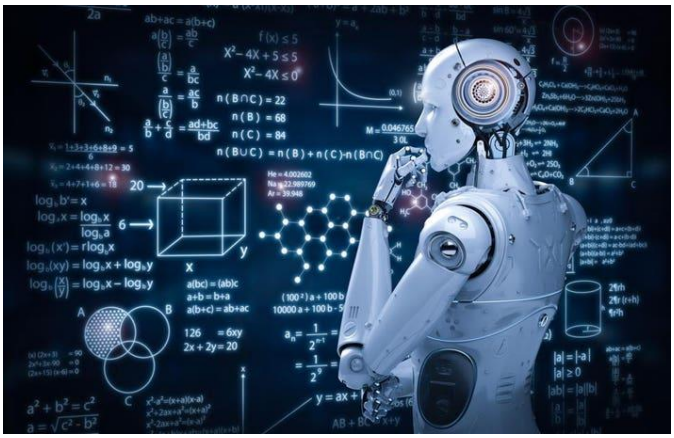
אוניברסיטת בן-גוריון בנגב
Ben-Gurion University of the Negev

המחלקה להנדסת תעשייה וניהול

קורס למידת מכונה 364-1-1811

פרויקט הקורס – חלק ב'

07/03/2024



קבוצה 16

205404965

315474205

תוכן עניינים

Model Training

בשלב זה בחרנו לבצע חלוקה של ה-dataset בשיטת hold-out. מכיוון שה-dataset אינו מאוזן, בעת החלוקה שלו ל-train set (80% מה-dataset) ול-validation set (20% מה-dataset), ביצענו את החלוקה באופן השומר על הפרופורציה בין סנטימנט חיובי לסנטימנט שלילי, בהתאם ל-dataset המקורי. בנוסף, בחרנו להשתמש בכל ה-features שנבחרו בסיום שלב ה-feature selection בחלק א', ללא ביצוע PCA לאחריו (35 features בסה"כ).

Decision Tree

סעיף 1

הפרמטרים שבחרנו לכוון בשלב זה הינם Max_depth, Criterion, Max_features ו-Class_weight.

- Max_depth – העומק המקסימלי של עץ ההחלטה. טווח הערכים שבחנו הוא בין 1 לבין העומק המקסימלי של העץ הדיפולטיבי.
- Criterion – מדד זה מסייע בבחינת איכות הפיצול בעץ ההחלטה. בחנו שימוש במדד Gini לבחינת איכות הפיצול ושימוש במדד Entropy.
- Max_features – המספר המירבי של features ששוקלים שיכללו בפיצול הטוב ביותר בעץ ההחלטה. עבור מדד זה בחנו את הערכים sqrt, log2, None.
- Class_weight – מעריך את משקל כל אחד מה-classes עבור dataset שאינו מאוזן, כלומר נותן משקולות שונות ל-labels בסט נתונים שאינו מאוזן. עבור פרמטר זה בחנו את הערכים balanced ו-None.

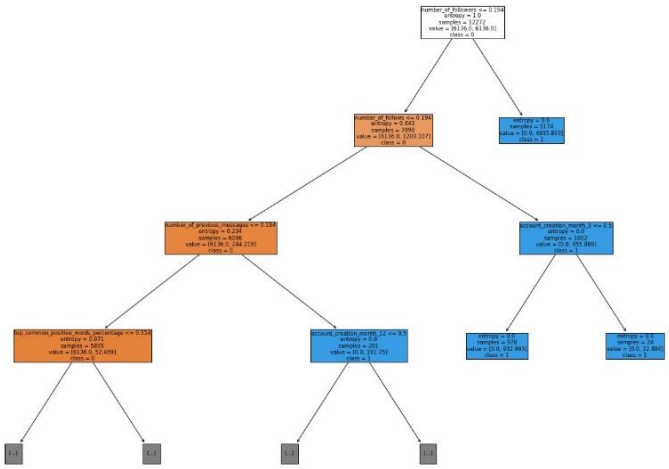
ציון ה-AUC ROC של ה-train set ושל ה-validation set הינו עבור המודל הטוב ביותר בשלב זה:

```
train set AUC ROC score: 0.9986403294
Test set AUC ROC score: 0.9980914253
```

תוצאות אלו עלולות להצביע על over fitting ופגיעה ב-Generalization של המודל, או מנגד על בחירה טובה מאוד של features וכיול טוב של ה-Hyperparameter (נספח 1 – עץ החלטה).

סעיף 2

יכולת ההסברה של מודל ה-decision tree (interpretability) מהווה יתרון בתהליך הלמידה של מודל ה- decision tree בהשוואה למודלים אחרים מכיוון שהאפשרות של המודל להסביר את ההחלטות שקיבל בצורה ברורה ופשוטה עשויה להקל עלינו להבין אילו תכונות ומאפיינים השפיעו על ההחלטות והבחירות של המודל, ואילו features הם בעלי חשיבות גבוהה יותר במודל (למשל ניתן להסיק כי ה-feature הנמצא בשורש עץ ההחלטה הינו feature בעל חשיבות גבוהה למשימת ה-classification של סוג ה-sentiment). אלו יאפשרו לנו לקבל תובנות חדשות ולזהות דפוסים מסוימים בכדי ללמוד מהנתונים בצורה יעילה יותר.



סעיף 3

עץ ההחלטה יכול לספק מסקנות אודות קשרים ודפוסים ב-dataset.

- מתוך עץ ההחלטה המיטבי ניתן לראות שה-feature בעל החשיבות הגבוהה ביותר למשימת הלימוד הינו number_of_followers, בשל מיקומו בעץ ההחלטה (שורש העץ).
- המבנה של השכבות העליונות של עץ ההחלטה מרמז שהמודל מצליח למצוא יחסית בצורה מהירה דפוסים ב-dataset. ניתן להסיק זאת מתוך כך שבשתי רמות של העץ מתוך 3 רמות המוצגות יש עלים, כלומר המודל מצליח לסווג בצורה יחסית מהירה samples מתוך ה-dataset. ברמה הראשונה והשלישית של עץ ההחלטה המתקבל, המודל הצליח לסווג samples כ-positive (class=1).
- אינדיקציה נוספת לכך שהמודל מצליח לסווג בצורה יחסית מהירה את ה-samples היא מדד ה-entropy שיורד בצורה משמעותית בין הרמות של העץ. ניתן לראות כבר ברמה השלישית של עץ ההחלטה כי מדד ה-entropy שווה או שואף לאפס (כתלות באם מדובר בעלה או בצומת בעץ בהתאמה), כלומר יש יותר סדר בנתונים ופחות חוסר וודאות.

Feature_name	Importance
number_of_followers	0.6349
number_of_follows	0.2705
number_of_previous_messages	0.0889
top_common_positive_words_percentage	0.0023
blue_tick	0.0012
message_date_year_2022	0.0012
message_date_16-23	0.0009
email_verified	0.0
account_creation_month_1	0.0
email_suffix_edu	0.0
email_suffix_ke	0.0
account_creation_year_2015	0.0
account_creation_month_5	0.0
account_creation_month_2	0.0
account_creation_month_3	0.0
message_date_month_10	0.0
account_creation_month_10	0.0
account_creation_month_11	0.0
account_creation_month_12	0.0
message_date_month_11	0.0
message_date_month_3	0.0
message_date_month_9	0.0
message_date_month_6	0.0
message_date_month_5	0.0
message_date_month_2	0.0
message_date_month_1	0.0
mp4	0.0
none_embedded_content	0.0
x	0.0
instagram	0.0
F	0.0
sum_top_common_positive_words	0.0
top_common_negative_words_percentage	0.0
sum_top_common_negative_words	0.0
account_creation_0-7	0.0

טבלת ה-feature_importances מציגה את החשיבות היחסית של כל feature בתהליך קבלת ההחלטות של המודל.

- ניתן לראות כי קיימת הלימה בין טבלת feature_importances לבין המודל של עץ ההחלטה המיטבי. הלימה זו באה לידי ביטוי בהימצאות של features בעלי חשיבות גבוהה בתחילת העץ, למשל ה-feature בשם number_of_followers הינו בעלת החשיבות הגבוהה ביותר ונמצא בשורש העץ.
- ניתן להסיק כי שבעת ה-features המופיעים בראש הטבלה מסייעים בזיהוי קשרים חשובים בין ה-samples וזיהוי דפוסים בתוכם. אלו מסייעים בהבנה של מבנה ה-data מבחינת אילו גורמים משפיעים על סיווג משתנה ההחלטה. כלומר, אלו הם ה-features בעלי המשמעות הכי גדולה ל-classification של משתנה המטרה.

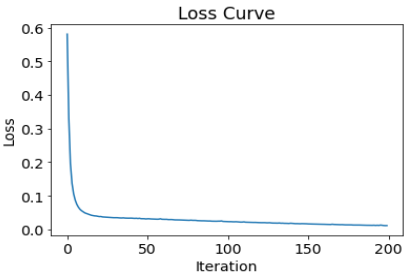
Artificial Neural Networks

(נספח 2 – רשת נוירונים)

סעיף 1

הקונפיגורציה עבור המודל הדיפולטיבי שנבחרה:

- מספר נוירונים בשכבת הכניסה – כמספר ה-Features ב-Dataset (35 features).
- מספר שכבות חבויות – מוגדר להיות שכבה אחת חבויה.
- מספר נוירונים חבויים בכל שכבה – מוגדר להיות 100 בשכבה חבויה אחת.



עבור מדד AUC ROC התקבלו התוצאות הבאות עבור המודל הדיפולטיבי (בחלוקה ל-train set ול-validation set):

```
Train set ROC AUC score: 0.9999086841
Test set ROC AUC score: 0.9975751988
```

. מהתוצאות ניתן להסיק שעבור מודל זה יכול להיות שנוצר מצב של overfitting עקב התוצאות הגבוהות, הן על ה-train set והן על ה-validation set ובכך ה-generalization עלול להיפגע, משמע חיזויים שגויים רבים יותר של ה-target feature בעת הזנת samples

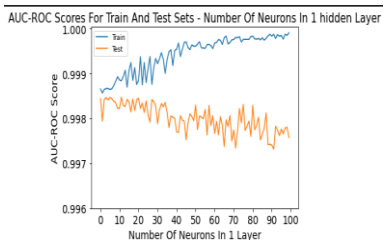
חדשים. בנוסף, לאור התוצאות, אפשר להסיק מסקנה מנוגדת האומרת שה-feature extraction שביצענו הינו טוב מאוד ולכן ציוני ה-AUC ROC גבוהים וגם ההפרש ביניהם קטן מאוד וה-generalization טוב מאוד. בנוסף, ניתן לראות בגרף Loss Curve שיש ירידה דרסטית עבור הערך Loss, כלומר כמות ה-error ירדה במהירות עבור כמות epoch (iteration) קטנה יחסית, דבר היכול לחזק את ההסקה השנייה שלנו, שכן יש אחוז נמוך של errors אחרי מספר מועט של הרצת epoch וה-generalization נשמר ברמה גבוהה.

<u>סעיף 2</u>

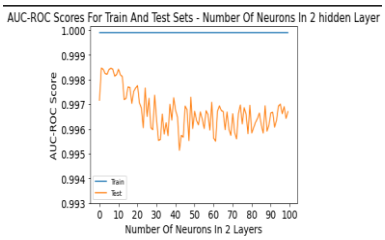
הצגת גרפים עבור כל hyperparameter בהשוואה בין train set ו-validation set:

- מספר השכבות החבויות ומספר הנוירונים בשכבה – מדובר פה בשני hyperparameters שונים אשר יצרנו גרפים המאחדים את שניהם במקביל. המשמעות לכיול מספר השכבות החבויות הינו שעבור מספר מועט מדיי, כמו שכבה אחת, עלול להוביל למודל פשוט ולא מורכב מספיק, ותוצאות ה-AUC ROC יהיו נמוכות יותר. מצד שני, בחירה במספר גדול של שכבות חבויות עלולה להוביל למצב של overfitting. כיוון שיש כמות רבה של samples אנו נרצה לבדוק טווח ערכים עבור סט פשוט יותר של 1-2 שכבות, וכן עבור 3 שכבות המהוות מודל מורכב יותר. כמו כן, יש משמעות לכייל את מספר הנוירונים בשכבה מפני שהנוירונים מכילים נתונים שבאמצעותם ניתן לבצע המרה של ערכים באמצעות פונקציית האקטיבציה, הם משתתפים בחישוב של ה-error בין השכבות החבויות ובכך עוזרות לכייל את המשקולות - W. אנו נרצה לבדוק טווח ערכים של 1-100 נוירונים בכל שכבה כאשר ערכים נמוכים עלולים להוביל למודל פשוט וערכים גבוהים עלולים להוביל למצב של overfitting. ערכים גבוהים מעידים על מורכבות המודל וכן זמן הריצה עלול להתארך בהתאם, אל מול ערכים נמוכים אשר עלולים לייצר מודל פשוט עם ציון AUC ROC נמוך יותר. נציג כעת את הגרפים שהתקבלו עבור שכבה אחת, שתי שכבות ושלוש שכבות:

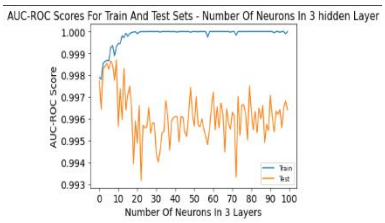
ניתן לראות שעבור 3 שכבות חבויות ו-10 נוירונים בכל שכבה חבויה מתקבל ה-Test AUC הטוב ביותר. כמו כן, ניתן לראות באופן יחסי עבור כל הגרפים שככל שה-Train AUC עולה, ה-Test AUC יורד. זהו ממצא הגיוני, שכן ככל שעולים במספר הנוירונים בשכבה המודל מאומן טוב מדי עד כדי שיש overfitting מפני שה- Train AUC מתקרב לערך 1 ובהתאמה הערך של ה-Test AUC יורד ובכך ה-generalization גם יורד.



train AUC score	best test AUC score	best size of neurons
0.9999086841194403	0.9984726080616492	12

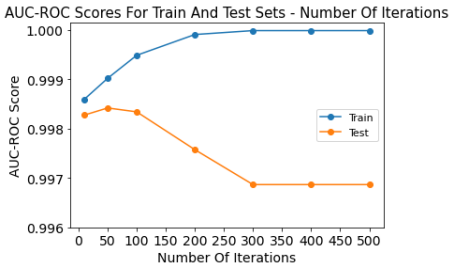


train AUC score	best test AUC score	best size of neurons
0.9999086841194403	0.9984606337346063	2



train AUC score	best test AUC score	best size of neurons
1.0	0.9987054421985929	10

- מספר איטרציות מקסימלי (max_iter) – זהו למעשה מספר ה-epoch, משמע מספר הפעמים שנריץ את האלגוריתם של ANN על מנת להפחית את ה-errors בין השכבות החבויות ברשת הנוירונים, דבר המוביל לכיול המשקולות - W בצורה טובה יותר עבור יותר הרצות. אנו נרצה שהאלגוריתם יעצור עבור ערך max שמביא את המשקולות לערכים המכוילים שלהן, וכן שה-error בין השכבות כמעט ולא משתנה עבור כמות רבה יותר של

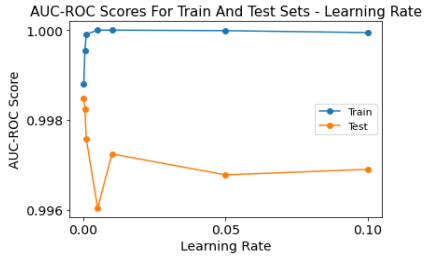


train AUC score	best test AUC score	best Number Of Iterations
0.9999880186908423	0.9984200540707391	50

epochs. נרצה לבדוק טווח ערכים הנע בין ערכים נמוכים יותר לערכים גבוהים יותר (באופן יחסי): 10, 50, 100, 200, 300, 400, 500. זאת על מנת לראות איך מספר האיטרציות במצבים שונים משפיע על טיב המודל - AUC Scores. באופן עקרוני הגדלת מספר האיטרציות אמור להוביל למצב של overfitting ועבור מספר נמוך של איטרציות באופן יחסי אנו אמורים לקבל ציון AUC ROC נמוך יותר. נציג כעת את הגרף המתקבל:

ניתן לראות שעבור 50 איטרציות מתקבל ה-Test AUC הטוב ביותר. בנוסף, ככל שעולים במספר האיטרציות ה-AUC Score של ה-train set עולה לערך 1, וניתן להסיק מכך שהמודל כנראה נמצא במצב של overfitting, שכן ה-Test AUC יורד בהתאמה עם עליית האיטרציות וה-generalization יורד בהתאמה.

- Learning Rate – קובע את הקצב שבו המשקולות מתעדכנות במהלך האימון והוא מכונה גודל הצעד. ערכים נמוכים עלולים להוביל לתהליך למידה ארוך מאוד וערכים גבוהים עלולים להוביל לקצב למידה מהיר מדי אשר עשוי להוביל לפספוס ערכי ביניים אופטימליים, ובכך לקבלת דיוק נמוך יותר. קצב למידה מהיר מדי עלול להוביל גם לשינויים קיצוניים בערך Loss. לפיכך, בחרנו טווח ערכים מגוון, מערכים נמוכים לערכים גבוהים והם: 0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1. נציג כעת את הגרף המתקבל:



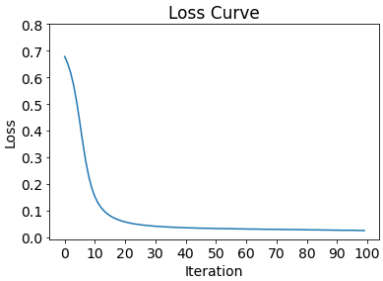
train AUC score	best test AUC score	best Learning Rate
0.9999999167964642	0.9984892390714308	0.0001

ניתן לראות שעבור learning rate ששווה 0.001 מתקבל ה-Test AUC הטוב ביותר. ניתן להסיק לאור הממצאים כי ערך נמוך של learning rate מוביל לתוצאה טובה מפני שהתהליך איטי וכיול המשקולות מדויק יותר, וכן לאור התוצאה של ה-Train AUC נראה שאנו לא במצב של overfitting ושומרים על generalization יציב וטוב.

```
Train set ROC AUC score: 0.9986403294
Test set ROC AUC score: 0.9980914253
```

תוצאות מדד ה-AUC ROC עבור ה-train set וה-validation set הינן:

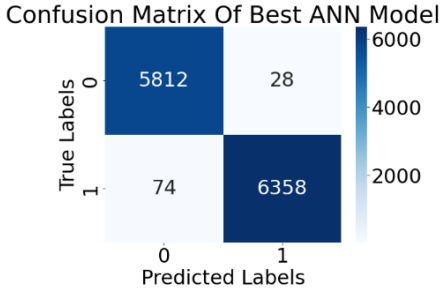
כמו כן, הוצאנו גרף של Loss Curve לאחר ה-hyperparameter tuning כמוצג להלן:



ניתן להסיק מתוצאות ה-AUC ROC והגרף, להבדיל מתוצאות סעיף 1, כי ציון ה- Test AUC טוב יותר וציון ה-Train AUC נמוך יותר, דבר המעיד על שיפור ביכולת ה-generalization עבור הקומבינציה המכוילת החדשה של ה-hyperparameter . זאת ועוד, ניתן לראות שעבור 3 שכבות חבויות, וכן עבור מספר שונה של מספר הנוירונים בשכבה חבויה לאחר הכיול, מתקבל Test AUC טוב יותר לעומת המודל הדיפולטיבי בעל שכבה אחת חבויה ו-100 נוירונים בשכבה הזו. מגרף ה-Loss Curve נראה שהגרף בתחילתו יורד בשיפוע פחות חד לעומת המודל הדיפולטיבי וזה יכול לנבוע שהמודל לאחר הכיול הינו מורכב יותר עם מספר שכבות חבויות רב ו-learning rate נמוך המאט את קצב עדכון המשקולות ולכן תיקון ה-error יכול לקחת יותר איטרציות וזמן ריצה באופן יחסי.

סעיף 3

יצרנו Confusion Matrix עבור המודל הסופי שלנו (על כל ה-dataset):



ביאור המטריצה:

TN = True Negative = 5812	FP = False Positive = 28
FN = False Negative = 74	TP = True Positive = 6358

הערכנו את המודל שלנו באמצעות 3 מדדי KPI כדי לאמוד את טיב המודל האופטימלי. נסביר כל מדד ונחשב עבור KPI ולבסוף נכתוב מסקנה כוללת. הבהרה- N = TN + FP + FN + TP.

- מדד Accuracy ("נכונות") – מודד את היחס של הפרדיקציות הנכונות של המודל, כלומר מידת ההצלחה הכוללת של המודל. תוצאת המדד: $ACC = \frac{TP+TN}{N} * 100 = 99.168\%$.
- מדד Sensitivity ("רגישות") – מודד את היחס בין התצפיות שסווגו כ-true positive לבין כלל התצפיות שסווגו כ-positive ב-dataset המקורי (כלומר true positive ו-false negative). תוצאת המדד: $TPR = \frac{TP}{TP+FN} * 100 = 98.849\%$.
- מדד Precision ("דיוק") – מודד את היחס בין התצפיות שסווגו כ-true positive לבין כלל התצפיות שסווגו כ-positive (כלומר true positive ו-false positive). תוצאת המדד: $PPV = \frac{TP}{TP+FP} * 100 = 99.561\%$.

המסקנה העולה מן הממצאים הינה שהמודל חוזה נכון ומסווג נכון באחוזים גבוהים מאוד את ה-labels של ה-samples. כמו כן, באחוזים גבוהים המודל מזהה נכון את הרוב המכריע של ה-samples המסווגים כחיוביים ככאלה (כלומר מזהה sample בעל label חיובי כחיובי) והוא בעל שיעור נמוך של תוצאות false positive. מהאמור לעיל ניתן להסיק שהמודל הסופי מבצע את משימת הסיווג ברמה גבוהה.

SVM

(נספח 3 – SVM)

```
train set AUC ROC score: 0.9984839484
Test set AUC ROC score: 0.9983368990
```

- תוצאות מדד AUC ROC עבור ה-Train set וה-Test set על פי ה-C המיטבי:

C Hyperparameter	Mean test score	Mean train score
0.1	0.9984368691	0.998482509
1	0.9982701082	0.9983714198
3	0.9982638444	0.9983585537
4	0.9982480403	0.9983649323
2	0.9982476089	0.9983484617
5	0.9982463897	0.9983662163
6	0.9982413974	0.9983722096
7	0.9982372431	0.9983693949
8	0.9982351668	0.9983694412
9	0.9982285033	0.99836864
10	0.9982251843	0.9983706997
50	0.9982018827	0.998359298
100	0.9981981508	0.9983626261
1000	0.9981869124	0.9983645626

- תוצאות הכיוונון מוצגות לעיל (כאשר הכיוונון הטוב ביותר נמצא בחלקו העליון של הטבלה):
- משוואת הישר המפריד עבור המודל הטוב ביותר:

```
Equation of the dividing line: [ 0.32887847  0.29700322
3.93389629  3.71270723  3.77376528 -0.91307248
 -0.80873464  1.18879177  1.17692056  0.29852853 -0.30224051
-0.1564518
 -0.15313946  0.09891201  0.28476444 -0.21530569 -0.04465312
-0.34600953
 -0.03769043 -0.2         0.062748    0.13426145  0.31447665
0.21876363
 -0.13253241  0.26030041 -0.24511823 -0.35996565 -0.40656598
0.08610574
 -0.32022322 -0.30176592 -0.15989632 -0.21297193
-0.06575208]
Maximum coefficient: 3.933896290525972
Minimum coefficient: -0.913072482606154
Feature with maximum coefficient:
number_of_previous_messages
Feature with minimum coefficient:
sum_top_common_negative_words
```

ניתן לראות את המקדמים (coefficients) של כל ה-features במודל שלנו. גודל המקדם וסימנו מספק מידע על איך ה-feature יעזור לסווג את משתנה המטרה שהינו ה-sentiment באשר אם הוא positive או negative. במקרה שלנו גודל המקדם הגבוה ביותר הינו של number_of_previous_messages וגודל המקדם הנמוך ביותר במודל הינו של sum_top_common_negative_words. Features אלו יסייעו לנו לסווג טוב יותר את ה-label של ה-samples.

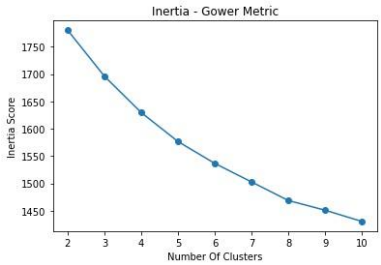
התוצאות שהתקבלו אינן מפתיעות מפני שלמשל עבור חשיבות ה-features בעץ החלטה קיבלנו כי number_of_previous_messages הינו במקום השלישי בחשיבותו בבניית העץ האופטימלי. לפיכך, שיש הלימה באופן יחסי של חשיבות ה-features עבור אלגוריתמים שונים.

Unsupervised Learning – Clustering

נריץ את האלגוריתם על סט נתונים שלא מכיל את משתנה המטרה (sentiment), כלומר נסיר את הסיווג של כל sample ל-classes הנתונים positive / negative, מכיוון שמדובר באלגוריתם ממשימת למידה מסוג Unsupervised Learning, בה המודל לא מקבל את הסיווג המקורי ל-classes. המודל מחלק את ה-samples הנתונים לו לקבוצות על סמך דימיון בין samples, מבלי לתת את המשמעות של כל אחת מהקבוצות. זאת בניגוד למודלים הקודמים בהם אימנו את המודל בעזרת סט נתונים שהכיל את משתנה המטרה, מכיוון שאלו היו מודלים מסוג משימת למידה Supervised Learning, שעושה שימוש במשתנה המטרה לטובת תהליך הלמידה של האלגוריתם בסיווג samples חדשים.

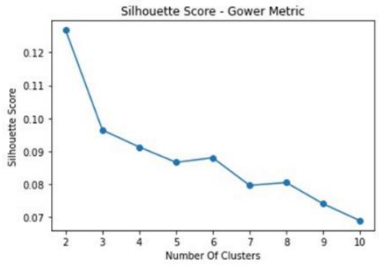
<u>סעיף 1</u>

בחרנו להריץ את האלגוריתם על סט ה-features אשר הגענו אליהם בסוף חלק א' ולהשתמש במטריקת "Gower" למדידת המרחק, עבור K בין 2 ל-10 (כולל). הסיבה לבחירה בשיטת "Gower" היא מכיוון שסט ה-features שהשתמשנו בו מכיל features נומריים ובינאריים, ושיטת מדידה זו מתאימה לסט נתונים המכיל mixed type features המכילים בין היתר features מהסוגים הנ"ל. נצפה לראות חלוקה ל-2 clusters, מכיוון שבסט הנתונים המקורי ה-samples מסווגים לשני classes.

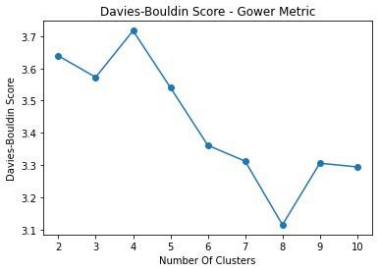


<u>סעיף 2</u>

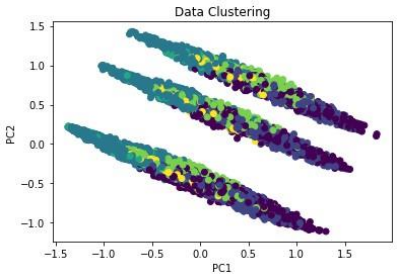
השתמשנו בשלושה קריטריונים להשוואה: Inertia, Silhouette, Davies-Bouldin.



- Inertia – קריטריון זה מודד כמה טוב בוצעה החלוקה ל-clusters על ידי המודל באמצעות מדידת סכום המרחקים הריבועיים בין ה-samples למרכז ה-clusters אליו ה-sample משויך. מספר ה-clusters האופטימלי מופיע באזור בגרף בו נוצר "מרפק", כלומר הנקודה שבה הירידה בערך ה-inertia מתחיל להאט. מכאן, ערך ה-k האופטימלי שבחרנו על פי הגרף הינו k=6.
- Silhouette – קריטריון זה מודד עד כמה חלוקת ה-samples לקבוצות בוצעה בצורה נכונה על ידי מדידת ההפרדה בין ה-clusters, מדידת הגודל של כל cluster ובדיקת ההתאמה של כל sample ל-cluster אליו הוא שויך ביחס להתאמה ל-clusters אחרים. מספר ה-clusters האופטימלי יופיע אחרי הירידה המשמעותית ביותר בגרף. מכאן, ערך ה-k האופטימלי שבחרנו על פי הגרף הינו k=3.



- Davies-Bouldin – קריטריון זה מודד את טיב החלוקה ל-clusters באמצעות מדידת הדימיון בין ה-clusters, המבוסס על מדידת הפיזור וההפרדה בין ה-clusters. מספר ה-clusters האופטימלי יופיע עבור הערך הקטן ביותר של המדד. מכאן, ערך ה-k האופטימלי שבחרנו על פי הגרף הינו k=8.



בעזרת מדדים אלו בחרנו את מספר ה-clusters להיות 6 (ממוצע תוצאות המדדים מעלה עבור k). סט הנתונים המקורי מסווג באופן בינארי, כלומר קיימים שני class, לכן קשה לייחס משמעות לכל cluster כאשר k=6 וניתן להסיק כי המודל אינו מתאים לאופי הנתונים.

השוואה בין המודלים – Evaluation

<u>סעיף 1</u>

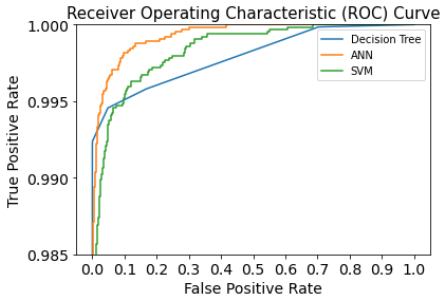
המטריקה בה בחרנו להשתמש להשוואה בין המודלים הוא AUC ROC אשר מודד את היחס בין החיזויים המוצלחים לבין החיזויים הכוזבים - $\frac{TPR-True\ Positive\ Rate}{FPR-False\ Positive\ Rate}$

תוצאות האלגוריתמים הן כדלקמן:

best svm model AUC score
0.9983345526221631

best tree model AUC score	best ann model AUC score
0.9979208330138689	0.999295394815307

ניתן לראות שהמודל עם התוצאה המיטיבית הינו מודל ה-ANN.



ניתן לראות זאת גם בגרף ROC Curve. השטח מתחת לכל גרף מעיד על טיב המודל, משמע ככל שהשטח גדול יותר כך ציון ה-AUC ROC עולה. מהגרף ניתן לראות שהשטח הגדול ביותר שייך למודל ANN והדבר מחזק את הבחירה בו כמודל המיטבי.

<u>סעיף 2</u>

אנו נמשיך מעתה עם אלגוריתם ANN שקיבל את ציון ה-AUC ROC המירבי עם ה-hyperparameters הבאים:

The Best ANN Model

best ann model AUC score	Hidden Layer Size	Learning Rate Init	Max iterations
0.999295394815307	(94, 13, 83)	0.0001	100

שיפור המודל הנבחר – Improvements

<u>סעיף 1</u>

שיפור נתונים – בחלק א׳ של העבודה בחרנו לבצע הורדת מימד לנתונים לאחר שלב ה-feature selection. בחלק זה בחרנו בעת החלוקה של סט הנתונים ל-hold out לא לבצע הורדת מימד כפי שעשינו בסוף חלק א׳. למרות שקיבלנו תוצאת AUC ROC יחסית גבוהות עבור המודלים שבחרנו לאורך חלק זה של הפרויקט, אנו חוששים שאי ביצוע של PCA עלול להוביל למספר בעיות:

- מימדיות גבוהה הנובעת מכמות גדולה יחסית של features שעשויה להוביל למורכבות חישובית גבוהה, זמני אימון ארוכים של המודל ו-overfitting.
- מולטיקולינאריות, כלומר קורלציה גבוהה בין ה-features, שיכולה להוביל ל-interpretability נמוך (למשל במודל של עץ החלטה) ול-overfitting.
- רעש ב-dataset שעלול להחליש את המודל עקב פגיעה ביכולת ה-generalization של המודל.

באמצעות ביצוע PCA על הנתונים ובחירת קומפוננטות המכילות כ-80% מהשונות המוסברת של המודל, אנו מצפים לראות שיפור בזמני הריצה של המודל והפחתה ב-overfitting של המודל (ימדד באמצעות גרף Loss Curve).

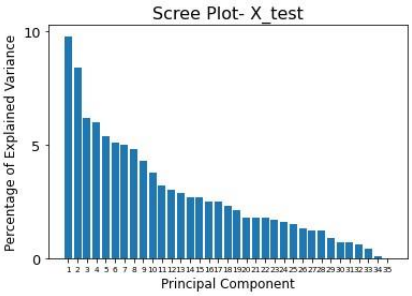
שיפור מודל ANN – המודל האופטימלי שקיבל את ציון ה-AUC ROC הגבוה ביותר הינו ANN. למרות שהוא קיבל את הציון הגבוה ביותר, אנו חוששים מכמה בעיות שבגינן נרצה לנסות לשפר את איכות המודל.

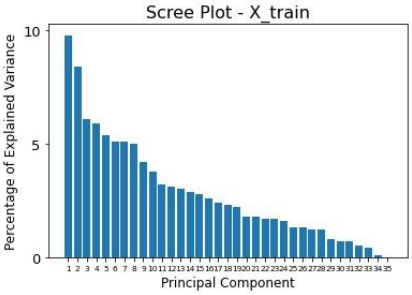
- תיתכן בעייה ביכולת ה-generalization של המודל, על אף שקיבל ציון AUC ROC גבוה.
- תיתכן רגישות לחלק מה-features במודל או לנתונים חריגים, שעלולה להוביל את המודל להיות מושפע מרעש או מתכונות לא רלוונטיות ובכך להחליש את המודל ואת היכולת שלו לסווג נכונה.

על ידי שיפור המודל באמצעות אלגוריתם bagging אנו מצפים לראות יכולת generalization גבוהה יותר של המודל ויכולת classification גבוהה יותר (ימדדו באמצעות ציון AUC ROC).

סעיף 2

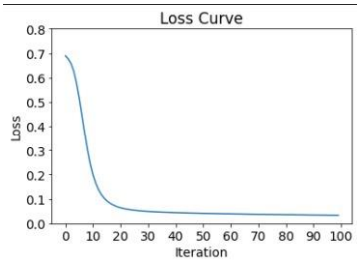
שיפור נתונים – בוצע באמצעות הורדת הממד של הנתונים עם PCA. התחלנו ביצירת גרף המציג את אחוז השונות המוסברת של כל PC, בחלוקה ל-train set ול-test set. לפי גרף זה בחרנו להוריד את המימד ל-20, מכיוון ש-20 PC מכילים 80% מהשונות המוסברת. את הורדת המימד ביצענו בנפרד על ה-train set ובנפרד על ה- validation set ואנו מניחים כי התוצאות של הורדת המימד יהיו דומות מכיוון ששני ה-datasets מגיעים מאותה ההתפלגות וכל ה-features הם מבוססי מומחה.





אימנו את מודל ה-ANN האופטימלי שקיבלנו מוקדם יותר בעזרת ה-train set לאחר הורדת המימד, ובחנו את המודל לאחר האימון על ה-validation set לאחר הורדת המימד. תוצאות מדד ה-AUC ROC שהתקבלו הינן:

```
Train set ROC AUC score: 0.9990599249
Test set ROC AUC score: 0.8141624357
```



. בנוסף, יצרנו גרף Loss Curve על מנת לבחון את מידת ה-overfitting של המודל לאחר הורדת המימד. מהתוצאות של מדד ה-AUC ROC ומהגרף, אנו רואים כי לא ניכר שיפור. הסיבה שיכולה להוביל לכך שביצוע PCA על הנתונים לא שיפר את המודל הינה מספר ה-PC שנבחרו. ייתכן ואם היינו בוחרים מספר שונה של מימדים, התוצאה הייתה משתפרת, למשל אם היינו מסתמכים על כלל האצבע M^d קטן ממספר ה-samples ב-datasets (M= מספר ה-classes , d= מספר המימדים/features). אנו חושבים שאי עמידה בכלל האצבע יצר overfitting, שעלול להוביל לפגיעה ב-generalization, וניתן לראות זאת בתוצאת ה-AUC ROC.

שיפור המודל – השתמשנו באלגוריתם bagging, אשר דוגם מספר samples מתוך ה-train set ומאמן את המודל לפיהם. לאחר מכן, האלגוריתם צובר את הפרדיקציות שבוצעו מכלל תתי ה-dataset כדי לבצע prediction סופי. בחנו טווח של 1-10 n_estimators, ובאמצעות grid search כיוונו את ה-n_estimators. קיבלנו n_estimators=8 ואת תוצאות ה-AUC ROC להלן:

```
Train set ROC AUC score: 0.9995673000
Test set ROC AUC score: 0.9984446680
```

. ניתן לראות כי לא חל שיפור במודל. אלגוריתם זה מסייע בהפחתת overfitting, המשפרת את יכולת ה-generalization של המודל. אנו מעריכים כי הסיבה לכך שלא חל שיפור היא שבמודל האופטימלי שלנו אין overfitting ולכן השימוש באלגוריתם זה לא שיפר את התוצאה.

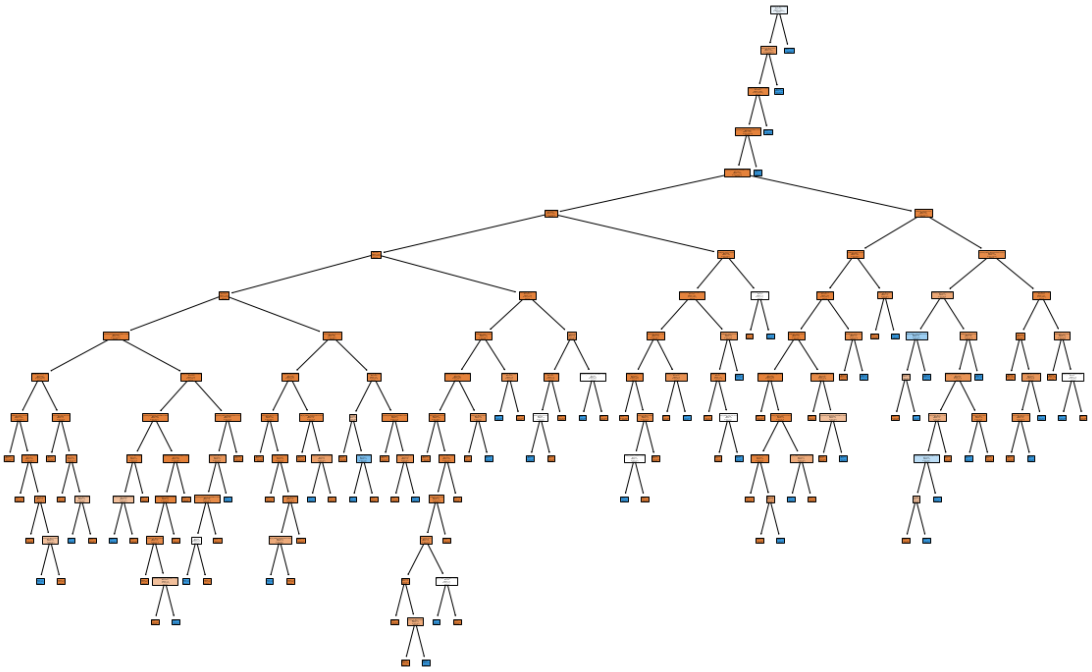
לסיכום, נבצע את החיזוי בעזרת המודל ANN האופטימלי.

נספח 1 – עץ החלטה

חזרה לסעיף Decision trees

התחלנו בבניית מודל דיפולטיבי בעזרת ה-train set, וקיבלנו ציון AUC ROC של 1, המצביע על מצב של over fitting. בהרצת ה-validation set קיבלנו ציון AUC ROC של 0.9912873466, שיכול להצביע על מצב של over fitting או שבחירת ה-features הייתה ממש טובה. לאור המשמעות שייחסנו לציון ה-AUC ROC של המודל הדיפולטיבי, השתמשנו באלגוריתם gridsearch לכיול ההיפרפרמטרים.

- עץ ההחלטה הדיפולטיבי שהתקבל



- ציוני AUC ROC עץ החלטה דיפולטיבי

```
train set AUC ROC score: 1.0000000000
test set AUC ROC score: 0.9912873466
```

- היפרפרמטרים שנבחרו בתום תהליך gridsearch בעץ ההחלטה

The Best Hyperparameters For The Decision Tree Model

Mean test score	Mean train score	Max depth	Criterion	Class weight	Max Features
0.9973119281	0.9986701249	6	entropy	balanced	

- ציון AUC ROC וערכי ה-hyperparameters הנבחרים לאחר כיול למודל המיטבי

The Best Decision Tree Model

best tree model AUC score	Max depth	Criterion	Class weight	Max Features
0.9979208330138689	6	entropy	balanced	

חזרה לסעיף Decision trees

נספח 2 – רשת נוירונים

חזרה לסעיף Artificial neural networks

עבור אלגוריתם ANN השתמשנו במודל MLP על מנת לבנות רשת נוירונים. התחלנו מבניית מודל דיפולטיבי, כאשר ה-hyperparameters אליהם נתייחס הינם מספר נוירונים בשכבת הכניסה, מספר שכבות חבויות ומספר נוירונים חבויים בכל שכבה.

בחרנו לבצע את ה-hyperparameter tuning באמצעות Random Search מפני שרצינו לבחון קומבינציות רבות של ה-hyperparameter שהצגנו, ואלגוריתם gridsearch אינו מתאים בשל זמן הריצה העולה בצורה אקספוננציאלית כתלות במספר ה-hyperparameters. לאור בחינת הקומבינציות הרבות כאמור, אנו צופים שיצאו ערכי פרמטרים שונים ממה שהצגנו בגרפים מפני שהתייחסנו לכל hyperparameter בנפרד. לאחר ההרצה קיבלנו שהערכים האופטימליים הינם:

Mean train score	Mean test score	Hidden Layer Size	Max iterations	Learning Rate Init
0.999347312	0.998468071	(94, 13, 83)	100	0.0001

על מנת לקבל את ציון ה-Test AUC הסופי של המודל, קבענו את ערכי ה-hyperparameter על פי ה-hyperparameter tuning המיטבי שקיבלנו ואימנו את המודל על כל ה-dataset שלנו. קיבלנו את התוצאה:

The Best ANN Model

best ann model AUC score	Hidden Layer Size	Learning Rate Init	Max iterations
0.999295394815307	(94, 13, 83)	0.0001	100

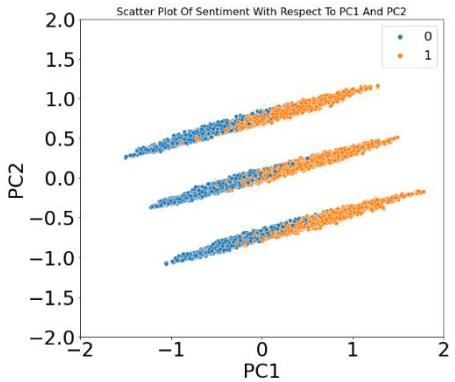
- מיקום הגרפים Loss Curve בקוד – הגרף הראשון שיצרנו נמצא אחרי אימון המודל הדיפולטיבי, והגרף השני נמצא אחרי הכיול של ה-hyperparameters. מיקום הגרפים בנקודות אלו בקוד מסייעות לקבל הבנה על טיב למידת המודל.

חזרה לסעיף Artificial neural networks

נספח 3 – SVM

חזרה לסעיף SVM

ראשית ביצענו הורדת מימד באמצעות PCA לכדי שני features על מנת שנוכל לראות את התפלגות משתנה המטרה בציר קרטזי. להלן הגרף שהתקבל:



ניתן לראות שעבור ערכי PC1 הגדולים מ-0 ה-sentiment מסווג יותר כ-positive, ולהיפך כ-negative. לאחר מכן עבדנו עם הנתונים שלא עברו את תהליך ה-PCA.

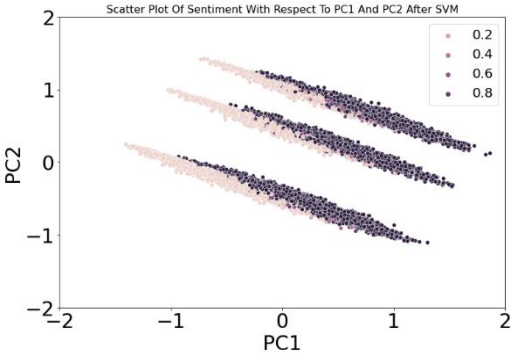
ביצענו hyperparameter tuning עבור הפרמטר C. פרמטר זה מגדיר למודל עד כמה הוא מוכן לטעות בסיווג של sample מסוים. עבור ערכים גדולים של הפרמטר, המודל יבחר מישור הפרדה עם שוליים קטנים יותר, אם מישור ההפרדה הזה מבחין בצורה טובה בין המחלקות. עבור ערכים קטנים של הפרמטר, מישור ההפרדה שייבחר יהיה עם שוליים רחבים יותר, גם אם מישור ההפרדה הזה יסווג בצורה שגויה יותר samples. השתמשנו ב- Grid Search. כפי שניתן לראות ה-C המיטבי הינו 0.1, אשר קיבל את ה-Test score הגבוה ביותר.

לאחר מכן, על מנת לקבל את Test AUC הסופי של המודל הרצנו על פי ה-hyperparameter tuning המיטבי שקיבלנו את המודל על כל ה-dataset שלנו וקיבלנו שהתוצאה הינה:

The Best SVM Model

best svm model AUC score	C Hyperparameter
0.9983345526221631	0.1

לבסוף, ביצענו שוב הורדת מימד באמצעות PCA לכדי שני features על המודל הסופי שקיבלנו על מנת שנוכל לראות את התפלגות משתנה המטרה בציר קרטזי. להלן הגרף שהתקבל:



ניתן לראות שעבור ערכי PC1 הגדולים מ-0 ה-Sentiment מסווג יותר כ-Positive, ולהיפך כ-Negative, כלומר המודל מצליח לסווג בצורה יחסית טובה את ה-samples.

חזרה לסעיף SVM