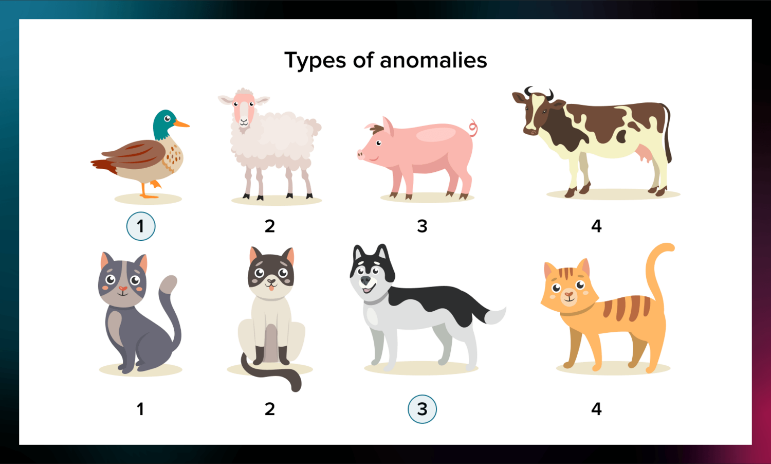
למידה בלתי מונחית (unsupervised learning). מטרת האלגוריתמים היא למצוא ייצוג פשוט וקל להבנה של אוסף הנתונים. שיטות נפוצות מסוג זה הן חלוקה לצברים (clustering), והטלה ליריעות ממד נמוך כגון ניתוח גורמים ראשיים (PCA).

למידת חיזוק (reinforcement learning). אלגוריתם הלמידה מקבל משוב חלקי על ביצועיו (רק לאחר סיום ביצוע המטלה) ועליו להסיק אילו מהחלטותיו הביאו להצלחה/כישלון.

לצורך הפרוייקט שלי בחרתי להשתמש בלמידה בלתי מונחית, שכן אין לי דרך לתת תווית סיווג לכל דוגמית של משהו שהוא "לא מוזיקה".

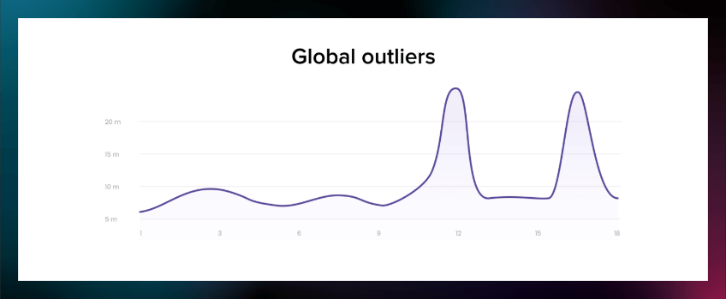
Anomaly vs Novelty Detection:

זיהוי חריגים (Anomaly Detection) הם אלגוריתמים אשר מזהים חריגים (Outliers/Anomalies) מתוך קבוצה של מידע תקין. חריג הוא כל נקודה השונה מהנורמה.

[[20]](#footnote-20)

ישנם שני סוגים עיקריים של חריגים, global ו-contexual.

Global: חריג גלובלי הוא חריג השונה באופן דרסטי מכל שאר הנקודות במאגר המידע, במילים אחרות, מקרה נדיר.

[[21]](#footnote-21)

Contexual: חריג הקשרי הוא חריג השונה מנקודות מידע אחרות ביחס להקשר המצופה מהתנהגות רגילה של נקודות המידע. למשל, בדרך כלל בחנויות יש עליה במכירות לפני תקופת החגים. אם יש עליה במכירות בזמן שהוא לא תקופת החגים זה חריג הקשרי.

[[22]](#footnote-22)

ישנם גם חריגים קבוצתיים (Collective), נקודות מידע שלא יהיו חריגים אם הם היו בודדים אבל כאשר יש קבוצה שלהם הם ניהיים חריגים.

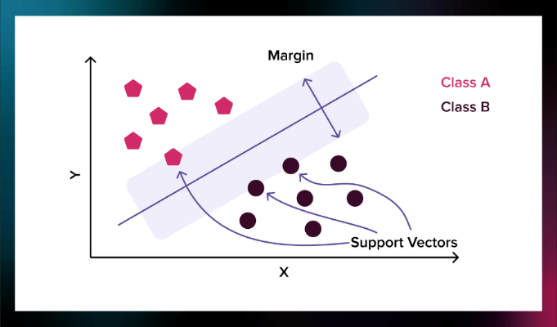
זיהוי חריגים יכול למצוא לי אילו נקודות מידע הם החריגים בתוך מאגר מידע אך לא יוכלו לחזות האם נקודה חדשה היא חריגה או לא, לשם כך צריך להתמש במודל זיהוי חידוש (Novelty Detection).

מודל הזיהוי שאני השתמשתי בו הוא מודל OCSVM (One Class Support Vactor Machine).

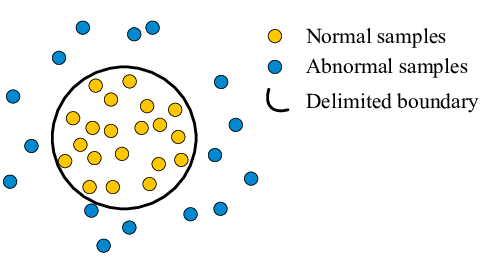
OCSVM:

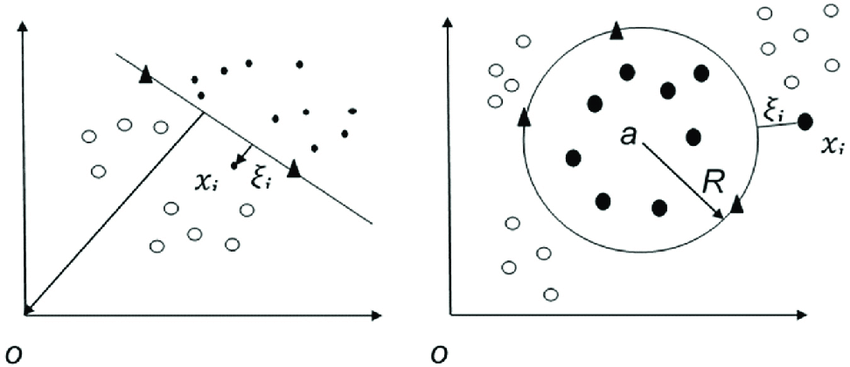
כדי להבין מה זה OCSVM תחילה יש צורך להבין מה זה מודל SVM רגיל.

מודל SVM הוא מודל המשמש בדרך כלל לחלוקת צברים (clustering) על ידי יצירת וקטור מפריד בין נקודות במימדים גבוהים. ניתן להשתמש בשוליים כדי לתת למודל אפשרות לזיהוי חריגים.

[[23]](#footnote-23)

מודל ה-OCSVM מאפשר להפוך את המודל הזה למודל חיזוי אשר מיצר מצבור (cluster) יחיד אשר מאחד את כל נקודות המידע התקינות סביב היקף של מעגל (במימד גבוה) ומיצר רדיוס למעגל הזה, כל נקודה הנמצאת מחוץ למעגל הינה חריגה.

[[24]](#footnote-24)

[[25]](#footnote-25)

כאשר Xi מייצג נקודת מידע ו-iξ מייצג את המרחק מהוקטור המפריד (ובמקרה של OCSVM מהיקף המעגל), אם iξ ≤ R אז הנקודה Xi היא לא חריגה, אחרת היא כן.

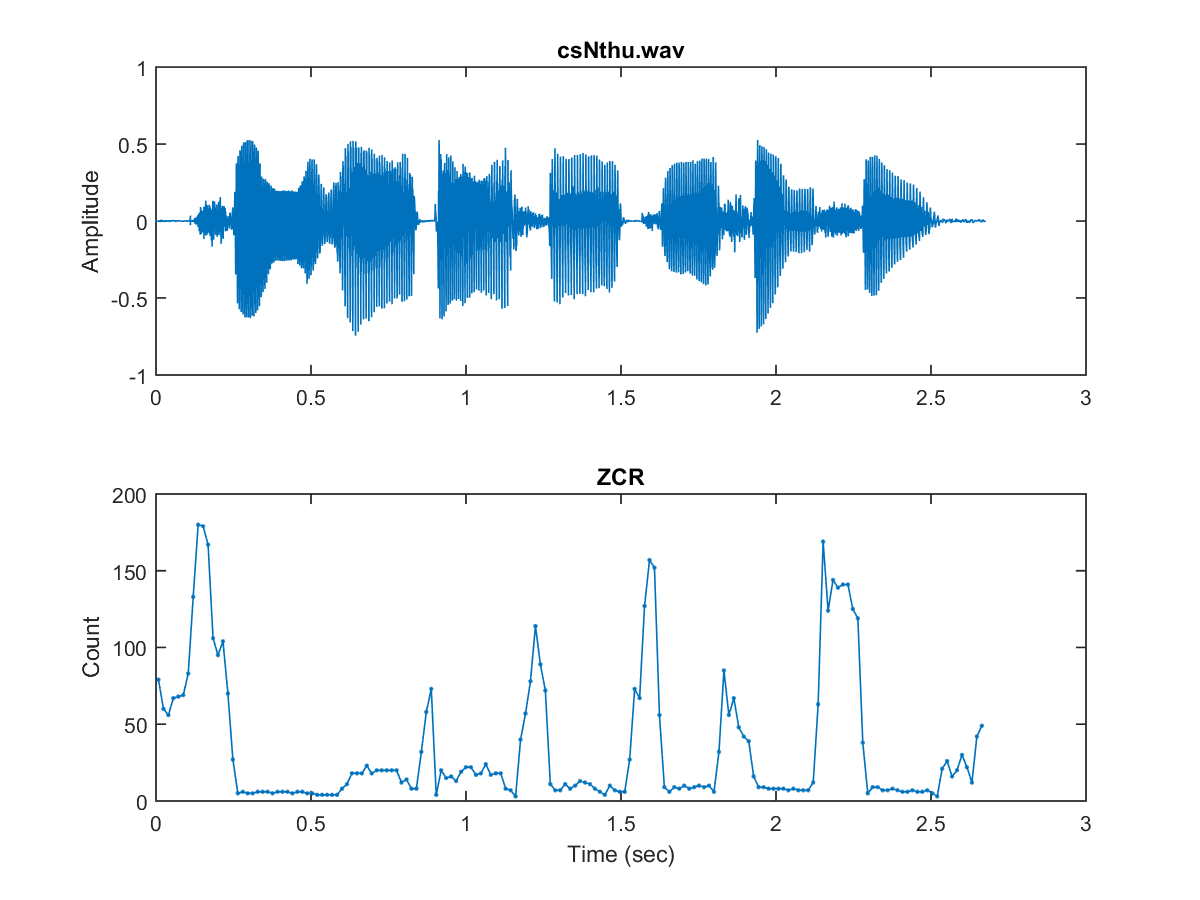
תכונות של אותות אודיו:

לאותות אודיו תכונות מסויימות היכולות לשמש לחיזוי וקטלוג של אותות אודיו לקבוצות שונות. תכונות אלו מחולקות לשלושה רבדים; תכונות במימד הזמן, תכונות במימד התדר ותכונות במימד התדר-זמן.

אני בחרתי לעבוד עם תכונות במימד הזמן משום שהן יותר פשוטות לזיהוי ומאפשרות להשיג מודל מדוייק ומהיר.

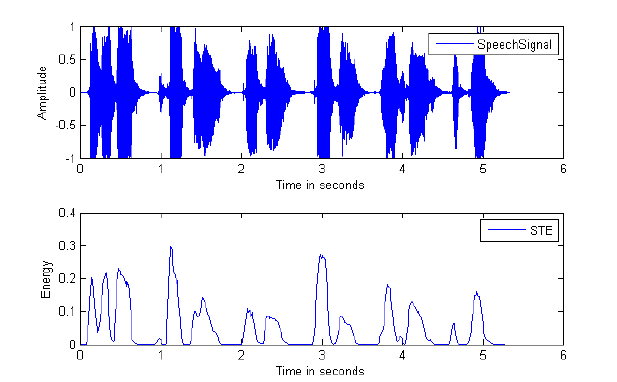
Zero Crossing Rate-ZCR:

ה-ZCR הוא קצת השינוי שבוא אות חוצה את קו האפס בתווך זמן מסויים. מחלקים את האות לחלונות של 512 דגימות וסופים כמה פעמים חוצה האות את קו האפס בכל חלון, לאחר מכן מוצאים את סתיית התקן כדי לדעת את ה-ZCR. ה-ZCR במוזיקה גדול יותר מזה של דיבור.

[[26]](#footnote-26)

Short Time Energy-STE:

ה-STE הוא ייצוג של השינויים באמפליטודה לאורך הזמן. גם פה סופרים את שינוי האפליטודה בחלונות של 512 דגימות ומוצאים את הvariance- שלהם. במוזיקה ה-STE גבוה מזה של דיבור.

[[27]](#footnote-27)

רקע-USRP:

SDR Software-Defined Radio – SDR הינו מערכת לתקשורת רדיו אשר רוב רכיבי התקשורת שבה ממומשים ע"י תוכנה על גבי מחשב PCאו מחשב Embedded. רכיבים שמסורתית היו אנלוגיים – מסננים, מחוללי אות, אפנונים, מפענחים וכו‘,ממומשים במקום זאת בתוכנה על בסיס עיבוד אותות דיגיטלי (DSP). תכונה זאת מאפשרת למערכת הרדיו מסוג SDR גמישות רבה במימוש פרוטוקולי תקשורת שונים ובתחומי תדר שונים על גבי אותה מערכת רדיו, שינוי באחד מהמרכיבים הנ"ל דורש ביצוע שינויי תוכנה בלבד (ללא שינויי חומרה) וניתן אף לעשות אותו תוך כדי פעולה ב-Real Time. מערכותUSRP פתוחות למשתמש–סכמות של שרשראות ה-RF,לוחות האם ושאר החומרות ניתנים לצפייה ע"י המשתמש.

רקע-:UHD

התממשקות מחשב המעבד DSP לחומרת ה-SDR נעשת בעזרת מנהל התקן (Driver) קוד חופשי בשם UHD. ה-API אחיד לכל ה-USRP – תוכנה אשר נכתבה תוך שימוש ב-UHD תעבוד על כלל ה-SDR מסדרת USRP ללא צורך בשכתוב קוד.

ניתן לעבוד מול התוכנה הנ"ל בצורה ישירה ע"י ה-API שלו בשפת C++ או Python אשר מכילה את העצמים והפונקציות הדרושים על מנת להעביר דגימות DSP ולהתאים את פרמטרי החומרה של ה-USRP (שעון, תדר מרכזי וכו')

ממשק תקשורת למכשיר ה-USRP:

המחשב שולט בחומרת ה-SDR דרך ממשק תקשורת דיגיטלי כגון USB או Ethernet. דרך ממשק זה, המחשב מתקשר עם רכיב אחר – בד"כ FPGA, אשר הוא עצמו מבצע את השינויים הדרושים בחומרה.

כדי לעמוד בקצבי הדגימה הגבוהים הדרושים לעיתים ע"י מערכות SDR, ממשק התקשורת הינו בעל קצב העברת נתונים גבוה – לדוגמה: PCI, USB 3, Ethernet 10Gbps.

## 7. תיאור המצב הקיים

קיום, השיטות העיקריות לזיהוי מוזיקה משומשות בעיקר למציאת שירים (תוכנות כמו shazam) אך לא להבדלה בין שירים לדיבור בחיי היום-יום.

פרוייקט זה מאפשר לבן אדם הרגיל לגשת למודל למידת מכונה היוכל להביל בשבילו בצורה גמישה באיזו תחנה משודרת מוזיקה.

## 8. ניתוח חלופות מערכתי

* ספרייה הנקראת PyTorch הנועדה לעזור ביצירת מודלים של למידת מכונה בPython-.
* ספרייה הנקראת Sklearn שגם כן נועדה לעזור ביצירת מודלים של למידת מוכנה.

## 9. תיאור החלופה הנבחרת

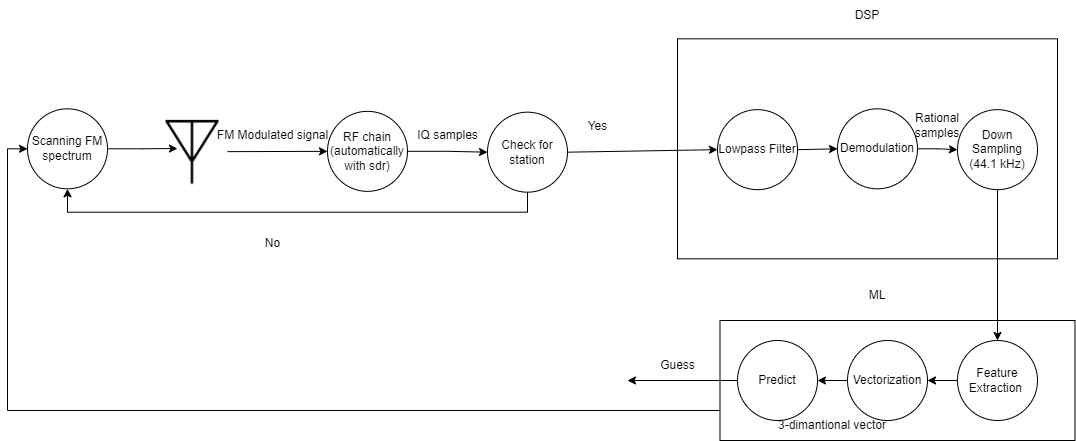
בחרתי להשתמש בSklearn- משום שPyTorch- מיועד בעיקר לזיהוי תמונות ו-NLP דבר שלא עוזר לי מאוד בתחום הידע שפרוייקט זה עוסק בו. לעומת זאת Sklearn מציע מודלים מתמטיים רבים ושליטה נרחבת בפרמטרים המאפשרת שליטה טובה ברמת הדיוק של המודל.

## 10. אפיון המערכת שהוגדרה

## 10.1 ניתוח דרישות המערכת:

* דגימות ייקלטו מהאוויר בעזרת מכשיר ה-USRP שמחובר באמצעות כבל למחשב.
* תכלול פונקציות לסינון, הורדת תדר ודימודולציה של אות מאופנן.
* תדע לשלוף תכונות מאות קול.
* תדע לחזות האם המידע שנקלט הוא מוזיקה או לא.

## 10.2 מודול המערכת:

[[28]](#footnote-28)

## 10.3 אפיון פונקציונלי:

זיהוי תחנות:

המערכת צריכה להיות מסוגלת לזהות תחנות.

כדי לזהות תחנה יש צורך למצוא את ממוצע האנרגיה של הספקטרום של אות על ידי ממוצע של סכום חזקות החלקים המדומים והחלקים הממשיים של דגימות האות.

[[29]](#footnote-29)

לאחר מציאת הממוצע בודקים האם הוא מעל ל-thresh hold מסויים. כדי למצוא את ה-thresh hold הזה השתמשתי בספקטרום אנלייזר (מכשיר המראה ספקטרום) כדי למצוא את התחנות. ה--thresh hold שנמצא הוא מינוס 75Db. אולם הממוצע מביא מספר ממשי שלא מיוצג בייצוג לוגריטמי ולכן יש צורך לנרמל אותו בעזרת הנוסחה:

הבעיה בנוסחה זו היא שהיא עובדת על ערכים הנמדדים בוואלטים והדגימות שלי הן דגימות IQ דיגיטליות. כדי "לתרגם" את הייצוג הלוגריטמי לאחד שיתאים עם ערכי הוואלטים יש צורך להוסיף קבוע נרמול, במקרה זה מינוס 32.

שרשרת DSP:

המערכת צריכה להיות מסוגלת לבצע שרשרת עיבוד אותות דיגיטליים (DSP) הכוללת סינון, דימודולציה ודסימציה.

**סניון:**

בפרוייקט השתמשתי במסנן תדרים נמוכים (Lowpass Filter) המעביר רק תדרים הנמוכים מתדר רצוי. המתרה של מסנן זה הוא לסנן רעש שנוצר מתחנו ש"עולות" אחת על השני ולעזור לי להתמקד בתחנה אחת.

**דימודולציה:**

היכולת לבצע דימודולציה על תדרים שאופננו באיפנון FM ולהוציא מתוכם את אות המידע ששודר.

**דסימציה:**

האפשרות להקטין את אות המידע שהתקבל לתדר שניתן לשמוע.

שליפת תכונות אודיו:

המערכת צריכה להיות מסוגלת לשלוף תכונות של גלי אודיו:

ZCR: המערכת צריכה להיות מסוגלת לספור את כמות הפעמים שהאות חותך את ציר האפס ולדעת לחשב את סתיית התקן שלהם.

STE: המערכת צריכה לדעת לחשב את מעטפת האנרגיה של האות ולחשב את ה-variance שלו.

Spectral Centroid: המערכת צריכה להיות מסוגלת לחשב את "מרכז הכבידה" של הספקטרום של האות.

חיזוי:

המערכת צריכה להיות מסוגלת להשתמש במודל למידת מכונה (OCSVM) כדי לזהות האם המידע ששודר היה מוזיקה או לא.

כדי להשתמש במודל אפשר לבנות אותו מחדש בכל ריצה או ללמד אותו מראש ולטעון אותו. במקרה של פרוייקט זה לימדתי את המודל מראש ושמרתי אותו כקובץ בינארי בעזרת הספריה pickle.

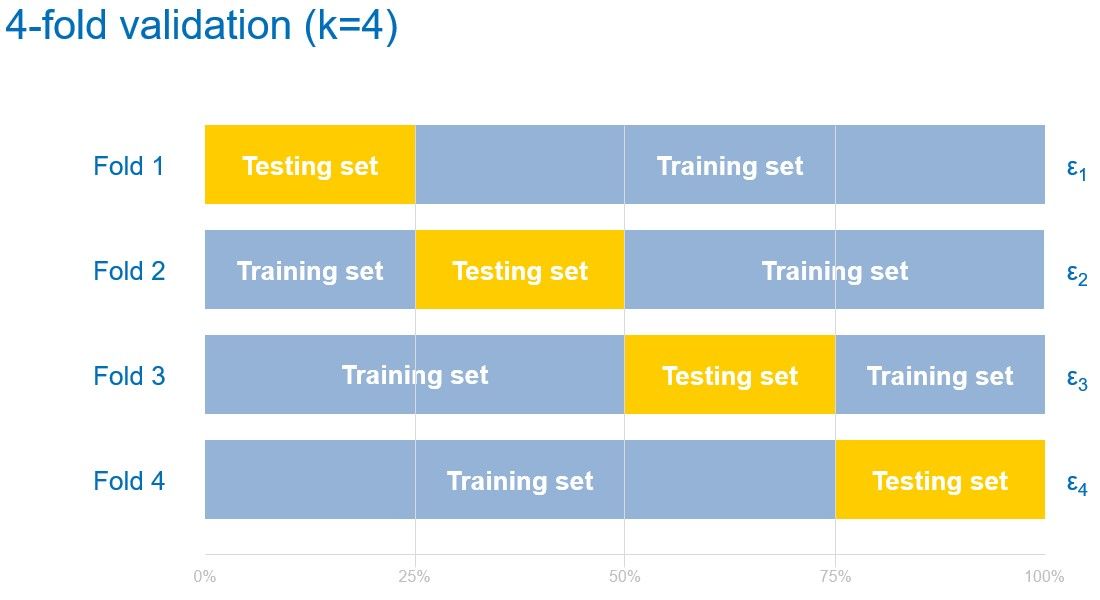
## 10.4 ביצועים עיקריים:

ביצועי המערכת נבדקו בשני סביבות שונות; ריצת זמן אמת והדגמה סטטית. בשני המקרים המערכת עבדה כרצוי.

בנוסף, כדי לבדוק את רמת הדיוק של המודל השתמשתי ב-Confusion Matrix וב-cross validation.

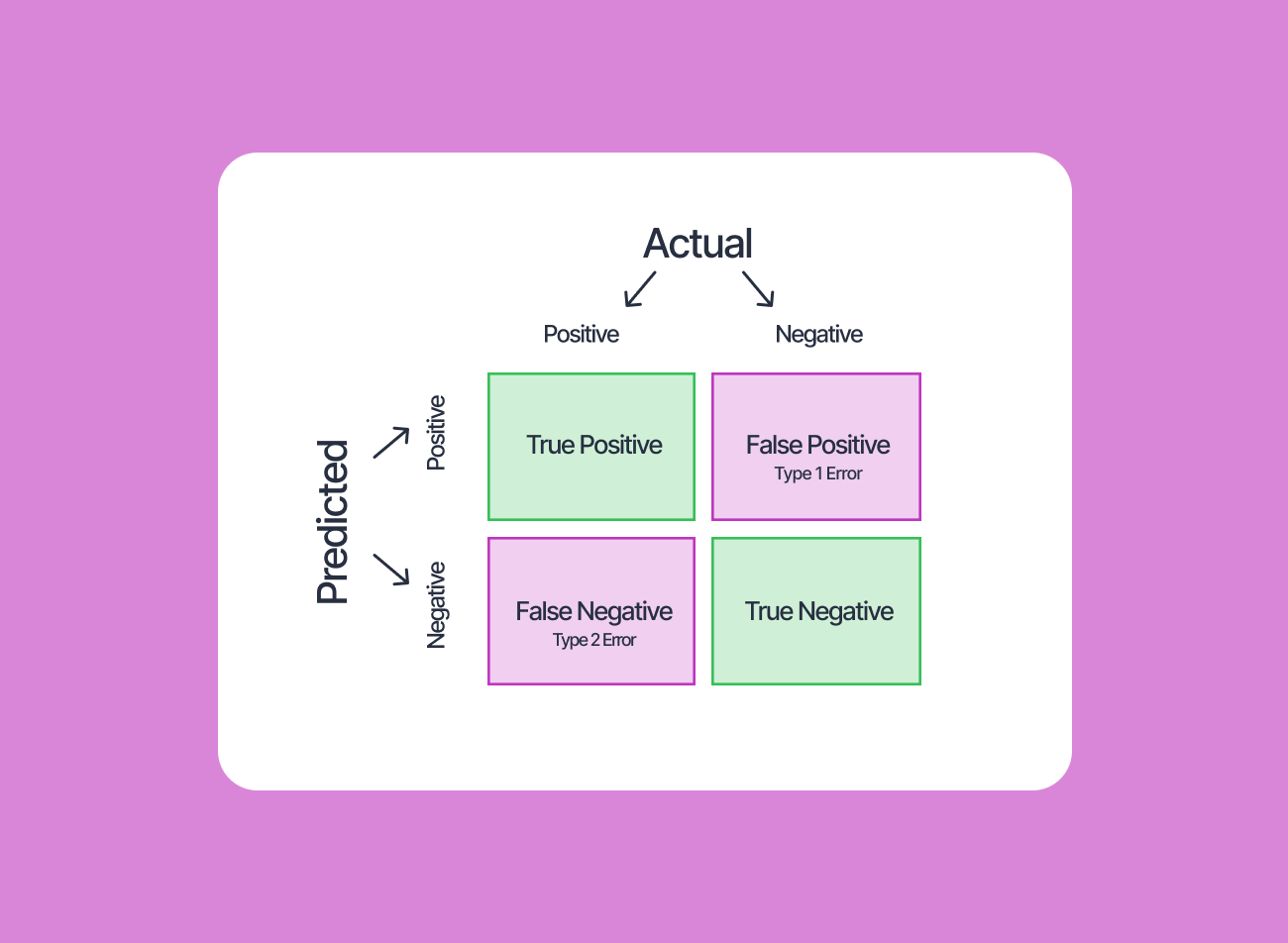
Cross Validation:

Cross validation היא שיטה המאפשרת למנוע over fitting של מודל. השיטה אומרת שיש לחלק את סט הנתונים שלי ל80% אימון ו-20% בדיקה. לאחר הבדיקה, במידה והמודל לא היה מדוייק מספיק, מבצעים עליו את שינויי הפרמטרים הרצויים. לאחר מכן, בודקים שוב, אך הפעם משתמשים ב-20% אחרים מסט הנתונים.

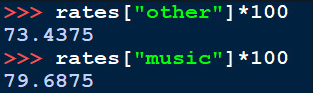
[[30]](#footnote-32)

Confusion Matrix:

הרעיון בשימוש ב-confusion matrix הוא לתת רמת דיוק יותר טובה לבדיקה. הדבר הראשון שחושבים עליו כדי לבדוק האם מודל הוא מדוייק הוא לתת למודל את כל סט הבדיקה ולראות כמה מהם הוא צדק. הבעיה בשיטה זו שיכול להיות אחוז גבוהה של זיהוי לא נכון שלא נשים לב אליו (אם למשל נבדוק כמה פעמים הוא טוען שמשהו הוא מוזיקה אבל לא כמה פעמים הוא טוען שתשדיר חדשות הוא גם כן מוזיקה). בשביל לפתור בעיה זו ניתן להשתמש במטריצת שגיאה אשר יכולה להדגים את טווח השגיאה של המודל.

[[31]](#footnote-33)

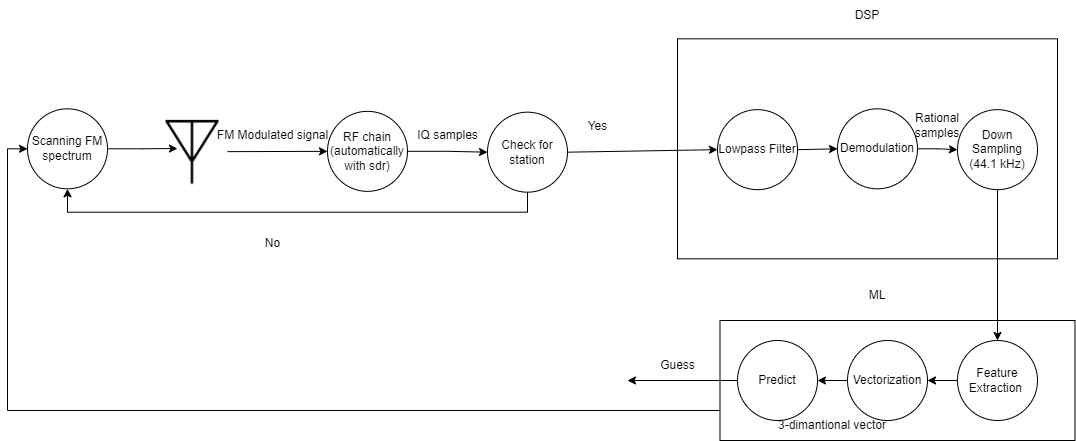
לאחר הרצת המערכת ובדיקתה ניתן לראות את אחוזי ההצלחה שלה:



## 10.5 אילוצים:

* סביבת פיתוח:
  + Visual studio code (VS Code)
* שפת תכנות:
  + Python (לצורכי התממשקות עם ה-USRP)
* תנאי הפעלה:
  + המערכת חייבת להיות מחוברת להתקנים מכל הצדדים (ethernet, sdr) לצורכי קליטה.
  + המערכת חייבת לרוץ במקום ללא הרבה הפרעות.

## 11. תיאור הארכיטקטורה

28

פרוייקט זה נעשה בשיטת Top-Down. תחילה, המערכת תוכננה ברמה האבסטרקטית, שרשרת הקליטה, שרשרת העיבוד ומודל לימוד המכונה. לאחר מכן התבצע מימוש מעשי של כל רכיב בשפת Python. הנתונים מועברים מפונקציה אחת לשניה דרך פרמטרים שונים, למעשה הפרמטרים עצמם כמעט ולא משתנים בלולאה הראשית, רק דרך פעולות מיוחדות אשר משנות ומשפיעות על הפרמטרים.

## 11.1 הסבר בלוקים מפורט:

בחלק זה אתן הסבר מפורט על כל אחד מהבלוקים העיקריים המרכיבים את המערכת.

Scanning FM spectrum:

סריקת ספקטרם ה-FM. מתחילים בתדר של 88mHz ועוברים בקפיצות של 400 kHz, משום שה-SDR קולט שם את התדר המרכזי כתדר האפס וקולט 200kHz לכל כיוון. גודל ממוצע של תחנה הוא 100kHz ולכן קפיצות של 400kHz יעזרו למנוע קליטה כפולה של תחנות. ממשיכים בקפיצות כאלו עד לתדר של 108mHz.

RF Chain:

תהליך קליטת האותות, דגימתם והצבת התדר המרכזי סביב ציר האפס נעשה באופן אוטומטי על ידי מכשיר ה-SDR.

Check for station:

בעזרת שימוש בהתמרת פורייה ניתן למצוא את מעטפת האנרגיה של האות, במידה והאנרגיה הממוצעת של האות היא מעל למינוס 75Db אז ישנה תחנה בדר הנקלט.

Lowpass Filter:

בלוק זה נכתב בשפת Python ובעזרת ספריית numpy. בלוק זה מבצע סינון בתדר של 100kHz על ידי קונבולוציה בין החלון לאות.

Demodulation:

בלוק זה נכתב בשפת Python ובעזרת ספריית numpy. הבלוק מקבל את דגום בשיטת IQ (מרוכב) ומבצע עליו פעולת דימודולציה (עם הנוסחה המתמטית שהוצגה קודם) ומחזירה מערך של דגימות ממשיות (אות אודיו).

Decimation:

בלוק זה נכתב בשפת Python. הבלוק מקבל את הדגימות ואת סדר הגודל בו רוצים להפחית את התדר (במקרה הזה, פי 25) ומחזיר אות חדש אשר הגודל שלו מופחת פי 25 (40kHz).

Feature Extraction:

בלוק זה נכתב בשפת Python ובעזרת ספריית numpy ו-librosa. הבלוק משתמש בפעולות מתמטיות כדי להחזיר את התכונות הרלוונטיות.

Vectorization:

תהליך איסוף המידע לוקטורים.

Predict:

המודל מקבל את הוקטור המייצג את המידע על אות הרדיו המדובר ומחזיר את החיזוי שלו; אחד אם זה מוזיקה ומינוס אחד אם לא.

## 11.4 ארכיטקטורת רשת:

Ethernet:

Ethernetהיא טכנולוגיה לתקשורת נתונים ברשתות מחשבים מקומיות על פי מודל.TCP/IP

Ethernet הוא פרוטוקול תקשורת שאחראי על השכבה הפיזית של התקשורת.

טכנולוגית ) CD/CSMAהנכללת בפרוטוקול זה( מאפשרת לכל רכיב לשלוח אותות לתוך תווך משותף בלי התראה מוקדמת ומטפלת ביעילות בהתנגשות.

## 13. למידת מכונה

## 13.1 תהליך איסוף הנתונים:

אחד הדברים החשובים ביותר בלמידת מכונה הוא איסוף מידע אמין. כדי לאסוף מידע השתמשתי באתר שנקרא kaggle אשר מכיל מאגרי נתונים בנושאים רבים. מצאתי מאגר נתונים המכיל קטעי מוזיקה באורך 30 שניות של מגוון סגנונות מוזיקה שונים. אולם מאגר הנתונים הזה לא כלל שירים (מוזיקה + מילים) אז הייתי צריך לחפש עוד. לאחר חיפוש באינטרנט מצאתי מאגר של שירים מלאים. לאחר מכן הייתי צריך לחתוך את הקבצים לגודל שבו המערכת שלי קולטת (שניה אחת) ולהעביר אותם דרך תהליך שליפת המידע. לאחר מכן הכנסתי אותם לקובץ CSV המכיל את הנתונים על כל חלק של שניה אחת. בנוסף, משום שמערכת רדיו תמיד מכילה מעט רעש (הנקרא רעש תרמי) הוספתי רעש מלאכותי לגלי הקול.

## 13.2 סוג הלמידה שנבחר:

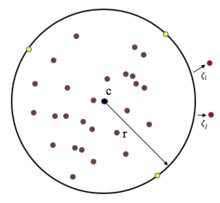
סוג הלמידה שנבחר הוא מודל מסוג OCC הנקרא OCSVM. מודל ה-OCSVM מבוסס על מציאת ה-hypersphere (מעגל במימד גבוה) עם רדיוס r ומרכז c המכולל את כל נקודות המידע.

אם מרחק הנקודה מהמרכז (D):

D>c: outlier (חריג)

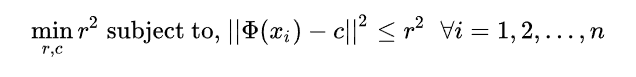
D<c: inlier (תקין)

D=c: support vector (וקטור הגבול – היקף המעגל)

[[32]](#footnote-34)

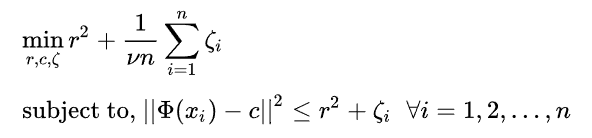
## 13.3 נוסחאות:

הנוסחה הבאה באה למצוא את הרדיו המינימלי ביותר של ה-hypersphere.

[[33]](#footnote-35)

כאשר היא פעולת העתקה של נקודת המידע R →F *:* כאשר *F* מימד ה-hypersphere ו- Rהוא מימד הנקודה.

הבעיה בנוסחה זו היא שהיא מאוד מגבילה חריגים, כלומר גם חריגים שהם מאוד מאוד קרובים לגבול יחשבו כחריגים. לכן, קיימת נוסחה יותר סלחנית.

[[34]](#footnote-36)

הנוסחה דומה מאוד לנוסחה הקודמת רק עם תוספת חדשה. החלק מייצג שוליים, כאשר נקודה נמצאת בתווח המותר (כאשר המרחק שלה היא ) אז היא נחשבת inlier אפילו אם היא מחוץ ל-support vector. הפרמטר v ϵ [0,1]  *קובע כמה נקודות יכולות להיכלל בתוך ה-*hypersphere*, ככול ה-*v *יותר קטן ניתן להכניס יותר נקודות ל-* hypersphere *וככול שהוא גדול יוצק גודל ה-* hypersphere *קטן יותר.*

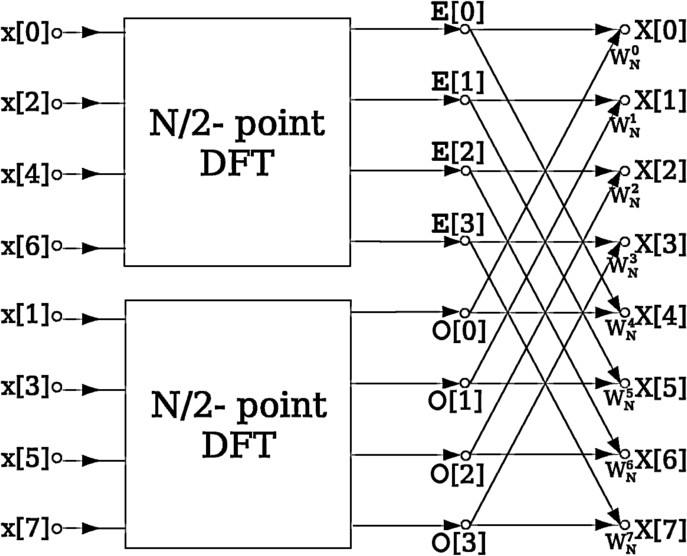
## 14. ניתוח ותרשים UML

## 14.1 אלגוריתמים ראשיים:

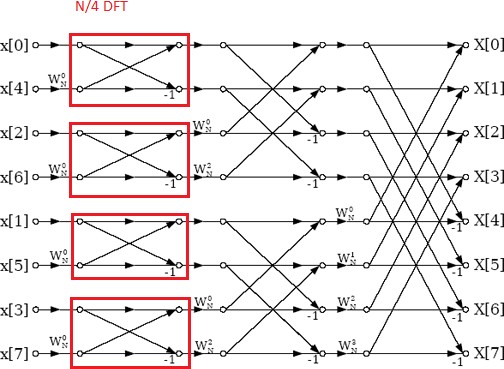
התמרת פורייה

אלגוריתם הFFT- המוכר ביותר והנפוץ ביותר הוא אלגוריתם FFT ,Cooley-Tukey הנקרא על שם מנסחיו. הוא בונה על כך שניתן להפריד התמרת פורייה בגודל N לשתי התמרות פורייה בגדלים ,N/2 ואותם לשתי התמרות פורייה בגדלים N/4 )אזי, הפרד ומשול,( עד שמגיעים למצב שכל התמרות הפורייה המורכבות מפוצלות רקורסיבית להתמרות פורייה פשוטות של .N=2 עם כל פיצול – האלגוריתם חוצה ב2- את מורכבות החישוב שלו ובכך הופכת אותו ליעילות בסדר גודל לוגריתמי. שיטה זו מגבילה את יכולת ה-FFT לעבוד רק על אוסף נתונים בגודל של חזקה של 2.

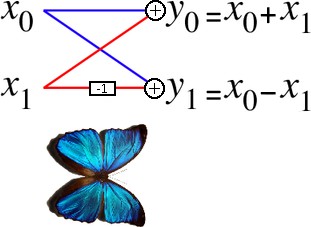
דוגמה לפיצול התמרת פורייה של N=8 לשתיים של ,N=4 כשאחד עובד על הדגימות במקומות הזוגיים והשני במקומות האי-זוגיים:

[[35]](#footnote-37)

ואז שוב פיצול של שני התמרות פורייה של N=4 לארבעה התמרות פורייה של ,N=2 כשאחד עובד על הדגימות במקומות הזוגיים )של התמרות (N=4 והשני על האי-זוגיים:

[[36]](#footnote-38)

התמרת הפורייה הבסיסית ביותר האפשרית היא התמרת פורייה N=2 .DFT התמרת פורייה זו ניתן לבצע בקלות ע"י משוואת "פרפר" )שם זה ניתן לה עקב צורתה בדיאגרמת זרימה:(

[[37]](#footnote-39)

לאחר ביצוע התמרת פורייה DFT N=2 עבור כל זוג נתונים משלבים אותם חזרה ומקבלים את האות במרחב התדר.

## 14.2 ניתוח תרשים וUse Cases-:

Usrp:

Usrp הוא מחלקה המבצעת את התקשורת עם ה-API של ה-SDR (UHD). בתוך המחלקה נמצאים פעולות המתאחלות את מכשיר ה-SDR, מתחילות stream תקשורת ומקבלות נתונים.

dsp:

מודול ה-dsp מכיל רשימה של פעולות הכוללות את כל הפעולות הנחוצות לעיבוד אותות. פעולות כמו סינון, FFT, דסימציה וכו'.

Feature Extraction:

מודול ה-Feature Extraction מכיל את הפעולות הנחוצות לשליפת התכונות של גלי אודיו (ZCR,STE וכו) כמו גם פעולה המחברת אותם לוקטור.

Model:

Model הוא אובייקט המודל המגיע דרך מחלקת sklearn ונשמר כקובץ בינארי עם העזרה של מחלקת pickle.

## 14.3 מבני נתונים:

מערך:

מערך (Array) הוא מבנה נתונים שמאפשר שמירה של משתנים רבים מאותו טיפוס תחת אותו שם, כאשר כל אחד מהמשתנים מקבל מספר מזהה ייחודי. שימוש במערכים מאפשר עבודה נוחה עם מידע שמורכב מחלקים רבים הזהים זה לזה.

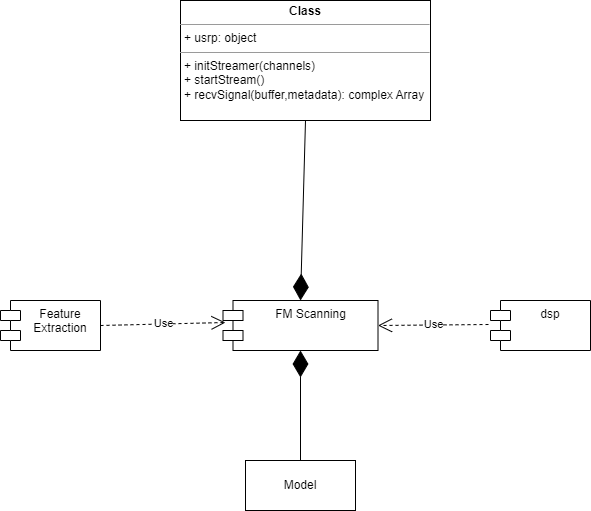
רשימה:

רשימה (List) הוא מיכל המאחסן אלמנטים, בדיוק כמו מערכים. ההבדל המהותי ביניהם הוא שווקטור הוא דינאמי, כלומר, בזמן ריצה, גודל הווקטור יכול לגדול אבל במקרה של מערכים, ברגע שהגודל מוגדר פעם אחת, לא קל לשנות את הגודל בהמשך התוכנית. בנוסף, בגלל שגודל המערכים הוא סופי בדרך כלל, הם נשמרים על זכרון הStack-, לעומת ווקטורים שנשמרים על זכרון ה.Heap- כמו כן, רשימה יכולה לאחסן סוגים שונים של נתונים באותו מיכל אחסון.

Data Frame:

Data Frame הוא מבנה נתונים טבלאי המאפשר להכיל בתוכו סוגים שונים של נתונים בפורמט טבלאי של עמודות ושורות בעלות שמות מזהים המאפשרים גישה ישירה לשורות ולעמודות. מבנה הנתונים זה משומש (במקרה של פרוייקט זה) כדי לקרוא את המידע מקובת ה-CSV המכיל את הנתונים על נקודות הלימוד של המודל ולבצע עליהם פעולות.

## תרשים UML:

[[38]](#footnote-40)

## 15. רכיבי ממשק

המערכת כוללת כמה רכיבים אלקטרוניים:

עמדת מפעיל – Control PC:

מחשב PC החייב לכלול חיבור Ethernet.

רדיו USRP של Ettus Research

רדיו Ettus USRP x310 שבאמצעותו נוכל לקלוט אותות.

## 16. תיכון המערכת

## ארכיטורת המערכת:

רכיב קליטה חומרי:

Ettus הינה חברת בת של Instruments National המתעסקת בפיתוח ושיווק של מערכות SDR תחת המותג USRP – Universal Software Radio Perioheral.

המודל שאני משתמש בו הוא x310 אשר בעזרתו אפשר לקלוט תדרים מ-6gHz עד 106 mHz, הוא כולל חיבור Ethernet ומאפשר התממשקות באמצעות ספריית UHD.

עיובד אותות:

לאחר קליטת האותות יש צור להעביר אותם בשרשת DSP לפני שמעבירים דרך מודל למידת המכונה.

## 17. תיאור התוכנה

## 17.1 סביבות עבודה:

Visual Studio Code הוא שמה של פלטפורמה מבוססת קוד פתוח לעריכת קוד ופיתוח תוכנה הפועלת על מערכות ההפעלה ,Windows לינוקס וX- .OS העורך תומך בניפוי שגיאות, בקרת גרסאות של גיט, המחשה סינטקטית של קטעי קוד, השלמת קוד חכמה, קטעי קודאוטומטיים, ושינויי קוד רוחביים.

הסיבה בה בחרתי לעבוד בעיקר היא הפשטות שבה ניתן לתכנת מכל מערכת הפעלה או סביבת עבודה, בתוכנה זו ישנה חנות תוספים (Extensions) שניתן להתקין ולהשתמש בקלות רבה אםמשהו חסר, העבודה עם תוכנה זו קלה ומהירה והיא אחת הסיבות למה בחרתי לעבוד איתה.

## 17.2 שפות תכנות:

Python היא שפת תכנות דינמית מהנפוצות ביותר. פייתון תוכננה תוך שימת דגש על קריאוּת הקוד, וכוללת מבנים המיועדים לאפשר ביטוי של תוכניות מורכבות בדרך קצרה וברורה. אחד המאפיינים הבולטים בתחביר השפה הוא השימוש בהזחה להגדרת בלוקים של קוד )ללא שימוש בסוגריים או במילים שמורות לצורך כך, כמו ברוב השפות הנפוצות.(

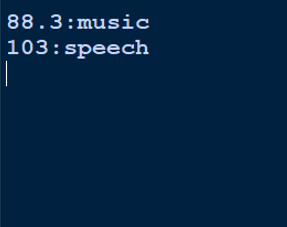
פייתון היא שפה מרובת־פרדיגמות, המאפשרת תכנות מונחה־עצמים, תכנות פרוצדורלי, ובמידה מסוימת גם תכנות פונקציונלי. לשפה ספרייה סטנדרטית גדולה וענפה, והיא תומכת באופן מובנה בהרחבה שלה אל שפות אחרות )בתלות במפרש: ,C ,C++ Java ו־.(C#

תוכניות פייתון נכתבות בקובצי טקסט, שעל־פי קונבנציה נשמרים במערכת ההפעלה בסיומת 'py.' . תוכניות פייתון עשויות לכלול קבצים רבים.

פייתון רצה בעיקר דרש מפרש (Interpeter) – למרות שאפשר להריץ אותה גם דרך מהדר (Compiler) – אשר מתרגם את שורות הקוד לפקודות שהמעבד יכול להריץ.

## 18. תיאור מסכים\ממשק משתמש

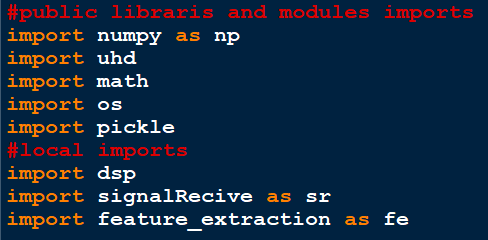
הפרוייקט עצמו הוא ללא ממש גרפי. התקשורת עם המשתמש תעשה דרך הקונסול (cmd, terminal). למשתמש יוצג את תדר התחנה (במידה והתדר המרכזי הוא תחנה) והם מדובר במוזיקה או דיבור.

[[39]](#footnote-41)

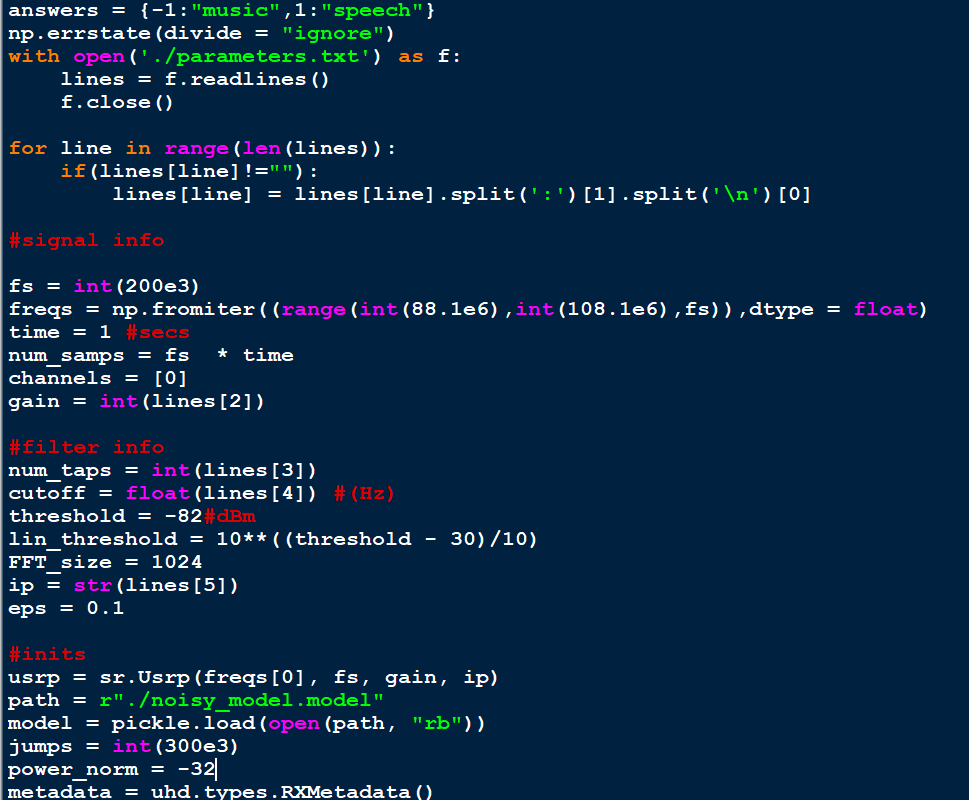
## 19. קוד התוכנית

לולאה ראשית:

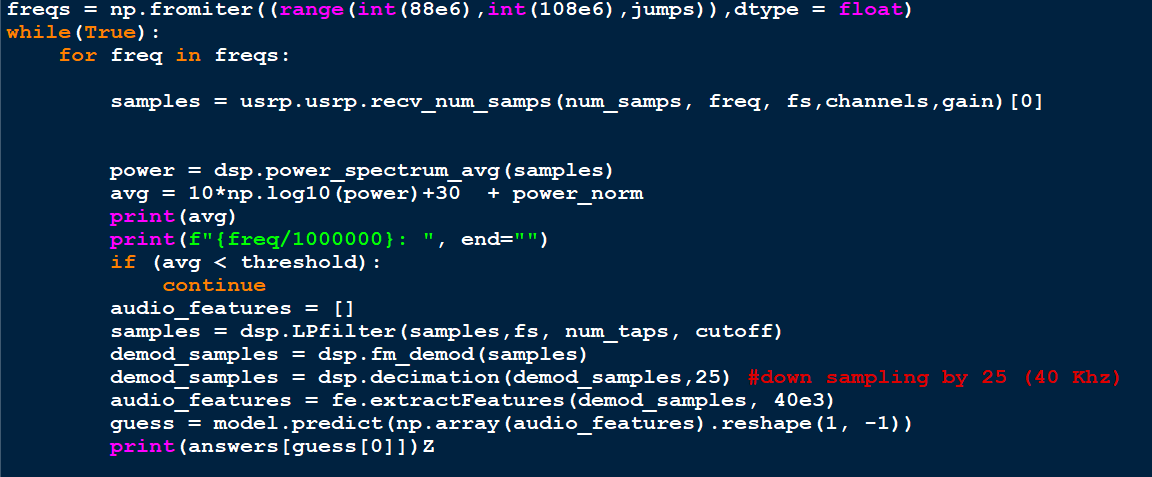
Imports של ספריות ומודולים, החלק הראשון הן ספריות ציבוריות והחלק השני הם מודולים שאני כתבתי.

1

אתחול נתונים; חלק מהנתונים מאותחלים דרך קובץ טקסט המכיל את הפרמטרים וחלק מהפרמטרים הם קבועים מראש ולא ניתנים לשינוי מבלי לערוך את הקוד.

2

הלולאה הראשית; לפני תחילת הלולאה יש אתחול של רשימת התדרים המרכזיים עליהם נעבור (ספקטרום ה-FM), לאחר מכן עוברים על כול תדר בספקטרום וקולטים נתונים המשודרים בו בעזרת הפונקציה recv\_num\_samps. לאחר מכן מחשבים את ממוצע הכוח, בעזרת הפונקציה power\_spectrum\_avg ומנרמלים את התוצאה. אם העוצמה היא מעבר ל-threshold אז מבצעים את שרשרת ה-DSP (LPfilter, fm\_demod, decimation), מוצאים את תכונות גלי האודיו (extractFeatures) ולבסוף מוצאים את הניחוש (predict) ומדפיסים את התוצאה.

3

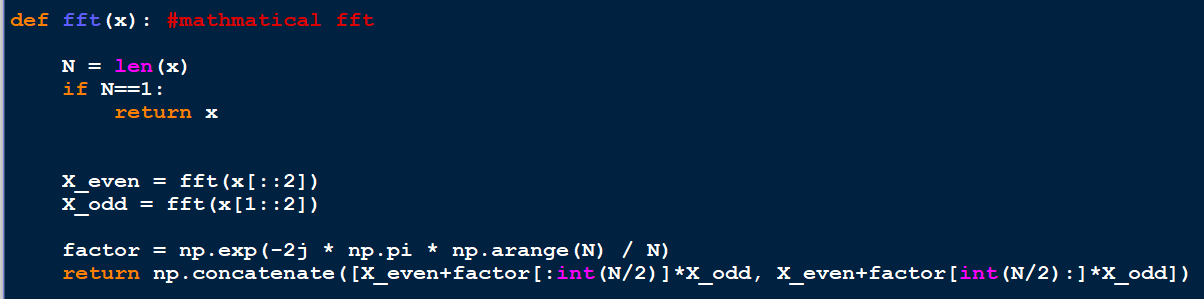
DSP:

FFT:

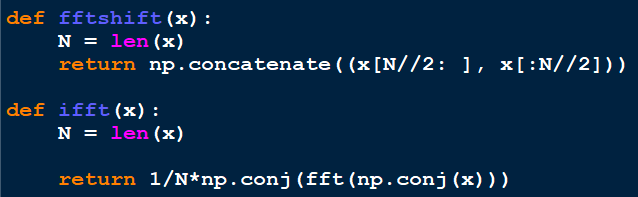
הפעולה מקבלת מערך של מספרים מרוכבים (דגימות במרחב הזמן) ומחזיקה מערך במרחב התדר.

אלגוריתם ה-FFT שמימשתי הוא אלגוריתם Cooley-Tukey, על שם מנסחיו.

האלגוריתם מחלק את המערך לאיבריו הזוגיים והאי-זוגיים ומבצעת על כל אחד מהם שוב את ה-FFT, לבסוף מחברים בן כל החלקים ומכפילים ב-factor ().

4

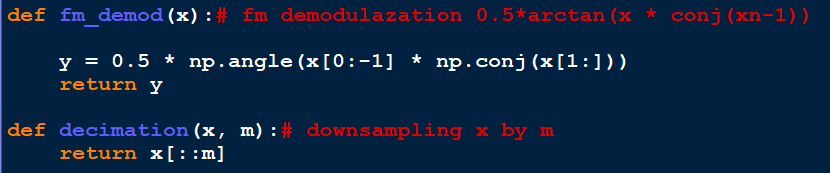
ישנם שתי פעולות נוספות, fft\_shift ו-ifft. fft\_shift "מזיז" אז הנתונים במימד התדר כך שלא יהיה תמונת מראה (משום שעבודה על ערכים מדומים תיצור תמונת מראה בצד השלילי של הציר) ו-ifft היא הפעולה ההפוכה של FFT וממירה חזרה ממרחב התדר למרחב הזמן.

5

דימודולציה ודסימציה:

פונקציית הדימודולציה מקבלת מערך של ערכים מדומים ומחזירה מערך של floats. פונקציית הדסימציה מקבלת מערך מכול זוג ומחזריה מערך מאותו סוג כאשר הוא קטן יותר בסדר גודל של m.

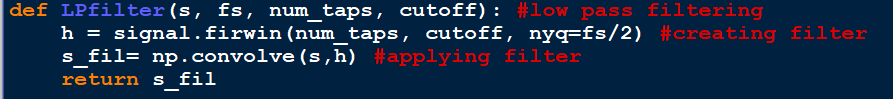
שתי הפעולות האלו משתמשות בנוסחאות המתמטיות שהוצגו קודם לכן כדי לבצע דסימציה ודימודולציה.

6

LP Filter:

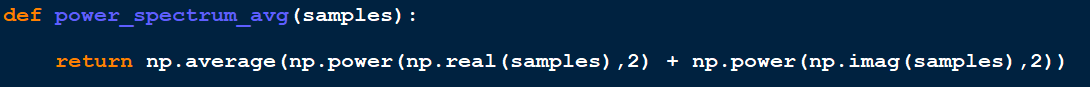
פונקציית ה- LPfilter מקבלת את האות, את תדר הדגימה, כמות המכפילים שאנחנו רוצים שיהיה לחלון (ככול שיש יותר מכפילים הסינון יהיה יותר טוב אך יקח יותר זמן, במקרה שלי אחרי מספר נסיונות נמצא כי 99 מכפילים הוא המספר הנכון) ואת תדר הגבול, התדר שאנחנו רוצים להעביר תדרים רק תחתיו.

הפונקציה משתמשת בפונקציה מספריית ה-signal כדי למצוא את מכפילי החלון (משום שזה פעולה מתמטית מסובכת מאוד) ואז אני מבצע קונבולוציה עם האות (במימד הזמן) כדי להפעיל עליו את המסנן.

7

power\_spectrum\_avg:

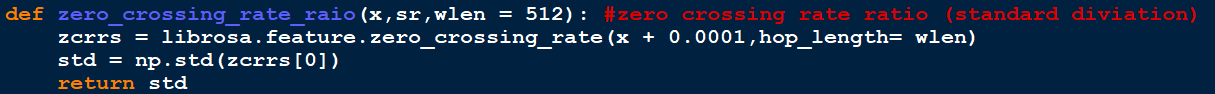
הפעולה מקבלת אות במימד הזמן (מערך של floats) ומחזירה את ממוצע האנרגיה של האות (מספר ממשי). הפעולה מממשת את הנוסחה המתמטית שהוצגה קודם לכן.

8

Feature Extraction:

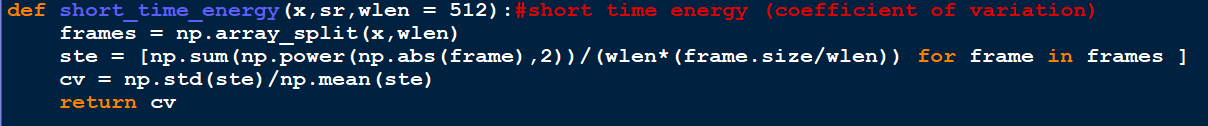
zero\_crossing\_rate\_raio:

הפונקציה מקבלת את האות (מערך של floats), את תדר הדגימה ואת הגודל של כל חלון (כברירת מחדל 512 דגימות) ומחזירה מספר ממשי. הפונקציה מחשבת את יחס השינוי של חציית האות את קו האפס ואז מחשבת את סתיית התקן בין החלונות.

9

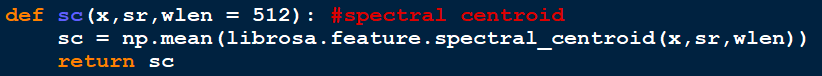
zero\_crossing\_rate\_raio:

הפונקציה מקבלת את האות (מערך של floats), את תדר הדגימה ואת הגודל של כל חלון (כברירת מחדל 512 דגימות) ומחזירה מספר ממשי. הפונקציה מחשבת את השינוי באנרגיה בין חלון לחלון ואז מחשבת את ה-coefficient of variation – סתיית התקן חלקי הממוצע.

10

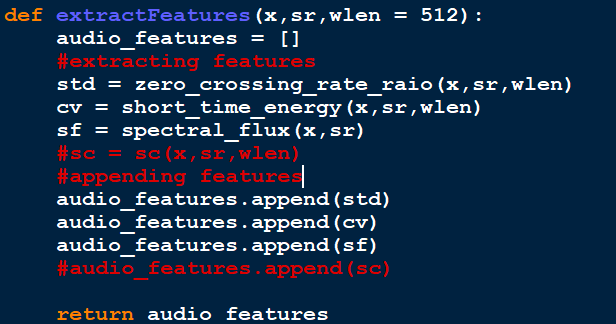
sc:

פונקצייה זו מחשבת את ה-spectral centroid – "מרכז הכבידה" של האות. הפונקציה מקבלת את האות (מערך של floats), את תדר הדגימה ואת הגודל של כל חלון (כברירת מחדל 512 דגימות) ומחזירה מספר ממשי. הפונקציה משתמשת בספריה כדי לבצע את הפעולה.

11

extractFeatures:

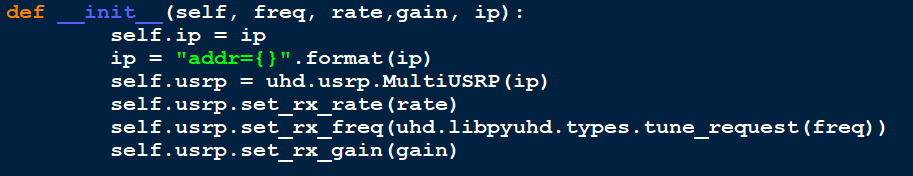
פונקצייה זו מחשבת את ה-spectral centroid – "מרכז הכבידה" של האות. הפונקציה מקבלת את האות (מערך של floats), את תדר הדגימה ואת הגודל של כל חלון (כברירת מחדל 512 דגימות) ומחזירה מערך המכיל את כל הנתונים. הפונקציה רק קוראת לפונקציות האחרות ומחברת הכל לוקטור.

12

signalRecive:

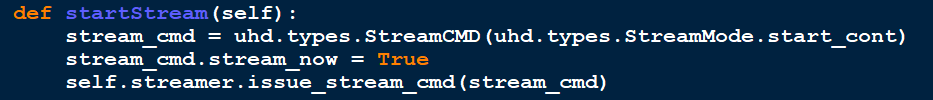
init:

הפונקציה מאתחלת את כל הנתונים עבור ה-USRP. היא מקבלת מספר ממשי המייצג את תדר הקךיטה המרכזי, קצב הדגימה, ההגברה וכתובת ה-ip של הרדיו.

13

initStreamer:

הפונקציה מאתחלת את החיבור לקליטה רציפה בזמן אמת.

14

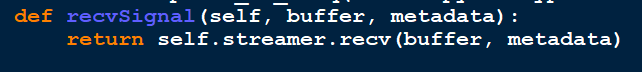
updateFreq:

הפונקציה מקבלת מספר ממשני המייצג את תדר הקליטה החדש ומעדכנת את הרדיו.

15

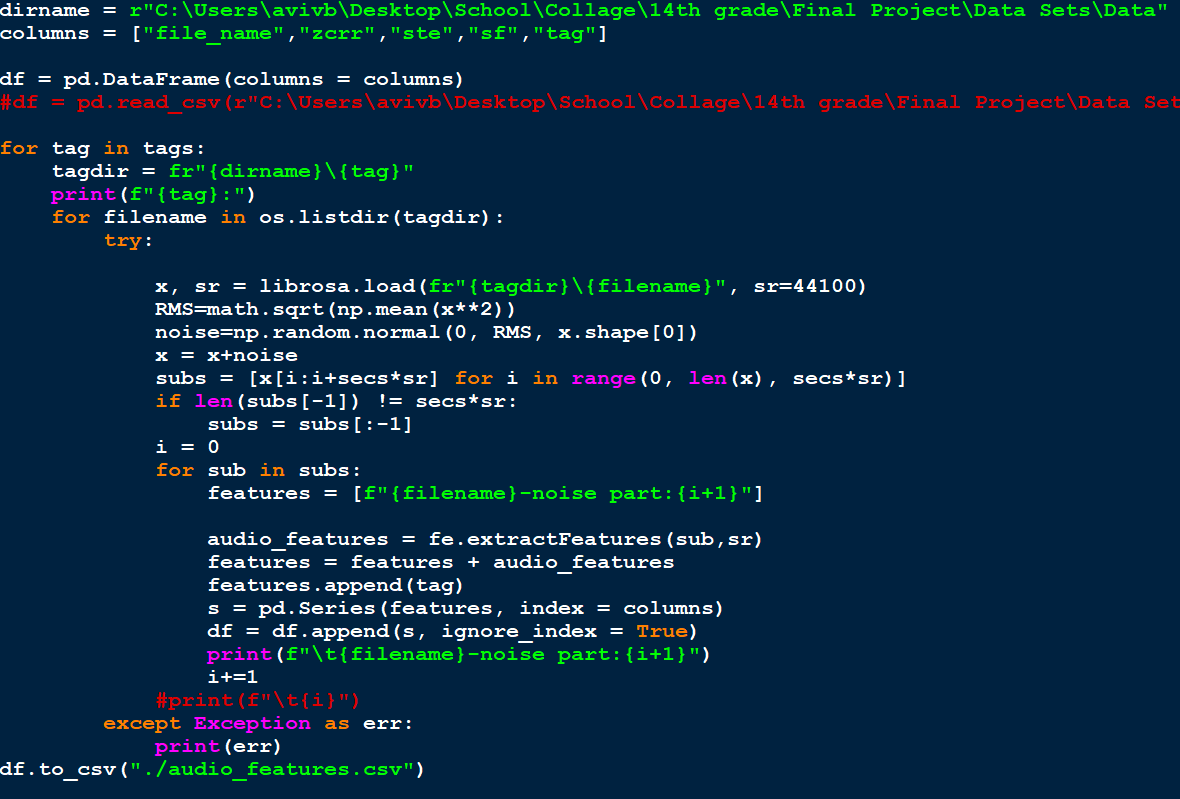
recvSignal:

הפונקציה מקבלת מערך ריק הישמש כ-buffer ו-metadata הנחוץ לקבלת מידע. הפונקציה מכניסה את הנתונים שהתקבלו לתוך ה-buffer.

16

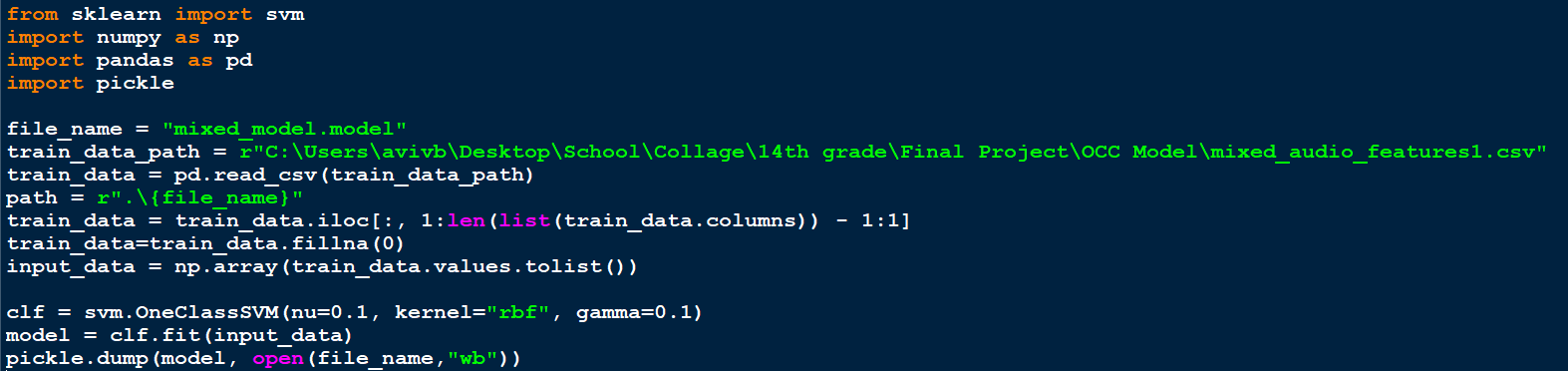
extractFeatures:

פונקציה זו פותחת את קבצי האודיו היועדים ללמיוד המכונה, מוסיפה להם רעש מלאכותי, מוציאה להם את התכונות הרלוונטיות וכותבת אותם לתוך DataFrame. לאחר מכאן הפונקציה מעבירה את ה-DataFrame לקובץ CSV.

17

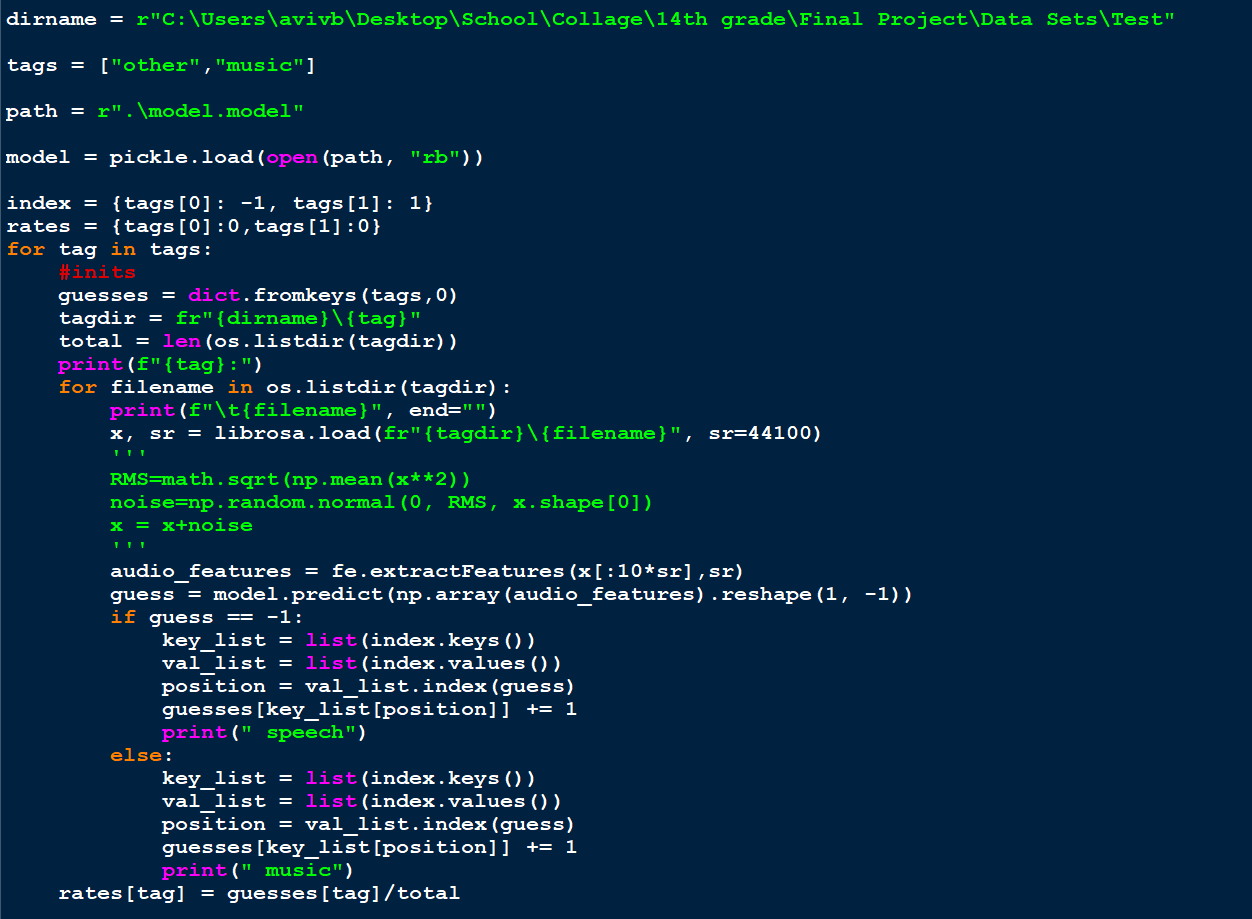
train:

הפונקציה קוראת את קובת ה-CSV, עורכת אותו כדי להיפתר ממידע לא נחוץ (כגון שם הקובץ, מספר סידורי וכדומה) ואז מעבירה את הנתונים לפונקציה של ספריית Sklearn אשר מייצרת לי את המודל. לאחר מכן המודל נשמר כקובץ בינארי (עם סיומת .model) בעזרת ספריית pickle.

18

test\_model:

הפונקציה בודקת את רמת הדיוק של המודל. היא עוברת על ספריית ה-Test במחשב שלי המכילה את קבצי הבדיקה שלי. לאחר מכן היא מעבירה את הגלים דרך פעולת ה-extractFeatures ובודקת את הניחושים על פי ההגיון של מטריצת השגיאה (positve,negative,false postive, false negative) ולבסוף מדפיסה את התוצאות כאחוזי הצלחה.

19

במהלך כתיבת הפרוייקט צצו לי כמה אתגרים, הראשון מביניהם היה לגרום לפונקציות ה-DSP שלי להיות מספיק מהירות כדי שהן לא יפגעו במהירות של שאר התוכנית שלי. תחילה הייתי צריך ליעל את אלגוריתם התמרת פורייה ולהפוך אותו מ-DFT ל-FFT על ידי שיטת הפרד ומשול. בנוסף בחרתי להשתמש בפונקציות מתמטיות של ספריות כמו numpy שכן הספריה פועלת בצורה מהירה ויעילה מאוד. לאחר מכן, אתגר נוסף שצץ לי היה מעבר על ספקטרום ה-FM. במהלך הנסיונות שלי יצא לי מצב שקלטתי בטעות את התחנות פעמיים או שדילגתי על תחנות מסויימות. כדי ךפתור את זה הייתי צריך לבצע קפיצות של 400kHz כדי למנוע מצב של קליטה כפולה. לבסוף, היו לי כמה וכמה אתגרים בנסיון כתיבת המודל כגון בחירת מודל לא נכון, מימוש המודל בצורה לא טובה ובעיות בסינון המידע אך את כולם פתרתי לאחר הסתכלות בדוקומנציה ובדיקות חוזרות.

## 20. מדריך למשתמש

על המשתמש לוודא שמכשיר ה-USRP מחובר לחשמל, אנטנת קליטה ושידור מחוברות, וכבל Ethernet מחובר למחשב בו אנו מריצים את התוכנית, לאחר מכן יש להריץ את קובץ FMScanning.py והתוכנית תתחיל. את הקובץ ניתן להריץ דרך עורך תוכנה (VS Code, IDLE), דרך הקונסול (cmd, terminal) או על ידי לחיצה כפולה על קובץ ההרצה.

## 21. בדיקות והערכה

המערכת מורכבת מכמה שלבים עיקריים אשר זמן הריצה המקסימלי של כל אחד מהם הוא.O(N) אף על פי שהשתמשתי באלגוריתם התמרת פורייה מהיר שיעילותו היא O(NlogN) פעולות אחרות כמו סינון, דימודולציה ודסימציה מצריכות מעבר על כלל הדגימות, כלומר מעבר על כל N האיברים. למרות זאת, ישנם הבדלים בין החלקים השונים שכן ישנו שיפור בקבוע לאחר הדסימציה (במקרה של תוכנית זו קבוע של פי 25 – N/25). כלומר, כל פעולות שליפת המידע מאות האודיו (Feature Extraction) יעשו גם ביעילות ליניארית אך בשיפור בקבוע.

## 22. מסקנות

עבודה על הפרוייקט עזרה לי להבין על הקשיים הקיימים בתרגום העולם הפיזיקלי לעולם התוכנה. מחקר על הפרוייקט נתן לי ידע על תחומים שלא התעסקתי בהם לפני בתחום האותות. בנוסף, העבודה על הפרוייקט עזרה לי להיכנס ולחקור יותר לעומק את עולם למידת המכונה. בעבר חשבתי שלמידת מכונה זה רק דבר אחד – רשת נוירונים – אך עכשיו אני יודע על מגוון רחב של סוגי מודלים, שימושים שונים שלהם והקשר ביניהם לתחומים כמו אלגברה ליניארית, סטטיסטיקה וחדוא. לבסוף, העבודה על הפרוייקט עזרה לי בלמידה על תכנון מערכות, שיטות תכנון שונות (Agile, Top-Down) ועל הצורך בתכנון מעמיק הכולל שרטוטים, טבלאות זרימה ורשימות מטלות כדי לעמוד בזמנים ולעשות מערכת מורכבת.

## 23. פיתוחים עתידיים

בעתיד אוכל להוסיף אפשרות לבחור תחנה ספציפית ולא רק מעבר על כלל הספקטרום. בנוסף, אוכל להוסיף ייצוג גרפי המאפשר למשתמש לראות מידע נוסף כגון ייצוג גרפי של האות בזמן אמת, לראות את הספקטרום בזמן אמת, לראות את אות הקול המשודר בזמן אמת, שמירת תמונת מצב (ייצוג גרפי, אות הקול כקובץ, הדגימות כקובץ) וכדומה.

## 24. ביבליוגרפיה

<https://kb.ettus.com/UHD_Python_API> - UHD API documentation

Kaggel Dataset - <https://www.kaggle.com/datasets/andradaolteanu/gtzan-dataset-music-genre-classification>

Anomaly detection - <https://serokell.io/blog/anomaly-detection-in-machine-learning>

OCSVM theory - <https://www.analyticsvidhya.com/blog/2022/06/one-class-classification-using-support-vector-machines/>

Sklearn documentation - <https://scikit-learn.org/stable/modules/generated/sklearn.svm.OneClassSVM.html>

Confusion matrix - <https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62>

Cross validation - <https://towardsdatascience.com/what-is-cross-validation-60c01f9d9e75>

FFT - <https://en.wikipedia.org/wiki/Fast_Fourier_transform>

DSP (IQ sampling, Modulaiton, Demodulation…) - <https://pysdr.org/content/intro.html>

Speech vs Music classification - <https://www.sciencedirect.com/science/article/pii/S101836391830850X>

Librosa documentation - <https://librosa.org/doc/main/feature.html>

Numpy FFT module documentation - <https://numpy.org/doc/stable/reference/routines.fft.html>

1. הספקטרום האלקטרו-מגנטי. [↑](#footnote-ref-1)
2. דוגמה לדגימה של אות רציף לאות בדיד . [↑](#footnote-ref-2)
3. ייצוג מתמטי של רכיבי ה-I וה-Q. [↑](#footnote-ref-3)
4. ייצוג ויזואלי של ערכי הה-I וה-Q. [↑](#footnote-ref-4)
5. ייצוג מתמטי של אות בעזרת הזהות הטריגונומטרית כאשר a ו-b הם I ו-Q. [↑](#footnote-ref-5)
6. ייצוג כרטזי וקוטבי של מספר מרוכב שהחלק הממשי שלו הוא 2 והחלק המדומה שלו הוא 3. [↑](#footnote-ref-6)
7. ייצוג ויזואלי של היחס בין ייצוג כרטזי וקוטבי. [↑](#footnote-ref-7)
8. ייצוג הציר הממשי והמדוה כ-I ו-Q. [↑](#footnote-ref-8)
9. נוסחאות בשביל למצוא את האמפליטודה והפאזה בעזרת ערכי ה-a וה-b. [↑](#footnote-ref-9)
10. הצבת הנוסחאות (ייצוג בעזרת I ו-Q) לייצוג של אות (אמפליטודה ופאזה). ייצוג של אות בעזרת ערכי I ו-Q. [↑](#footnote-ref-10)
11. ייצוג ויזואלי של התמרת פורייה. [↑](#footnote-ref-11)
12. נוסחה של התמרת פורייה DFT. [↑](#footnote-ref-12)
13. חלוקת תדרי הרדיו בישראל. [↑](#footnote-ref-13)
14. אפנון FM של אות מידע (אדום) עם אות נושא (ירוק). התוצאה היא אות מאופנן (כחול). [↑](#footnote-ref-14)
15. הדגמה ויזואלית של קונבולוציה של ריבוע (אדום) ומלבן (תכלת) כאשר התוצאה היא טרפז (טורקיז). [↑](#footnote-ref-15)
16. Lowpass Filter כפי שהוא נראה מסורטט במרחב התדר. [↑](#footnote-ref-16)
17. ייצוג מתמטי של קונבולוציה. [↑](#footnote-ref-17)
18. נוסחה לדימודולציה. מציאת הפאזה על ידי הכפלת אות באות ההופכי והמנוגד שלו. [↑](#footnote-ref-18)
19. ייצוג מתמטי של דסימציה. [↑](#footnote-ref-19)
20. דוגמה לסוגי חריגים. בשורה העליונה החריג הוא חריג גלובלי (הוא עוף בזמן שכל השאר יונקים) ובשורה התחתונה החריג הוא חריג הקשרי (ביחס לכל השאר הוא שונה). [↑](#footnote-ref-20)
21. חריג גלובלי כפי שנראה על מידע רציף. [↑](#footnote-ref-21)
22. חריג הקשרי כפי שנראה על מידע רציף. [↑](#footnote-ref-22)
23. ייצוג ויזואלי של מודל ה-SVM. [↑](#footnote-ref-23)
24. ייצוג ויזואלי של מודל ה-OCSVM. [↑](#footnote-ref-24)
25. המעבר בין SVM ל-OCSVM. [↑](#footnote-ref-25)
26. סרטוט של ערכי חיתוך האפס עבור כל חלון. [↑](#footnote-ref-26)
27. סרטוט של ערכי האנרגיה עבור כל חלון. [↑](#footnote-ref-27)
28. מודול המערכת. [↑](#footnote-ref-28)
29. מציאת ממוצע האנרגיה של אות. [↑](#footnote-ref-29)
30. ייצוג של cross validation עם ערך k=4. [↑](#footnote-ref-32)
31. הסבר הלוגיקה של מטריצת שגיאה. [↑](#footnote-ref-33)
32. ייצוג ויזואלי של מודל OCSVM עם שוליים (). [↑](#footnote-ref-34)
33. נוסחה מוגבלת למודל ה-OCSVM. [↑](#footnote-ref-35)
34. נוסחה סלחנית לחריגים למודל ה-OCSVM. [↑](#footnote-ref-36)
35. התמרת פורייה של N=8. [↑](#footnote-ref-37)
36. התמרת פורייה של N=4. [↑](#footnote-ref-38)
37. "משוואת הפרפר" – התמרת פורייה של N=2. [↑](#footnote-ref-39)
38. תרשים UML. [↑](#footnote-ref-40)
39. תצוגה גרפית למשתמש. [↑](#footnote-ref-41)