מגישים: שלמה גולאייב, מיכה בריסקמן ומיכאל בסוב

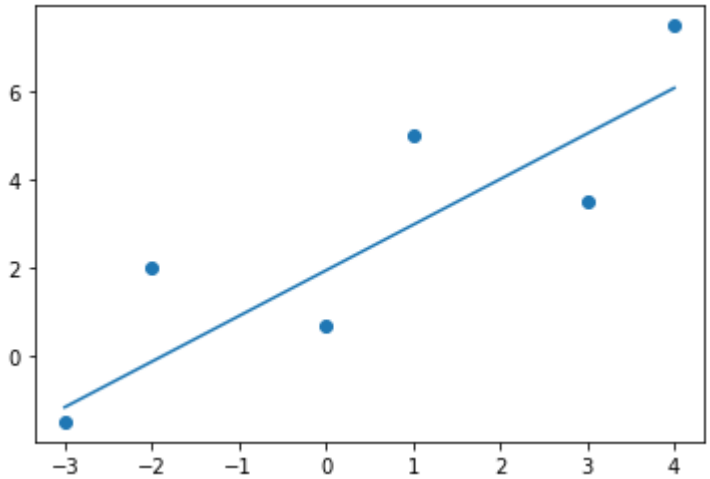
ת.ז. 318757382, 208674713, 315223156

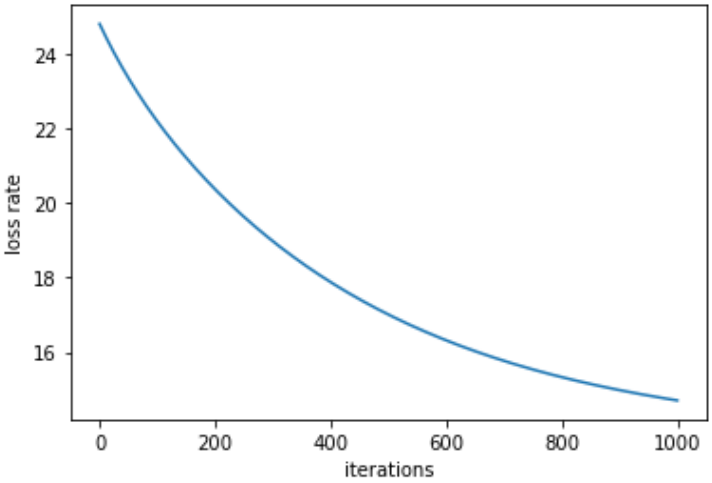
פתרון פרויקט מסכם – מערכות לומדות:

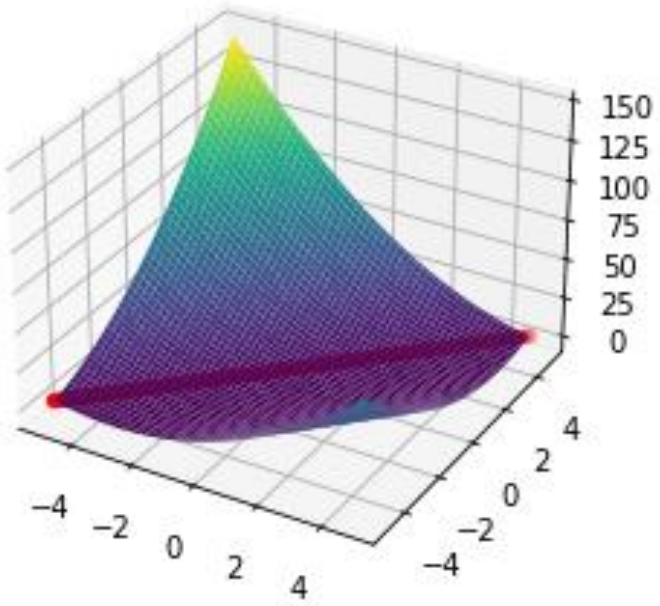
שאלה 1

שאלה 1א:

ערכי הפרמטרים הסופיים שנמצאו הם: a = 1.0361717424613948, b = 1.9367717845926653



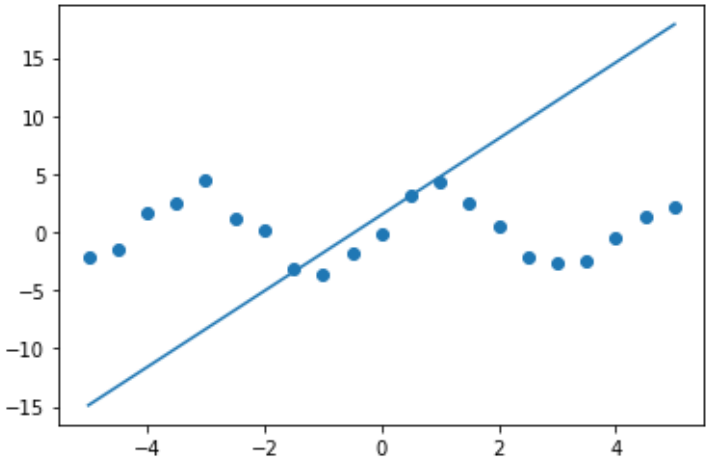


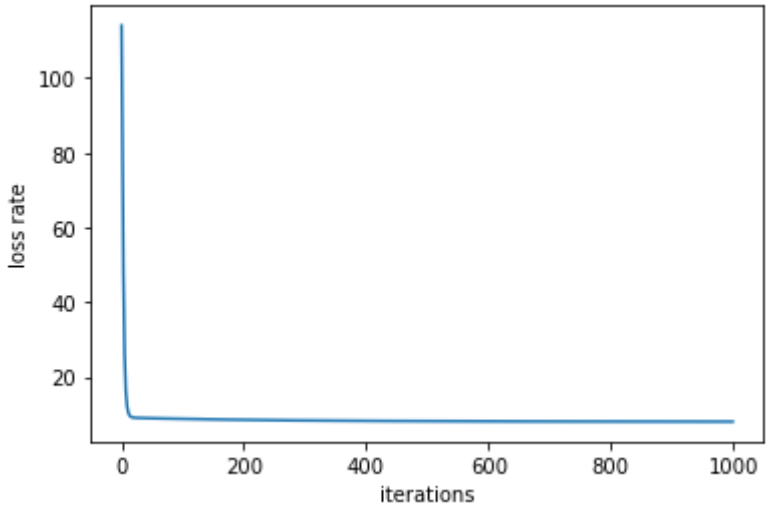


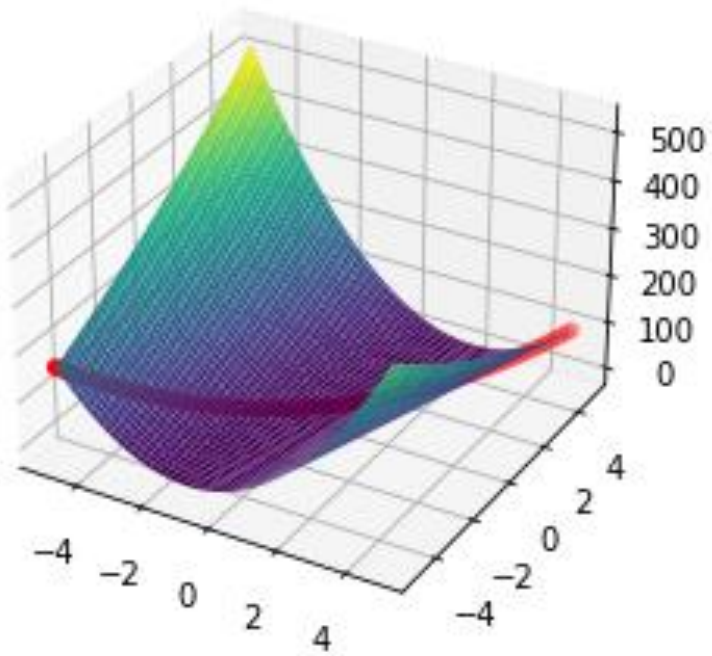
<u>שאלה 1ב:</u>

ערכי הפרמטרים הסופיים שנמצאו הם: a = 3.2756036907353647, b = 1.4895838124109755

קצב הלמידה הוא: 0.0001, מספר האיטרציות הוא: 1000



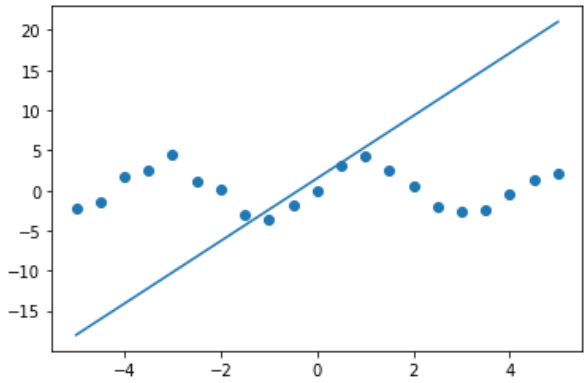


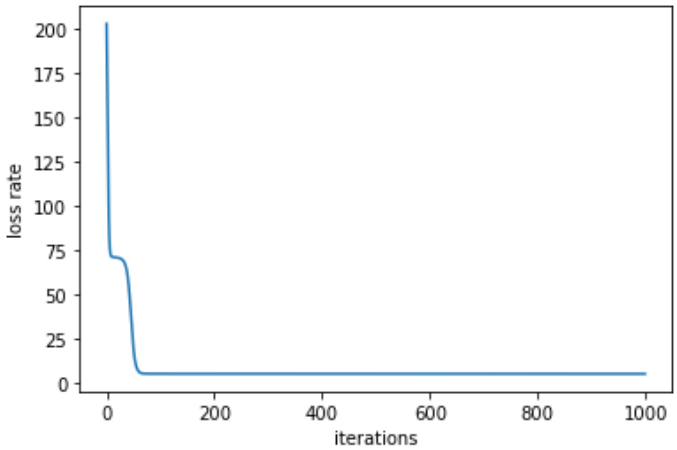


<u>שאלה 1ד:</u>

ערכי הפרמטרים הסופיים שנמצאו הם: a = 3.318145436025057, b = 1.488977981749276

שאלה 1ה:





הפרמטרים שנמצאו:

a = 3.906473499047692, b = 1.503704120940236, c = 0.1826230927089416

נראה כי הפרמטרים שנמצאו בסעיף זה דומים לפרמטרים הנמצאו בסעיף קודם, ניתן לראות כי ערכי a,b קרובים מאוד לערכים בסעיף ד' אך מתווסף פרמטר c שנראה שמוסיף/מאזן את הנתונים

בסעיף זה למודל לקח 73.32 מילישניות למצוא את הפרמטרים

בסעיף ד' למודל לקח 1.01 מילישניות למצוא את הפרמטרים

שאלה 2

א. נירמול המטריצות 8X28) המייצגות ספרות, חילוק לסט אימון וסט מבחן

ב. עברנו על אלגוריתמי האשכול שיש באתר Sickit-learn והשוונו בניהם:

Method name	Parameters	Scalability	Usecase	Geometry (metric used)
K-Means	number of clusters	Very large `n_samples`, medium `n_clusters` with MiniBatch code	General-purpose, even cluster size, flat geometry, not too many clusters, inductive	Distances between points
Affinity propagation	damping, sample preference	Not scalable with n_samples	Many clusters, uneven cluster size, non-flat geometry, inductive	Graph distance (e.g. nearest-neighbor graph)
Mean-shift	bandwidth	Not scalable with `n_samples`	Many clusters, uneven cluster size, non-flat geometry, inductive	Distances between points
Spectral clustering	number of clusters	Medium `n_samples`, small `n_clusters`	Few clusters, even cluster size, non-flat geometry, transductive	Graph distance (e.g. nearest-neighbor graph)
Ward hierarchical clustering	number of clusters or distance threshold	Large `n_samples` and `n_clusters`	Many clusters, possibly connectivity constraints, transductive	Distances between points
Agglomerative clustering	number of clusters or distance threshold, linkage type, distance	Large `n_samples` and `n_clusters`	Many clusters, possibly connectivity constraints, non Euclidean distances, transductive	Any pairwise distance
DBSCAN	neighborhood size	Very large `n_samples`, medium `n_clusters`	Non-flat geometry, uneven cluster sizes, outlier removal, transductive	Distances between nearest points
OPTICS	minimum cluster membership	Very large `n_samples`, large `n_clusters`	Non-flat geometry, uneven cluster sizes, variable cluster density, outlier removal, transductive	Distances between points
Gaussian mixtures	many	Not scalable	Flat geometry, good for density estimation, inductive	Mahalanobis distances to centers
BIRCH	branching factor, threshold, optional global clusterer.	Large `n_clusters` and `n_samples`	Large dataset, outlier removal, data reduction, inductive	Euclidean distance between points
Bisecting K-Means	number of clusters	Very large `n_samples`, medium `n_clusters`	General-purpose, even cluster size, flat geometry, no empty clusters, inductive, hierarchical	Distances between points

סינון ראשוני היה להתמקד באלגוריתמים שעובדים עם מספר אשכולות, כמטרת התרגיל ולכן נשארנו עם שלושה אלגוריתמים:

K-Means, Spectral clustering, MiniBatch kmeans

על מנת לבחור את האלגוריתם האופטימלי בשבילנו, השוונו בין שלושת הנ"ל וראינו ש-MiniBatch יתאים בצורה הטובה ביותר מבחינת גודל סט הנתונים.

ג. לכל מחלקה יש features שונים במספר ולכן אם נשתמש באותו מספר אשכולות, נגלה בהמשך שהמודל מתקשה לזהות ולסווג נכונה מספרים אשר יש להם מספר אפשרויות כתיבה (לדוגמא הספרה 4), ספרות שיש להן יותר מצורת כתיבה אחת או כמה מאפיינים שדומים לספרות אחרות יהיה צריך להוסיף יותר אשכולות על מנת לנסות ולמצוא את כל מאפייני הספרה

ד. מכל אשכול נוכל לאגור labels המייצגים את המחלקה, אנחנו מעוניינים בהם כי באשכול על סט המבחן, המידע מהאשכולות על כל מחלקה יעזור למודל ללמוד, בנוסף, אשכול על כל מחלקה נותן את מספר האשכולות שנמצאו לאותה מחלקה, על מנת לבצע סיווג כמה שיותר טוב ניקח את סכום האשכולות וזה יהיה מספר האשכולות על סט האימון

ה. הלאגוריתם מתבסס על הרצת אשכול על כל ספרה בנפרד ובאמצעות זה למצוא את מספר האשכולות שצריך על מנת לסווג בצורה הטובה ביותר את הספרות (באלגוריתם שלנו הכפלנו את המספר הזה בשלוש, מצאנו כי כך הוא מגיע לתוצאות טובות בזמן הגיוני) ומאמנים את המודל, לאחר האימון נותנים למודל להיבחן על סט המבחן, לאחר מכן, מכל אשכול אנחנו מוצאים את המספר הכי שכיח ומניחים שהספרה הזו מייצגת את האשכול הזה ושומרים במילון שבו המפתחות הן 10 הספרות והערכים הם מערכים הכוללים את מספרי האשכול שמצאנו שמייצגים את אותה הספרה. לבסוף אנו עוברים על התוצאות שהניב המודל מסט המבחן ומשווים לערכים הנמצאים במילון ולפי כך קובעים אם המודל סיווג נכונה את המספר או לא.

ו. במהלך האימון, לעקוב על האשכול ולראות ויזואלית האם הוא מסווג נכון, ניתן באמצעות גרפים, הדפסת תוצאות או לקחת מדגמית

ז. על מנת למצוא את המספר אשכולות האופטימלי השתמשנו אלגוריתם k-מרכזים והצגנו את הגרף היוצא לכל ספרה

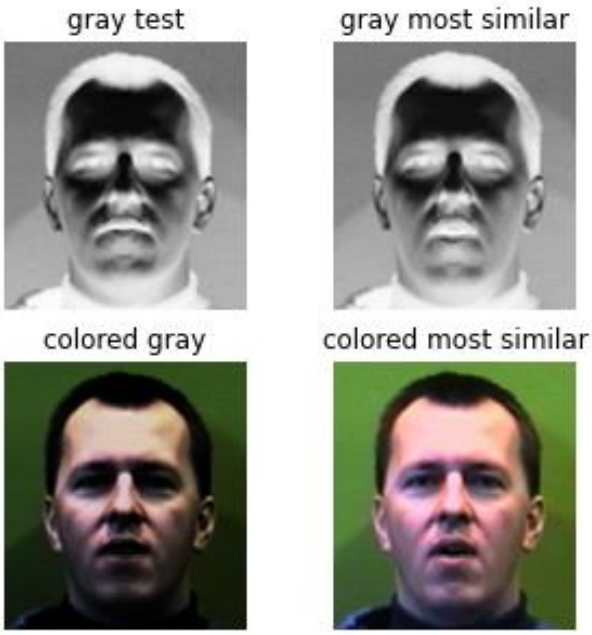
ח. חילקנו את סט הנתונים לסט אימון וסט מבחן בצורה יחסית שווה, ראינו כי כשסט המבחן גדול יותר, המערכת מגיעה לתוצאות טובות יותר הן מבחינת זמן והן מבחינת אחוזי דיוק, לבסוף חילקנו את סט הנתונים חצי חצי אך קיבלנו תוצאות דומות (קצת פחות טובות) בסט המבחן היה בגודל 0.6 מסט הנתונים

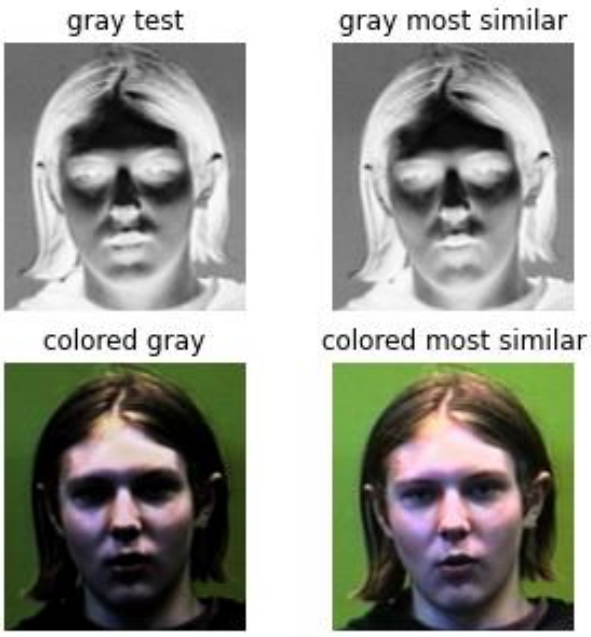
ט. ראינו ששינויים של מספר האשכולות וגודל סט האימון משפיעים מאוד על התוצאות, בעזרת אלגוריתם k-מרכזים הצלחנו למצוא מספר אשכולות למודל שמפיק תוצאה מספקת (של כ-83 אחוזים) אך ראינו שכשהגדלנו את המספר המתקבל פי 3 קיבלנו תוצאות טובות יותר (כ-92 אחוזים) ולכן יש לחשוב על אלגוריתם / מודל עזר שיכול לעזור למצוא את גודל האשכולות היעיל ביותר שיביא לתוצאות טובות בזמן ריצה הגיוני.

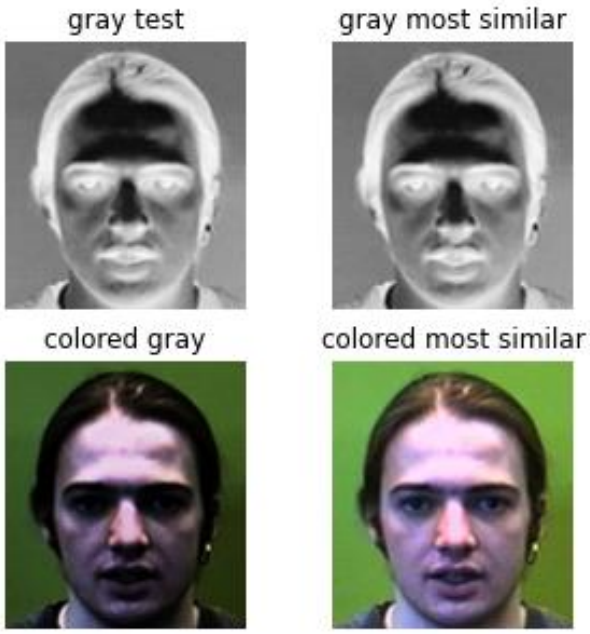
שאלה 3

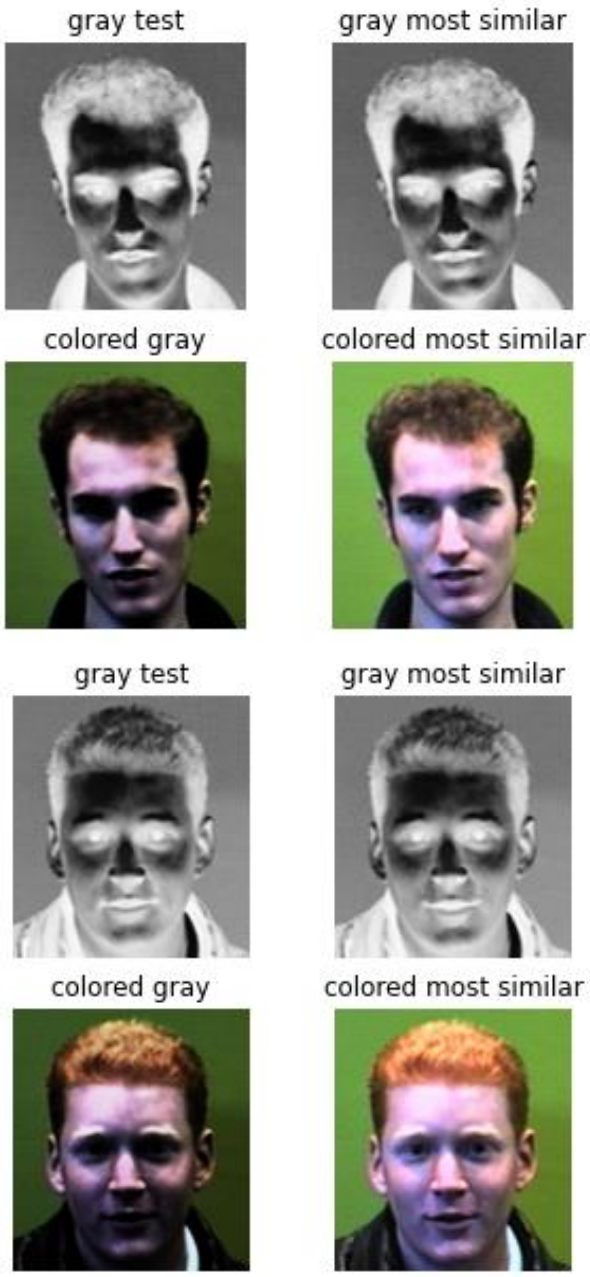
השיטה שלנו משתמשת בהכפלת מטריצות לצורך השלמת הצבע.
אנחנו לוקחים את התמונה הצבעונית שנמצאה כתמונה הדומה ביותר לתמונה האפורה מסט המבחן ומפצלים את את התמונה לשלושת מרכיבי הצבע (RGB). כל מערך של צבע בודד אנחנו מכפילים בתמונה האפורה. לבסוף אנחנו ממזגים את שלושת המערכים החדשים שקיבלנו ויוצרים את התמונה הצבעונית.

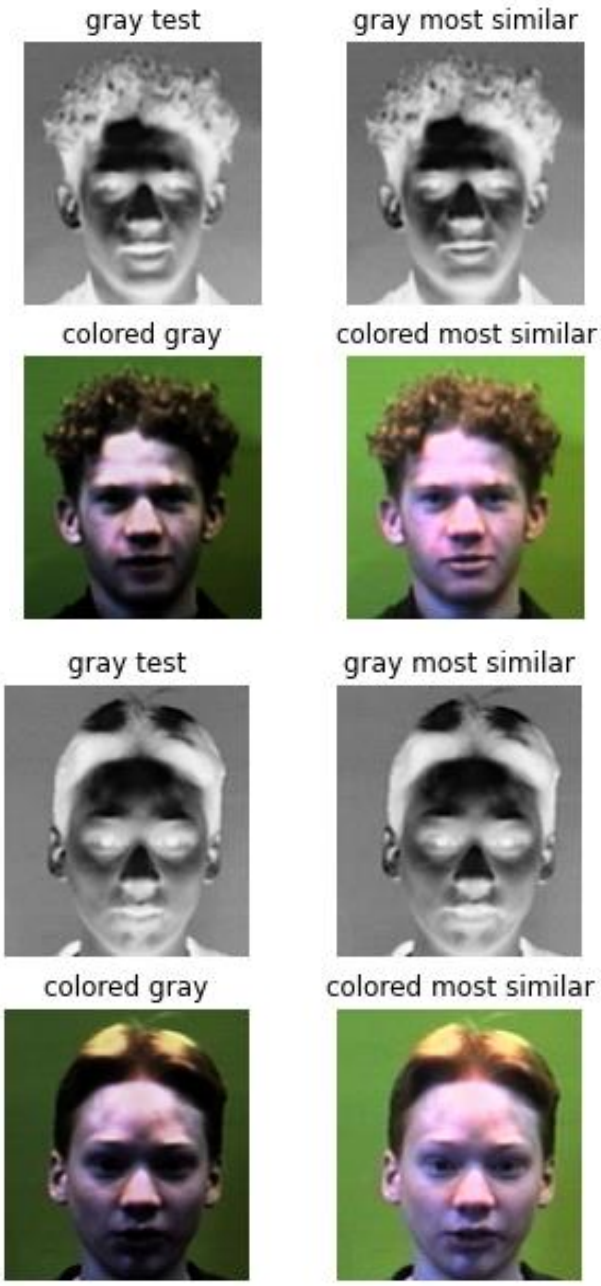
א. התמונה הדומה ביותר:



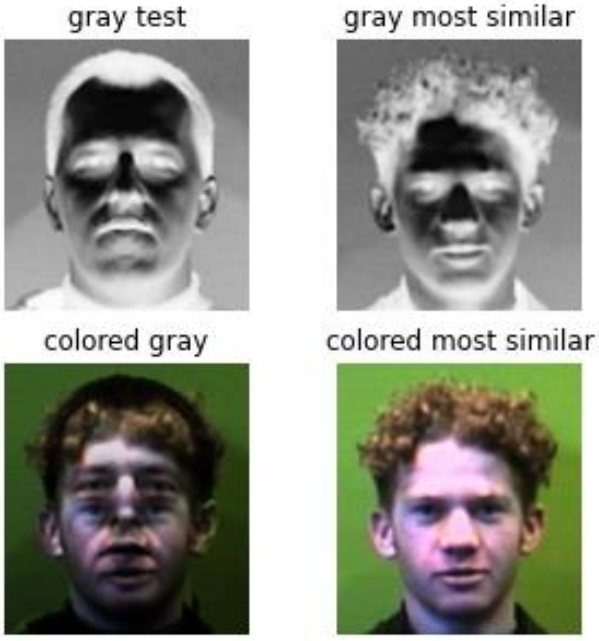

gray test
gray most similar
colored gray
colored most similar

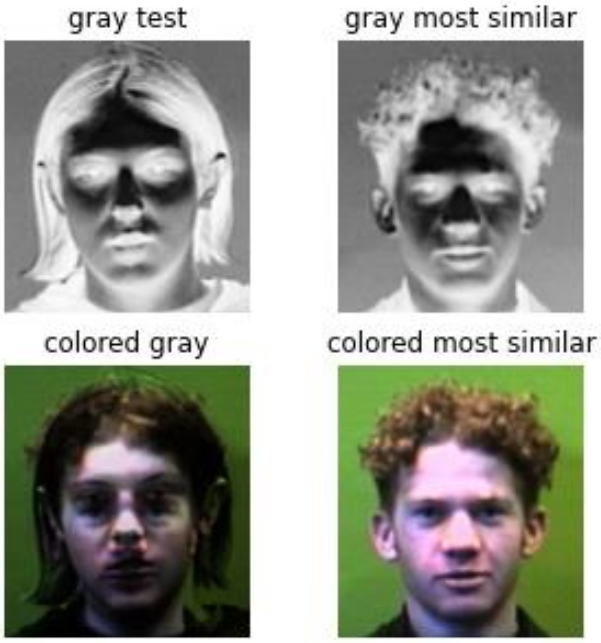

gray test
gray most similar
colored gray
colored most similar

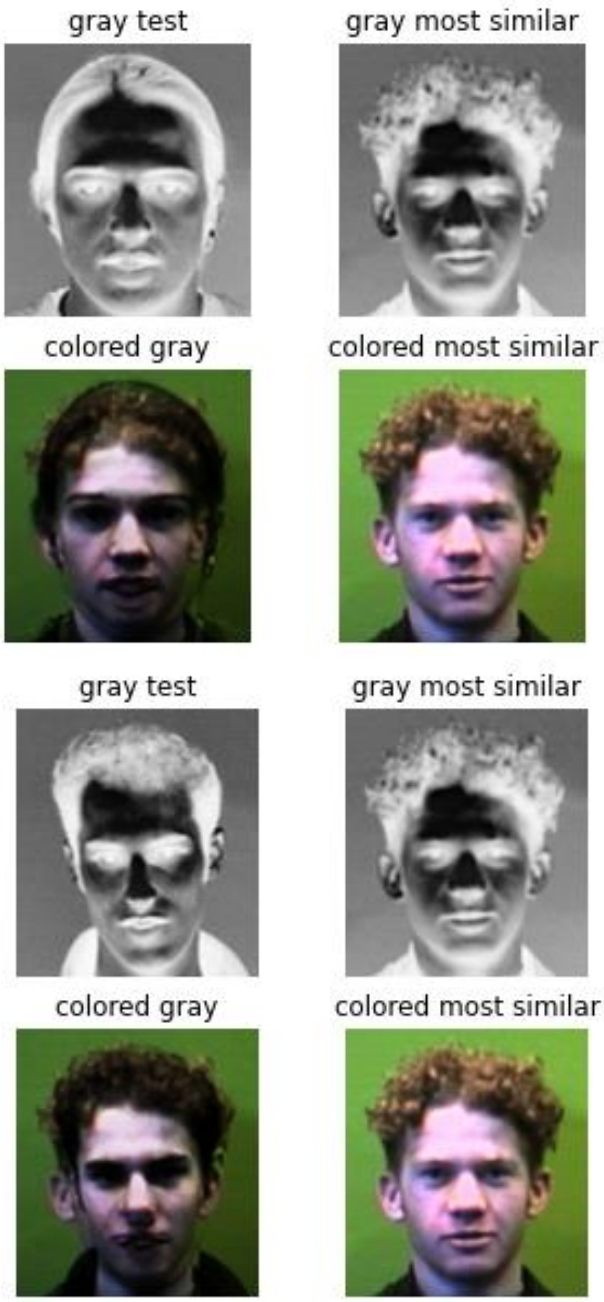

gray test
gray most similar
colored gray
colored most similar
gray test
gray most similar
colored gray
colored most similar

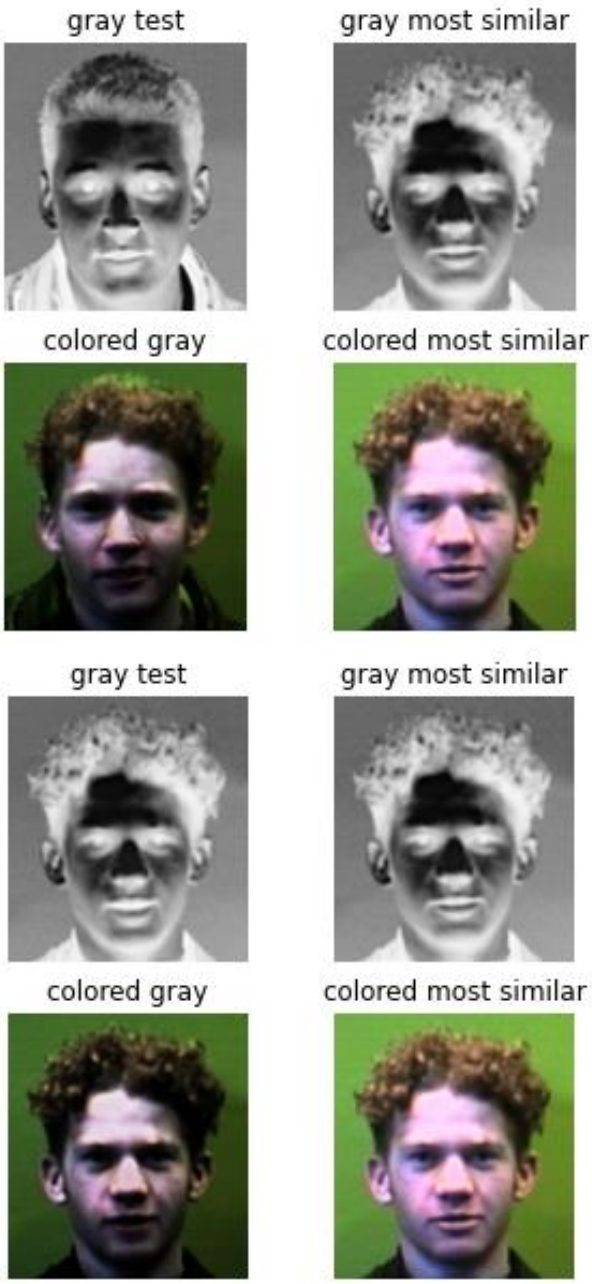

gray test
gray most similar
colored gray
colored most similar
gray test
gray most similar
colored gray
colored most similar

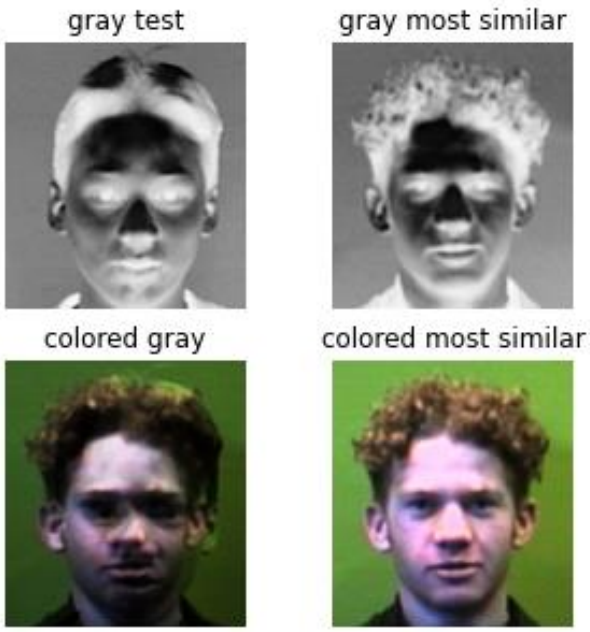
התמונה הממוצעת:






gray test
gray most similar
colored gray
colored most similar
gray test
gray most similar
colored gray
colored most similar


gray test
gray most similar
colored gray
colored most similar
gray test
gray most similar
colored gray
colored most similar



ב. השיטה נותנת תוצאות טובות יותר כאשר התמונות יותר דומות, משום שכך יותר קל להשלים את התמונה לתמונה המקורית ויש פחות "פספוסים". התוצאות הטובות ביותר כמובן נוצרות כאשר השלמת הצבע מתבצעת ע"י אותו פרצוף (אפילו אם הבעת הפנים שונה)

ג. התמונה הממוצעת הינה התמונה שבה תוצאת ה PCA היא הממוצעת, ולכן זו תמונה אחת שאינה משתנה. הבעיה בכך שכאשר אנחנו משווים את התמונה של איש אחד לתמונה של איש אחר, תווי הפנים והצבעים שונים, לכן הדרך הזו פחות טובה. ניתן לראות שקיבלנו לפעמים הכלאה בין שני הפרצופים בגלל שהתכונות שלהם שונות, אבל כשהפרצופים קצת דומים יש תוצאות סבירות.
לעומת זאת כאשר אנחנו משתמשים בתמונה הדומה ביותר, אנחנו מקבלים תמונות יחסית דומות, של אותם אנשים (כלומר תמונת המבחן היא של אותו אדם כמו התמונה הדומה ביותר). ולכן התוצאות טובות יותר משמעותית.

ד. הצעה לשיפור:
אלגוריתם אחר שחשבנו עליו היה לקחת את התמונה הדומה ביותר שמצאנו, להמיר אותה לתמונה רמת אפור. השלב הבא הוא לקחת את התמונה האפורה של סט המבחן ולעבור פיקסל פיקסל, עבור כל פיקסל לחפש את הפיקסל הכי דומה לו בתמונה האפורה החדשה שיצרנו (התמונה הדומה ביותר שהמרנו לאפורה). כשמצאנו את הפיקסל הזה, לגשת לתמונה המקורית הצבעונית (הדומה ביותר) ולהעתיק את הפיקסל הזה לתמונה האפורה מסט המבחן. בעזרת שיטה זאת נקבל תוצאות טובות יותר גם בשימוש בתמונה הממוצעת.

ה.
https://towardsdatascience.com/image-compression-using-principal-component-analysis-pca-253f26740a9f

https://towardsdatascience.com/rgb-color-image-compression-using-principal-component-analysis-fce3f48dfdd0

שאלה 4

א. קיבלנו במשימה 6 Wav files, כאשר כולם באותו אורך כבערך 6 שניות, כ50,000 אמפליטודות ותדירות של 8000. השתמשנו בכל 6 הקבצים לצורך הסיווג.

בחרנו לסווג את אותות הקול בצורה הבאה:

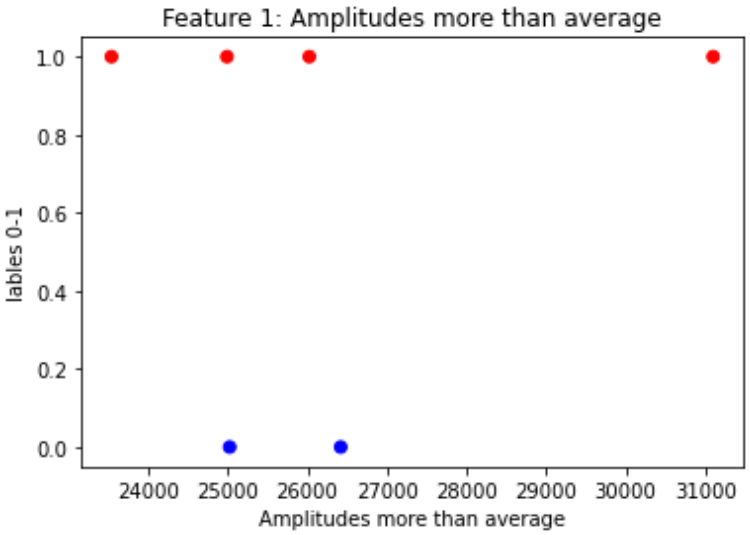
A non-speaking signal – 0
A speaking signal – 1

כלומר Soucre1 וSource5 יתוייגו 0, וכל שאר 4 האותות יתוייגו 1.

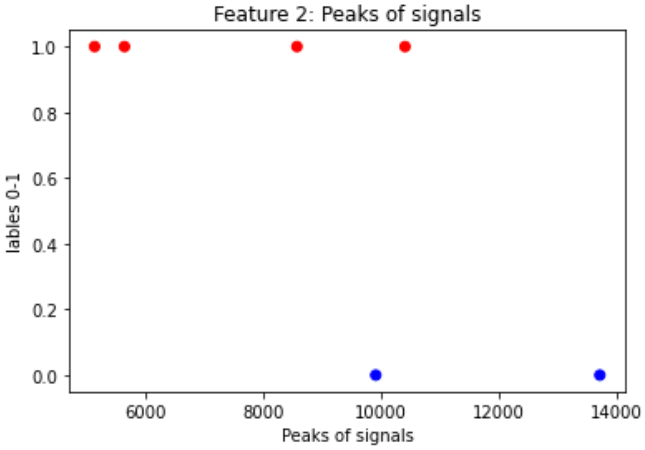
התיוגים שלנו הם: [0, 1, 1, 1, 0, 1] לפי סדר האותות.

בחרנו 4 Features:

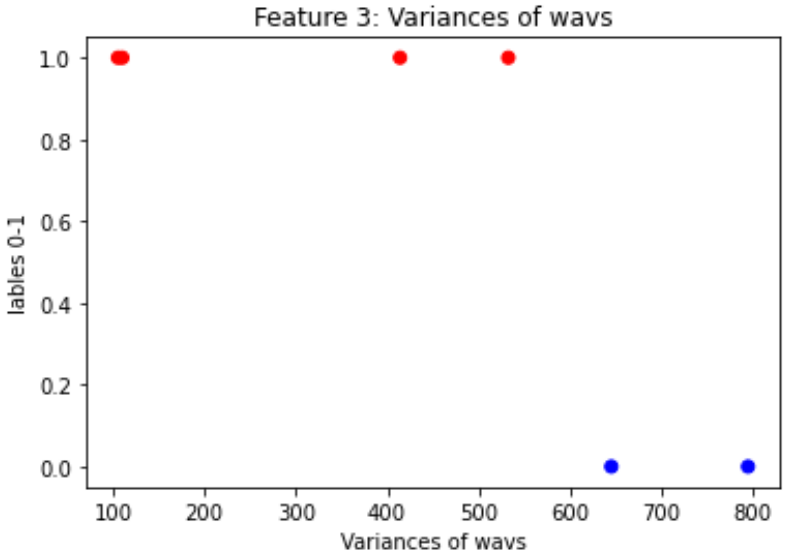
1. Feature המחזיר סקלאר כמה אמפליטודות נמצאות מעל אמפליטודת הממוצע של אות הקול.



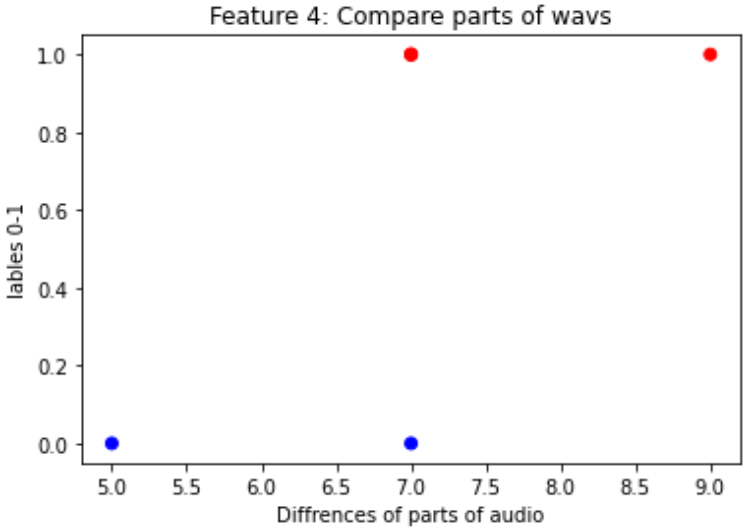
2. Feature המחזיר סקלאר כמה נק' מקסימום יש באות הקול.



3. Feature המחזיר סקלאר שהוא השונות של אות הקול.



4. Feature המשווה בין שניה לשניה באות הקול ומחשב אם סכום האמפליטודות באותה השניה גדול מסכום האמפליטודות בשניה הבאה, הוא מחזיר סקלאר את כמות הפעמים שהתנאי מתקיים.



חלקנו את 6 האותות המקוריים ל 3 אותות של סט אימון ו3 אותות של סט מבחן. 3 האותות הראשונים הם סט האימון ו3 אותות האחרונים הם של סט המבחן ככה שבכל סט יש תיוג של 0 ו-1.

התיוגים שלנו הם: [0, 1, 1, 1, 0, 1] לפי סדר האותות.

y_test	Array of int32	(3,)	[1 0 1]
y_train	Array of int32	(3,)	[0 1 1]

המרנו למרחב מאפיינים והשתמשנו במסווג של logistic regression לזיהוי האותות.

קיבלנו את התוצאות הבאות:

```
Logistic regression using [countMoreThanAvg, countPeaks, calcVarience, comparePartsOfWav] features:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         1
           1       1.00      1.00      1.00         2

    accuracy                           1.00         3
   macro avg       1.00      1.00      1.00         3
weighted avg       1.00      1.00      1.00         3


Confusion matrix:
[[1 0]
 [0 2]]
```

הרצנו את התוכנית שלנו 10 פעמים ובכל 10 הפעמים המסווג סיווג נכון את האותות שלנו, כלומר הfeatures שבחרנו הם טובים למשימת הסיווג. מנגד, זאת לא חוכמה כי כמות הנתונים קטנה וכמות התיוגים קטנה לכן סיכוי גבוה לסיווג נכון בכל פעם.

סעיף י'

לאחר מכן ניסינו לסווג את 6 האותות המשוחזרים.

דבר ראשון שעשינו הוא לזהות איזה אות משוחזר מתאים לאות המקורי על מנת לתייג את האותות המשוחזרים. הרי האותות המשוחזרים אינם בסדר הנכון ואינם בscale הנכון.

בהתחלה ניסינו לזהות את הדמיון בין כל אות מקורי לאות משוחזר לפי מרחקים אוקלידים. לכל אות מקורי חפשנו את המרחק מהאות המשוחזר, כלומר הפעלנו את הפונקציה 6 פעמים לכל אות מקורי. השתמשנו בפונקציה הבאה:

```
euclidean_dist = np.linalg.norm(original_signal - recontructed_signal)
```

כדי לוודא שהתכנית באמת מצליחה לזהות נכון את האותות השתמשנו בהשוואה וויזואלית שלנו. גילינו שישנם אותות שחוזרים על עצמם כלומר המרחקים שלהם לאות המקורי רחוקים יותר מאותות אחרים. ופונקציה זאת לא צלחה את הזיהוי הנכון.

לאחר מכן ניסינו לזהות לפי פונקציית correlation של numpy array הבאה:

```
correl = np.corrcoef(original_signal, recontructed_signal)[0,1]
```

אך גם כאן היה זיהוי לא נכון של האותות, והייתה חזרה של האותות, כלומר התכנית זיהתה את האותות המשוחזרים 2 ו-5 כאות 2 המקורי. (ניסינו לחפש מהו הcorrelation הכי גבוהה מתוך כל הcorrelations של אות מסוים). גם כאן לכל אות הפעלנו 6 פעמים את הפונקציה.

לאחר מכן ניסינו לזהות לפי אחד הfeatures שבנינו: Feature המחזיר סקלאר כמה נק' מקסימום יש באות. חפשנו איפה קיים המרחק הכי קטן בין כמות הנק' המקסימום. אך גם כאן הייתה חזרתיות, ולא תמיד הייתה התאמה נכונה לאות המקורי.

על מנת שהתכנית תרוץ בלי הפרעות של המשתמש בסוף בחרנו להשתמש בפונקציית ה- correlate לזיהוי כל אות. הזיהוי לא תמיד היה נכון, אך לא הייתה חזרתיות של אותות. הזיהוי היה נכון ב4 מתוך 6 האותות.

השתמשנו בפונקציה הבאה:

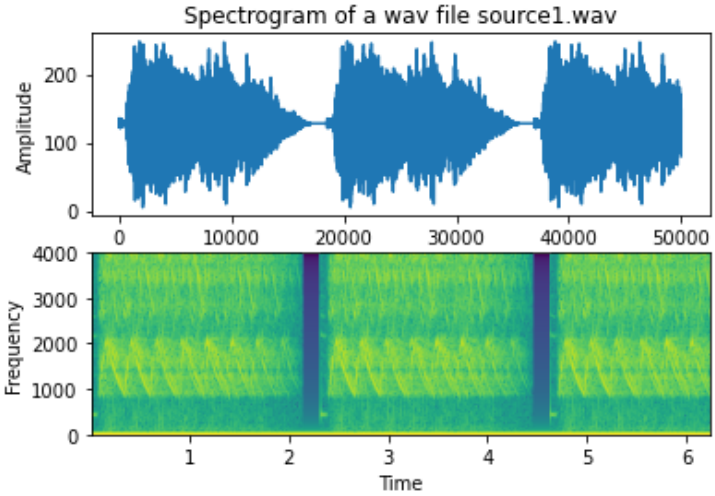
```
from scipy import signal
```

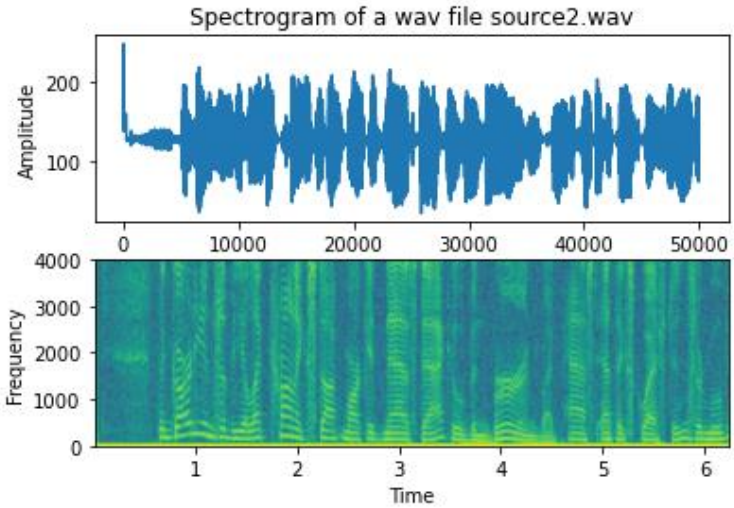
```
corr = signal.correlate(signal_1, recontructed_signal, mode='same')
```

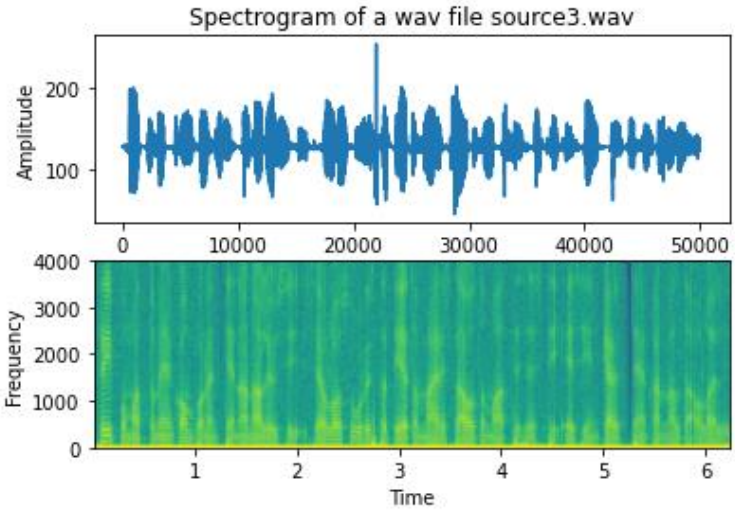
לכל אות בצענו את הפונק' 6 פעמים, ולאחר מכן בדקנו מהו ה-corr הכי מתאים לאות המקורי.

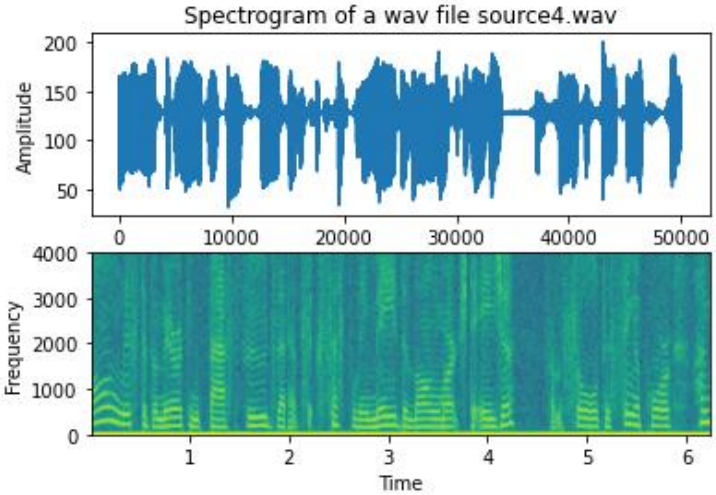
לטובת סיווג המשימה זיהינו וויזואלית ע"י הספקטוגרמות למי כל אות משוחזר מתאים לאות המקורי ובכך נתנו לכל אות משוחזר את התיוג הנכון.

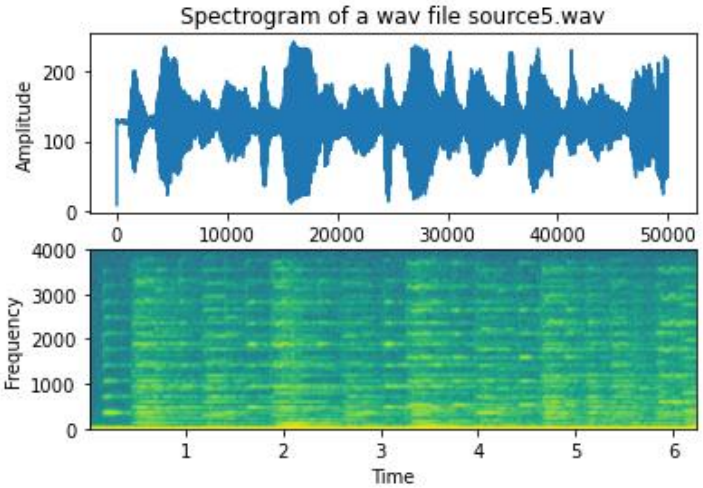
ספקטוגרמות של האותות המקוריים:

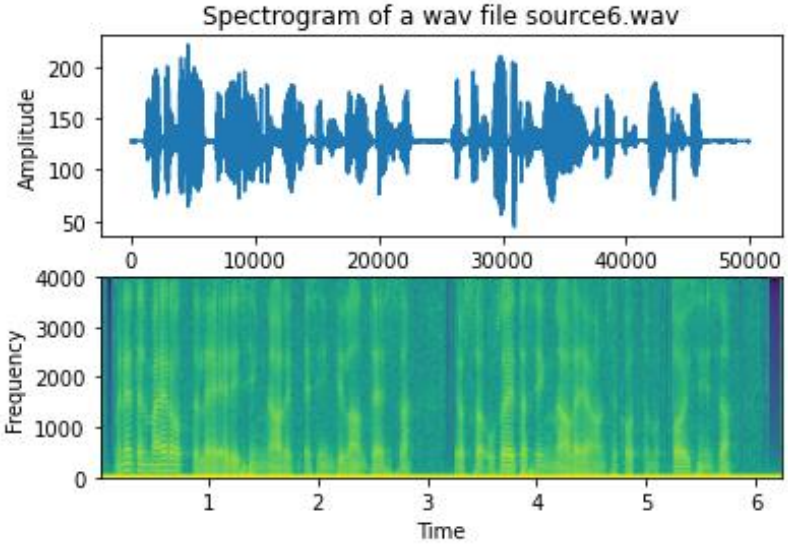




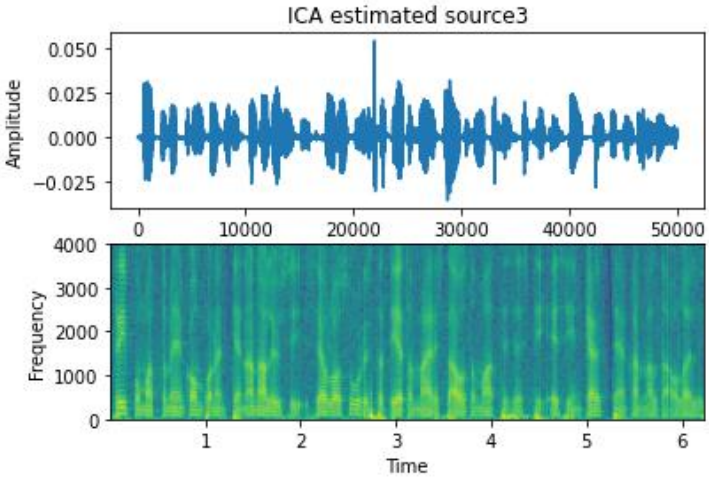

Spectrogram of a wav file source3.wav
Amplitude
200
100
0
10000
20000
30000
40000
50000
Frequency
4000
3000
2000
1000
0
1
2
3
4
5
6
Time

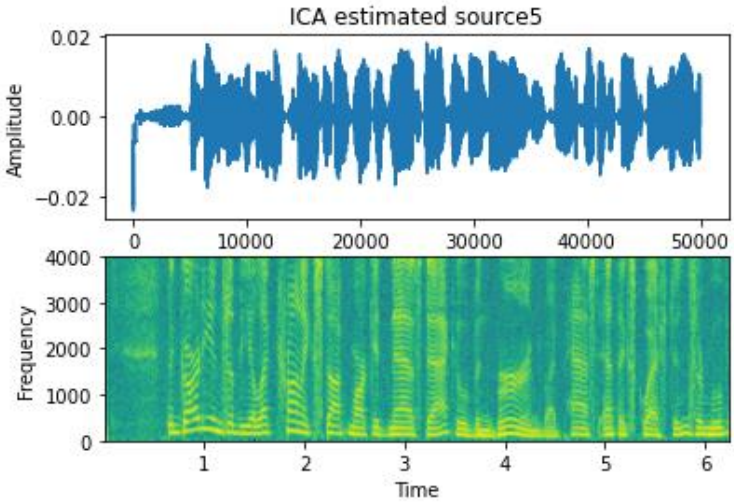

Spectrogram of a wav file source4.wav
Amplitude
200
150
100
50
0
10000
20000
30000
40000
50000
Frequency
4000
3000
2000
1000
0
1
2
3
4
5
6
Time

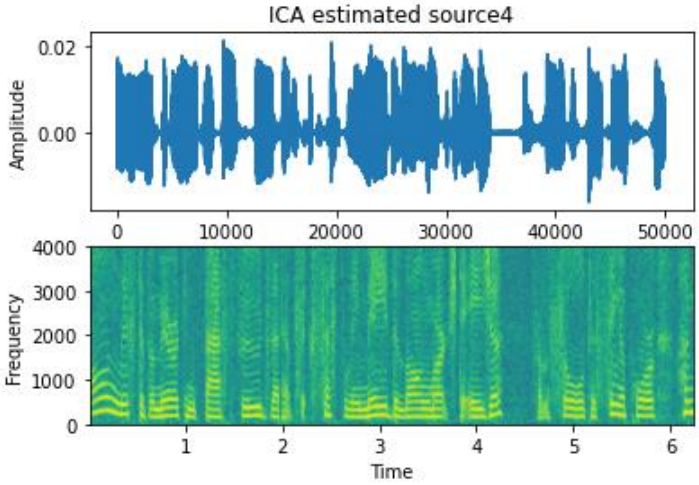


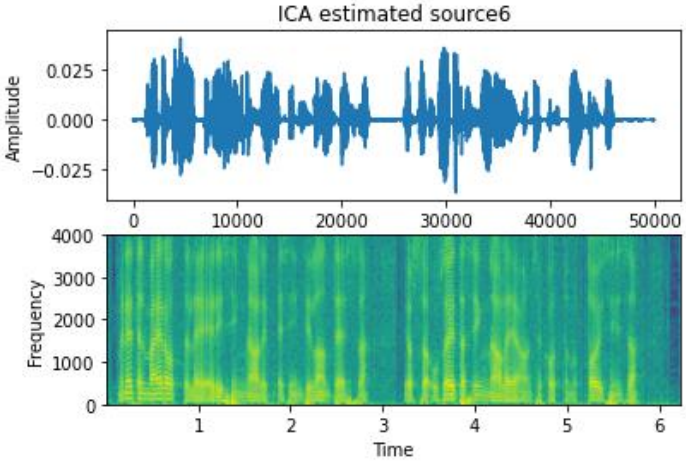


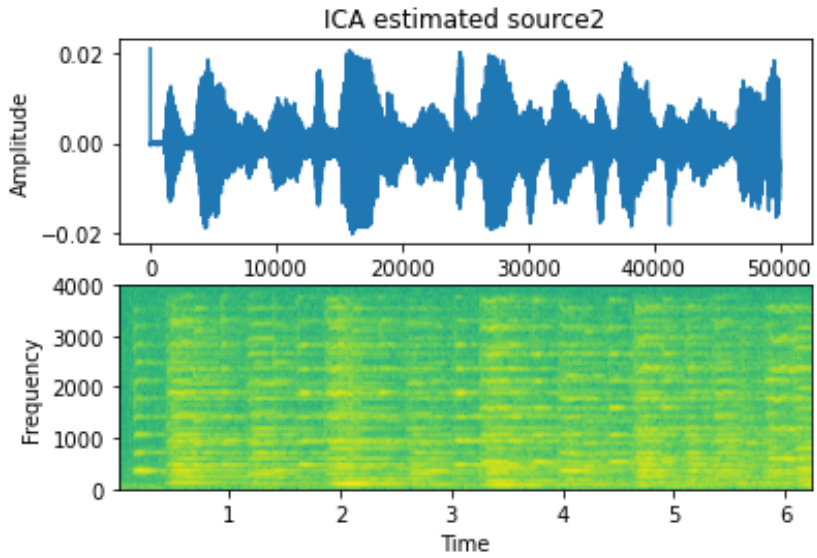
<u>ספקטוגרמות של האותות המשוחזרים עם הזיהוי של האות מתאים לאות המקורי</u>:

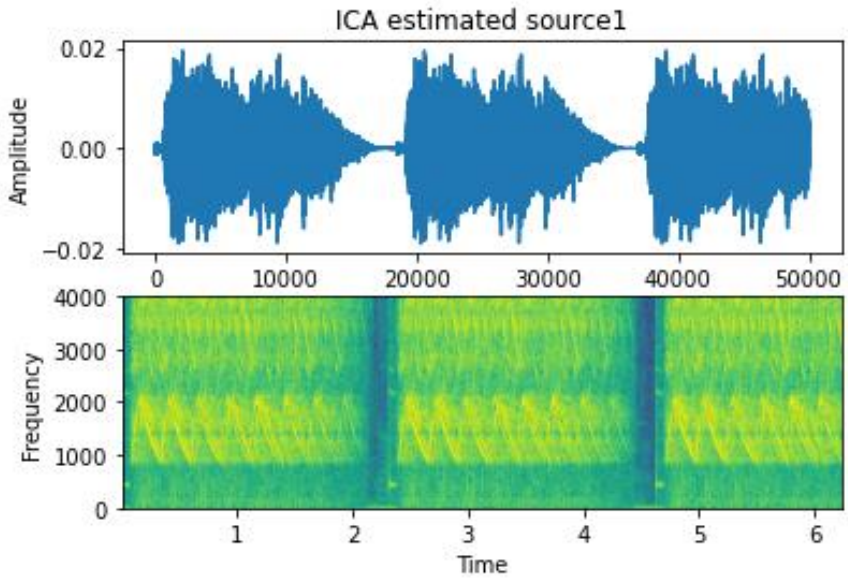



ICA estimated source5
Amplitude
0.02
0.00
−0.02
0
10000
20000
30000
40000
50000
Frequency
4000
3000
2000
1000
0
1
2
3
4
5
6
Time


ICA estimated source4
Amplitude
0.02
0.00
0
10000
20000
30000
40000
50000
Frequency
4000
3000
2000
1000
0
1
2
3
4
5
6
Time


ICA estimated source6
Amplitude
0.025
0.000
−0.025
0
10000
20000
30000
40000
50000
Frequency
4000
3000
2000
1000
0
1
2
3
4
5
6
Time





<u>סעיף ג':</u> וויזואלית ICA מצליח להפריד את האותות כראוי בחלק מהמצבים. בחלק מהרצות הוא לא מצליח להפריד כראוי לפעמים היו 2 ספקטוגרמות מאוד דומות אחת לשניה. בנוסף, כאשר ניסינו לחשוב על שיטה על זיהוי האותות המשוחזרים לאותות המקוריים לא צלחנו בזה ע"י חוקיות כלשהי וניסינו בעצמינו לראות את הדמיון (וויזואלית) ולתת את התיוג הנכון.

הפעלנו את המסווג שפתחנו בסעיף הקודם על האותות המשוחזרים וקיבלנו את התוצאות הבאות:

```
Logistic regression using [countMoreThanAvg, countPeaks, calcVarience, comparePartsOfWav] features:
              precision    recall  f1-score   support

         0.0       1.00      0.50      0.67         2
         1.0       0.80      1.00      0.89         4

    accuracy                           0.83         6
   macro avg       0.90      0.75      0.78         6
weighted avg       0.87      0.83      0.81         6


Confusion matrix:
[[1 1]
 [0 4]]
```

נראה כי מתוך 6 האותות רק אות אחד לא סווג נכון.

לדעתנו המאפיין של חישוב השונות הוא בעייתי מכיוון שהשונות באותות המשוחזרים הוא מאוד קטן ודומה אחד לשני כבערך $2*10^{-5}$, לעומת השונות של האותות המקוריים.

על מנת לשפר את הביצועים ניתן לחשוב על עוד מאפיינים, כמו:

1. Mel-frequency cepstral coefficients (MFCC): These are widely used in speech and audio analysis. They represent the spectral envelope of a signal and can capture important characteristics such as pitch and timbre.
   - לא השתמשנו במאפיין זה כי הוא מצריך התקנה של עוד ספריות כמו librosa שלא בטוח תהייה מותקנת אצל הבודק, אך זהו מאפיין שהיה עוזר לסיווג האותות.

2. Spectral features: These include spectral centroid, spectral flatness, spectral rolloff, and spectral bandwidth. They can capture information about the frequency content of a signal.
   - לא רלוונטי כי הfrequency היה זהה בכל האותות.

3. Zero Crossing Rate (ZCR): This measures the number of times the signal crosses the horizontal axis (zero) and can provide information about the periodicity of a signal.
   - בעייתי לאותות המשוחזרים שהממוצע בהם 0.

4. Mel-Spectrogram: Similar to MFCCs, this is a visual representation of the power spectrum of a signal.

5. Time-domain features: These include root-mean-square (RMS) energy, zero crossing rate, and other statistical measures. They can capture information about the overall amplitude and dynamics of a signal.

לטובת שיפור הביצועים צריך להוסיף עוד נתונים עם התיוגים הנכונים.

לטובת שיפור הביצועים צריך לחשוב על שיטה טובה לזיהוי של האותות המשוחזרים לאותות המקוריים.

מקורות בהם השתמשנו לתרגיל:

1. בספריה learn-scikit נמצא המימוש של אלגוריתם ICA ידוע- FastICA : http://scikit-learn.org/stable/modules/generated/sklearn.decomposition.FastICA.html http://scikit-learn.org/stable/auto_examples/decomposition/plot_ica_blind_source_separation.html

מציאת שיטה למציאת דמיון בין signals

2. https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.correlate.html

3. השתמשנו בchatGPT לחשיבה של מאפיינים נוספים לצורך סיווג האותות.