**דוח פרוייקט – מערכת המלצה לשירים**

**תקציר:**

תקציר – מערכת ההמלצה שלנו הינה מערכת המלצה לשירים. בהרבה מקומות (מסעדות, פאבים, ארועים, מועדונים) נדרשת מוזיקה רציפה ברקע שתנוגן באופן אוטומטי. המערכת שלנו תציע פתרון לבעיה הזו בכך שתמליץ על השיר הבא שכדאי לנגן בהנתן N השירים האחרונים שנוגנו. אנו נסתמך על תכונות שונות של שירים כגון – קצב השיר (טמפו), סגנון השיר, מפתח השיר וכו'

דוגמא למקרה – בסט מוזיקה המתנגן בפאב מנוגנים כבר 10 שירים בסגנון ישראלי. המערכת שלנו תזהה את היסטוריית השירים והסגנון שלהם, תחליט על השיר הבא באופן שמתחשב בשירים האחרונים כדי לא לצאת מאיזון ברמה המוזיקלית, אך עם זאת לא לשעמם את הקהל. למשל במקרה הנ"ל המערכת תציע שיר נוסף בסגנון ישראלי אך בזאנר טיפה שונה, או לחילופין מוזיקה דומה לסגנון הישראלי אך לא בעברית (לצורך גיוון).

**תאור האלגוריתם**

האלגוריתם מקבל כקלט מאגר שירים גדול (מעל 10000 שירים). כל שיר מכיל את תכונות כגון שם השיר, שם אמן, קצב, ז'אנר, סולם וכו'. המאגר הינו בפורמט של XML, ואנחנו מבצעים פרסור לכל שיר, שלאחר מכן השיר נשמר באובייקט Song.

לאחר הטעינה, אנו יוצרים רשימת השמעה (Playlist) ריקה, אותה האלגוריתם ימלא בשירים שיומלצו על ידיו. מאחר ובהמלצה הראשונה של האלגוריתם רשימת ההשמעה עדיין ריקה, אנחנו ניבחר שהשיר הראשון שיומלץ יהיה רנדומלי, תוך התחשבות בכמות הפעמים שהשיר נוגן בעבר.(זוהי תכונה של שיר בשם Play Count שנשמרת באובייקט Song).

לאחר ההמלצה הראשונה, האלגוריתם ממליץ על שירים ע"י חישוב דמיון בין שיר מסויים לרשימת ההשמעה הנוכחית. השיר שייבדק אינו שייך לקבוצת השירים ברשימת ההשמעה. עבור כל שיר שנבדק, האלגוריתם מחשב כאמור את הדמיון בינו לבין רשימת ההשמעה. הדמיון מחושב באופן הבא:

דומות בקצב השירים – חישוב ההפרש בין הקצבים, ככל שההפרש קטן יותר הדומות גדלה.

דומות בשם האמן – שירים של אותו אמן יקבלו דומות גדולה.

דומות בסולמות השירים – שירים שנמצאים על אותו סולם יזכו לדומות גדולה.

דומות בסגנון השירים – שירים באותו סגנון (היפ הופ, אלקטורוני וכדומה) יקבלו דומות גדולה.

עבור כל מרכיב – הדומות מנורמלת. לדוג' – הדומות בקצב השירים מנורמלת בטווח המקסימלי האפשרי בין קצבי שירים. הדומות בסגנון השירים מנורמלת בסה"כ הסגנונות שיש לשירים הנוכחיים שנבדקים. (כל שיר יכול להכיל מספר סגנונות).

לאחר שקיבלנו ניקוד עבור כל מרכיב, האלגוריתם משקלל את התוצאות לתוצאה סופית בצורה של ממוצע משוקלל – ככל שמתקדמים ברשימת ההשמעה לכוון השיר האחרון שנוגן, המשקל עולה. אנו יוצאים מנקודת הנחה שדמיון לשירים שנוגנו לאחרונה ברשימת ההשמעה צריכים לקבל משקל רב יותר מאשר דמיון לשירים שנוגנו מזמן.

לבסוף, האלגוריתם מוצא את השיר שקיבל את הניקוד הסופי המקסימלי, ואותו האלגוריתם מחזיר כשיר הבא שינוגן.

פסאודו קוד המתאר את חישוב הדומות:

**public Song Recommend(List<Song> playList)**

{

List data = generateData //contains songs that are not part of the current playlist

if (playlist is empty)

return most played song in data

foreach (song in data)

calcScore(song,playList)

return song with maximum score

}

**private double calcScore(Song song, List<Song> playList)**

{

double score = 0;

int index = 0;

foreach (pSong in playlist)

{

index++;

sim = calcSimilarity(song, pSong);

score += (index \* sim); //the last song that was played get higher weight in score calculation

});

score = score / playList.Count; //normalization

return score;

}

**double calcSimilarity(Song song1, Song song2)**

{

//Calculate BPM Similarity

bpmSim = (1 - song1.bpm-song2.bpm) / 60 ; //60 is the range of the bpm (approx from 70 to 130)

//Calculate Key Similarity

int countSimKey = 0;

foreach (key in song1)

if (song2 contains key)

countSimkey++

double keySim = countSimKey / 3; //normalization

//Calculate Generes Similarity

countSimGenre=0;

foreach (Genere g in song1)

if (song2 contins g)

countSimGenre++;

double genreSim = 0;

if (countSimGenre != 0)

{

genreSim = countSimGenre / (song1.genre.Count + song2.genre.Count - countSimGenre);

}

return genreSim + bpmSim + keySim;