בס"ד

פרויקט שנה ב' -

מבוא למדעי הנתונים AI

שם המרצה: דורון פרלמן

מגישות:

טל אלפר 322208703 | ירדן ממן 212149439

בס"ד

הסבר כללי

הדאטה עליו בחרנו לעשות את הפרויקט הוא סביב תחזית הסיכוי ללקות בשבץ מוחי תוך התחשבות במספר פרמטרים עליהם נפרט בהמשך.

קישור לדאטה: https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset

שאלת המחקר: מה הסיכוי ללקות בשבץ מוחי בהסתמך על מידע כמו סטטוס עישון, מגדר, מחלות רקע וכו'.

הרחבה על הדאטה:

הדאטה מורכבת מכ - 5110 דגימות.

מס' הדגימות בהם חלו בשבץ - 249 (4.87%).

מס' הדגימות בהם לא חלו בשבץ - 4861 (95.13%).

כבר מתוך הסתכלות ראשונה על הנתונים ניתן לראות בבירור כי הדאטה לא מאוזן.

```
# Count the number of sampeles that have stroke before cleaning (1)
(data['stroke'] == 1).sum()
```

```
249
```

```
# Count the number of sampeles that did not have a stroke before cleaning (0)
(data['stroke'] == 0).sum()
```

```
4861
```

פיצ'רים:

1. Id - מס' סידורי של המשתתף.
2. Gender - מגדר המשתתף, אופציות הבחירה הן male/female/other.
3. Age - גיל המשתתף.
4. Hypertension - לחץ דם.
5. Heart_disease - מחלות לב.
6. Ever_married - סטטוס זוגי, אופציות הבחירה הן yes/no.
7. Work_type - סוג העבודה, אופציות הבחירה הן-
   - children - ילדים (לא בתעסוקה, מתחת לגיל בו עובדים).
   - Govt_job - עובד במגזר הציבורי (עובדי ממשלה).
   - Never_worked - מעולם לא עבדו (כולל פנסיונרים).
   - Private - עובדים בשוק הפרטי (חברות, עסקים).
   - Self-employed - עצמאים (בעלי עסק, פרילנסרים).
8. Residence_type - סוג אזור מגורים, אופציות הבחירה הן urban/ rural.
9. Avg_glucose_level - רמת גלוקוז.
10. BMI.
11. Smoking_status - סטטוס עישון, אופציות הבחירה הן -
    - Formally smoked

בס"ד

- Never Smoked
- Smokes
- Unknown

12. Stroke - העמודה עליה עושים את התחזית, אופציות הבחירה הן 0/1.

- 1 לקה בשבץ מוחי.
- 0 לא לקה בשבץ מוחי.

לאחר טעינת הדאטה והצגתו שמנו לב לדברים הבאים:

- בעמודה של ה BMI יש כ 201 תשובות שהן NA.

```
data['bmi'].isna().sum()
```

```
201
```

- בעמודה GENDER ישנה שורה אחת עם התשובה "Other".

בס"ד

<u>מחברת ראשונה - בניית רשת נוירונים :</u>

1. **טעינת הדאטה**
2. **ויזואליזציה של הדאטה**
3. **הינדוס ובחירת פיצ'רים**

ביצענו ניקיון לדאטה:

- הסרנו את השורות בהן ה BMI היה NAN.
- הסרנו את המודה ID משום שאין לה תרומה ללמידת הדאטה.
- הסרנו את השורה בה gender הוא other?

```
# Count the number of sampeles that have stroke after cleaning (1)
(fix_data['stroke'] == 1).sum()
```

```
209
```

```
# Count the number of sampeles that did not have a stroke after cleaning (0)
(fix_data['stroke'] == 0).sum()
```

```
4700
```

```
# Count the number of sampeles that Nan after cleaning (0)
fix_data['bmi'].isna().sum()
```

```
0
```

הגדרנו את הפיצ'רים לפיהם המודל ילמד את התופעה (X).

הגדרנו את הפיצ'ר אותו רוצים לחזות (Y).

4. **חלוקת הדאטה לשתי קבוצות test,train**

Train - מהווה 80% מהנתונים.

Test - מהווה 20% מהנתונים.

עם stratify לשימור יחס המחלקות.

5. **הכנת דאטה (קידוד משתנים קטגוריים וסטנדרטיזציה)**

חילקנו את הדאטה לשלושה סוגים -

- משתנים קטגוריים - smoking_status, work_type, gender. על פיצ'רים אלו ביצענו one hot encoding.
- משתנים בינארים - על פיצ'רים אלו לא ביצעו קידוד. על חלק מהפיצ'רים ביצענו המרה אשר משמעותה מפורטת בטבלה. את ההמרה עשינו בעזרת פונקציית map, פונקציה אשר ממפה ערכים בינארים במקום הערכים הקיימים.

פיצ'ר	0	1
residence_type	Urban	Rural
ever_married	No	Yes
heart_disease	-	-
hypertension	-	-

- משתנים מספריים - bmi, avg_glucose_level, age. על פיצ'רים אלו ביצענו סטנדרטיזציה (הורדת הממוצע וחלוקה בסטיית התקן).

חיברנו את כל סוגי הפיצ'רים למערך אחד רציף בעזרת np.hstack: הפיצ'רים הקטגוריאליים שעברו קידוד One-Hot, הפיצ'רים הבינאריים שהומרו ל־0/1 והפיצ'רים המספריים שעברו נרמול. בצורה זו יצרנו סט נתונים מאוחד (X_train_ready ו־X_test_ready) שבו כל המאפיינים מוצגים בערכים מספריים בלבד ובפורמט אחיד, כך שהמודל יכול לקבל אותם כקלט ישיר לאימון ולבחינה.

6. **בניית והרצת המודל**

<u>בניית המודל - הרצה 1:</u>

קלט: מספר הפיצ'רים לאחר קידוד וסקיילינג.

שכבות חבויות: 2 שתי שכבות חבויות עם 16 נוירונים ופונקציית אקטיבציה relu.

שכבת פלט: נוירון אחד (כמספר הנתונים אותם רוצים לחזות) ופונקציית אקטיבציה sigmoid משום שמדובר בבעיית קלסיפיקציה.

אופטימייזר: adam

פונקציית הפסד: binary_crossentropy

מדד ביצועים: accuracy, confusion matrix

<u>אימון המודל - הרצה 1:</u>

האימון מתבצע על הנתונים בקבוצת ה - X_train_ready, y_train.

הגדרנו שהמודל ירוץ 30 פעמים (epochs) ושבכל עדכון משקולות משתמשים ב 32 דגימות (batch_size).

הגדרנו ש 20% מהנתונים יוקצו לבדוק את ביצועי המודל במהלך האימון.

<u>הערכת הביצועים של המודל - הרצה 1:</u>

הערכת רשת הנוירונים על קבוצת הנתונים ב test, הרשת לא נחשפה לנתונים הללו. אנחנו בודקות מה אחוזי ההצלחה של המודל ומה אחוזי הכישלון לפי פונקציית השגיאה.

ניתן לראות כי בכל epochs של המודל אחוזי ההצלחה היו גבוהים מאוד (מעל 95% הצלחה).

מצב זה נוצר בעקבות העובדה שהדאטה אינו מאוזן ולכן הגיוני שהמודל יצליח לחזות את הסיכוי לחזות את הסיכוי ללקות בשבץ באחוזים גבוהים.

7. **ויזואליזציה**

בהערכת הביצועים של המודל (לפי הערכת הצלחה ושגיאה) קיבלנו את הפלט הבא:

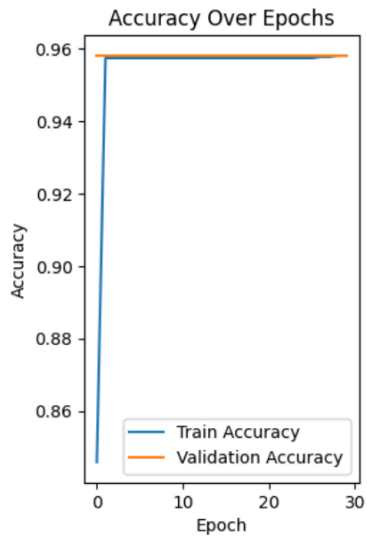
```
loss, accuracy = model.evaluate(X_test_ready, y_test, verbose=0)
print(f"Test Accuracy: {accuracy:.2f}")
```

```
Test Accuracy: 0.96
```

המודל הצליח לחזות את הסיכוי ללקות בשבץ מוחי ב 96% מהמקרים.

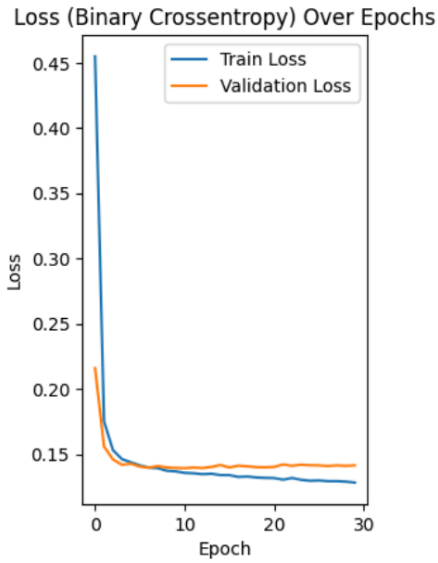
בס"ד

גרף הצלחה:



גם העקומה של **Train Accuracy** וגם של **Validation Accuracy** כמעט זהות לכל אורך האימון.
כלומר, המודל לומד מהר מאוד כבר מהepoch הראשון להגיע לדיוק גבוה ואין שיפור אמיתי בתחזית ולכן הקווים שטוחים. לא נראה שיש overfitting משום שאם אכן היה קיים היינו מצפים לראות את הtrain עולה ואת הvalidation קבוע או יורד.
מתוך כך ניתן להסיק שמדד ההצלחה (Accuracy) לא משקף ביצועים טובים בנתונים משום שהם לא מאוזנים.

גרף שגיאה:



גם כאן ניתן לראות שה־**loss** של ה־Train וה־Validation צונחים מהר מאוד בתחילת האימון.
כלומר, המודל מתכנס מהר לתחזית וממשיך ללמוד מעט מאוד לאחר הepochs הראשונים.

בס"ד

בהערכת הביצועים של המודל (לפי מטריצת הבלבול) קיבלנו את הפלט הבא:

הסבר על הפרמטרים:

precision = מדד דיוק חיובי, מתוך כל מה שהמודל *ניבא* כשייך למחלקה מסוימת, כמה מהם באמת נכונים.

מחלקה 1 (stroke):

אם precision גבוה > כשהמודל אומר "חולה", אפשר לסמוך עליו שזה באמת חולה.

אם נמוך > יש הרבה False Positives (אנשים בריאים שהמודל טעה וחשב שהם חולים).

מחלקה 0 (לא stroke):

אם precision גבוה > כשהמודל מנבא מישהו כבריא, זה באמת בריא.

Recall = מדד רגישות, מתוך כל הדוגמאות שבאמת שייכות למחלקה מסוימת, כמה מהן המודל הצליח לזהות נכון.

מחלקה 1 (stroke):

אם recall גבוה > רוב החולים שזוהו כחולים באמת נתפסו.

אם recall נמוך > המודל פספס חולים (False Negatives רבים).

מחלקה 0 (לא stroke):

אם recall גבוה אומר שהמודל מזהה היטב בריאים כלא חולים.

F1-score = משלב גם את היכולת *לזהות נכון* (recall) וגם *לא לטעות יותר מדי* (precision).

מחלקה 1 (stroke):

כאשר F1 נותן תמונה מאוזנת: כמה טוב המודל גם מזהה חולים וגם לא מתבלבל עם בריאים.

מחלקה 0 (לא stroke):

דומה, אבל בדרך כלל זו המחלקה הגדולה ולכן ה־F1 שלה נוטה להיות גבוה יחסית.

support = כמה דוגמאות אמיתיות היו מכל מחלקה בסט הבדיקה.

Accuracy = אחוז הניבויים שהמודל צדק בהם.

macro avg = ממוצע רגיל של שתי הקבוצות (0/1).

weighted avg = ממוצע משוקלל של שתי הקבוצות.

ROC AUC score = מתמקד ביכולת לזהות את המחלקה החיובית (במקרה שלנו stroke). כאשר ROC-AUC גבוה המודל מתעלם מהמחלקה הקטנה.

```
Classification Report:
              precision    recall  f1-score   support

           0      0.957     1.000     0.978       940
           1      0.000     0.000     0.000        42

    accuracy                          0.957       982
   macro avg      0.479     0.500     0.489       982
weighted avg      0.916     0.957     0.936       982

Confusion Matrix:
 [[940   0]
 [ 42   0]]
ROC AUC Score: 0.794630192502533
```

<u>מטריצת הבלבול:</u>

940 מקרים של "לא שבץ" (0) שנחזו נכון כ"לא שבץ".

0 מקרים של "לא שבץ" שחזו בטעות כ"שבץ".

42 מקרים של "שבץ" שחזו בטעות כ"לא שבץ".

0 מקרים של "שבץ" שחזו נכון כ"שבץ".

כלומר - המודל מעולם לא חזה מקרה של שבץ! תמיד ניבא "לא שבץ".

<u>דו"ח סיווג:</u>

1. דיוק כללי accuracy = 95.7%

מקרים בהם אנשים ללא שבץ (940 מתוך 982). המודל חזה תמיד שאין שבץ ולכן קיבלנו אחוזי דיוק גבוהים, אך בפועל המודל נכשל במשימה החשובה שהיא לזהות שבץ.

2. מדדי **Precision, Recall, F1 לפי מחלקות:**

מחלקה 0 (לא שבץ):

Precision = 0.957: מתוך כל מי שהמודל אמר עליהם "לא שבץ", כמה באמת לא חלו?

- המודל חזה "לא שבץ" לכל 982 האנשים.
- 940 מהם באמת לא חלו (True Negatives).
- 42 מהם כן חלו אבל הוחמצו (False Negatives).

Recall = 1.000: מתוך כל האנשים שבאמת לא חלו, כמה זיהה המודל?

- היו 940 "לא שבץ" אמיתיים. המודל מצא את כולם (לא פספס אף אחד).
- כלומר - המודל מושלם בזיהוי מי שלא חלה.

F1 = 0.978: הערך גבוה מאוד כלומר המודל מזהה טוב מקרים של "לא שבץ".

מחלקה 1 (שבץ):

Precision = 0.000: מתוך כל מי שהמודל חזה עליהם "שבץ", כמה באמת חלו?

- בפועל המודל לא חזה אף פעם מקרה של "שבץ".

Recall = 0.000: מתוך כל האנשים שבאמת חלו (42 מקרים), כמה המודל מצא?

- המודל לא חזה מקרה של "שבץ" על אף אחד.

F1 = 0.000: המודל לא מזהה אפילו מקרה אחד של "שבץ".

3. מדד ROC AUC Score = 0.79

המודל כן יודע במידה מסוימת להבדיל בין מי שבסיכון ומי שלא, אבל התוצאה בפועל היא שהמודל תמיד חוזה "לא שבץ".

מסקנה:

המודל במצבו הנוכחי מזהה מצוין מי לא חולה, אבל מתעלם לחלוטין מהחולים האמיתיים - ולכן רשת זו לא טובה לתופעה אותה אנו חוקרות.

בס"ד

<u>מחברת שנייה - בניית רשת נוירונים (שכבה חבויה אחת):</u>

1. **טעינת הדאטה לאחר הניקוי (שנעשה במחברת בראשונה)**
2. **הינדוס ובחירת פיצ'רים**
3. **חלוקת הדאטה לשתי קבוצות test,train**

Train - מהווה 80% מהנתונים.

Test - מהווה 20% מהנתונים.

עם stratify לשימור יחס המחלקות.

4. **הכנת דאטה (קידוד משתנים קטגוריים וסטנדרטיזציה)**

חילקנו את הדאטה לשלושה סוגים -

- משתנים קטגוריים - smoking_status, work_type, gender. על פיצ'רים אלו ביצענו one hot encoding.
- משתנים בינארים - על פיצ'רים אלו לא ביצעו קידוד. על חלק מהפיצ'רים ביצענו המרה אשר משמעותה מפורטת בטבלה. את ההמרה עשינו בעזרת פונקציית map, פונקציה אשר ממפה ערכים בינארים במקום הערכים הקיימים.

פיצ'ר	0	1
residence_type	Urban	Rural
ever_married	No	Yes
heart_disease	-	-
hypertension	-	-

- משתנים מספריים - bmi, avg_glucose_level, age. על פיצ'רים אלו ביצענו סטנדרטיזציה (הורדת הממוצע וחלוקה בסטיית התקן).

חיברנו את כל סוגי הפיצ'רים למערך אחד רציף בעזרת np.hstack: הפיצ'רים הקטגוריאליים שעברו קידוד One-Hot, הפיצ'רים הבינאריים שהומרו ל־0/1 והפיצ'רים המספריים שעברו נרמול. בצורה זו יצרנו סט נתונים מאוחד (X_train_ready ו־X_test_ready) שבו כל המאפיינים מוצגים בערכים מספריים בלבד ובפורמט אחיד, כך שהמודל יכול לקבל אותם כקלט ישיר לאימון ולבחינה.

5. **בניית והרצת המודל**

<u>בניית המודל - הרצה 2:</u>

קלט: מספר הפיצ'רים לאחר קידוד וסקיילינג.

שכבות חבויות:שכבה חבויה אחת עם 320 נוירונים ופונקציית אקטיבציה relu.

שכבת פלט: נוירון אחד (כמספר הנתונים אותם רוצים לחזות) ופונקציית אקטיבציה sigmoid משום שמדובר בבעיית קלסיפיקציה.

אופטימייזר: adam

פונקציית הפסד: binary_crossentropy

מדד ביצועים: accuracy, confusion matrix

בס"ד

<u>אימון המודל - הרצה 2:</u>

האימון מתבצע על הנתונים בקבוצת ה - X_train_ready, y_train.

הגדרנו שהמודל ירוץ 30 פעמים (epochs) ושבכל עדכון משקולות משתמשים ב 32 דגימות (batch_size).

הגדרנו ש 20% מהנתונים יוקצו לבדוק את ביצועי המודל במהלך האימון.

<u>הערכת הביצועים של המודל - הרצה 2:</u>

הערכת רשת הנוירונים על קבוצת הנתונים ב test, הרשת לא נחשפה לנתונים הללו. אנחנו בודקות מה אחוזי ההצלחה של המודל ומה אחוזי הכישלון לפי פונקציית השגיאה.

ניתן לראות כי בכל epochs של המודל אחוזי ההצלחה היו גבוהים מאוד (מעל 95% הצלחה).

מצב זה נוצר בעקבות העובדה שהדאטה אינו מאוזן ולכן הגיוני שהמודל יצליח לחזות את הסיכוי לחזות את הסיכוי ללקות בשבץ באחוזים גבוהים.

6. ויזואליזציה

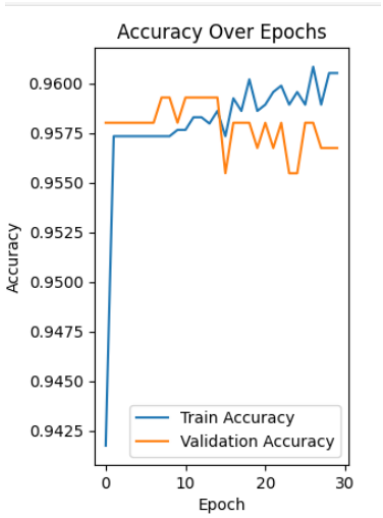
בהערכת הביצועים של המודל (לפי הערכת הצלחה ושגיאה) קיבלנו את הפלט הבא:

```
# Evaluate the trained model on the test dataset and display accuracy
loss, accuracy = model.evaluate(X_test_ready, y_test, verbose=0)
print(f"Test Accuracy: {accuracy:.2f}")
```

```
Test Accuracy: 0.95
```

המודל הצליח לחזות את הסיכוי ללקות בשבץ מוחי ב 95% מהמקרים.

<u>גרף הצלחה:</u>



הגרף של ה Train Accuracy מתחיל נמוך באפוק הראשון וקופץ מהר ל 95%. משם הוא ממשיך לעלות בהדרגה עד סוף האימון.

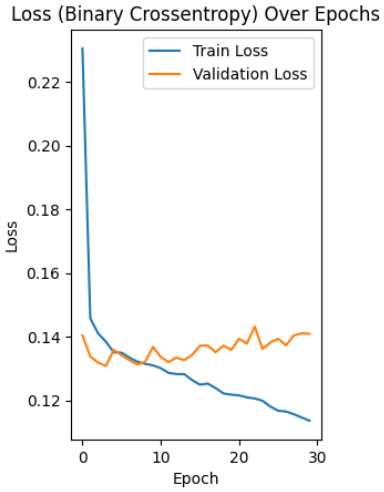
הגרף של ה Validation Accuracy מתחיל יחסית גבוה ולא ממש משתפר לכל אורך 30 האפוקים.

כלומר ישנה התכנסות מהירה, כבר אחרי 2-3 אפוקים המודל הגיע לביצועים המקסימליים שלו על ה־Validation. זה אומר שהרשת למדה את רוב הדפוסים החשובים מהר מאוד. בנוסף

בס"ד

ישנו פער קטן בין Train ל־Validation זה מראה שיש Overfitting קל - המודל למד טוב יותר על על ה־Train מאשר על הדאטה החדש ב test. יציבות ב־Validation Accuracy ניתן לראות כי בערך לאחר האפוק ה 5 המודל לא משתפר משמעותית.

<u>גרף שגיאה:</u>



גרף ה Train Loss מתחיל גבוה ואז יורד בצורה חדה כבר סביב האפוקים הראשונים. אחר כך הוא ממשיך לרדת בצורה יציבה מה שמעיד על כל שהמודל ממשיך ללמוד ולשפר את עצמו על ה־Train. גרף ה Validation Loss מתחיל נמוך יחסית, יורד מעט באפוקים הראשונים ואז מתייצב ואף מתחיל לטפס קלות סביב האפוק ה 10 בערך.
כלומר ישנה התכנסות מהירה, בדיוק כמו בגרף ה Accuracy, גם כאן המודל "תפס" מהר את הדפוסים החשובים, ואחרי האפוק 5 בערך כבר לא משתפר על Validation.
בסוף האימון של המודל קיים פער בין ה Train ל־Validation הפער הזה מראה שיש Overfitting קל - המודל מצליח לזכור טוב יותר את הדאטה של האימון מאשר את הדאטה החדש.

מתוך גרפים אלו ניתן לראות כי שכבה אחת של נוריונים עדיפה מאשר שתי שכבות אך יש להוסיף רגולריזציה כמו עצירה מוקדמת ו dropout.

בס"ד

בהערכת הביצועים של המודל (לפי מטריצת הבלבול) קיבלנו את הפלט הבא:

```
Classification Report:
              precision    recall  f1-score   support

           0      0.958     0.991     0.974       940
           1      0.111     0.024     0.039        42

    accuracy                          0.950       982
   macro avg      0.534     0.508     0.507       982
weighted avg      0.922     0.950     0.934       982

Confusion Matrix:
 [[932   8]
 [ 41   1]]
ROC AUC Score: 0.7645390070921986
```

<u>מטריצת הבלבול:</u>

932 מקרים של "לא שבץ" (0) שנחזו נכון כ"לא שבץ".

8 מקרים של "לא שבץ" שחזו בטעות כ"שבץ".

41 מקרים של "שבץ" שחזו בטעות כ"לא שבץ".

1 מקרים של "שבץ" שחזו נכון כ"שבץ".

מסקנה - עבור מחלקה 0 (לא לקו בשבץ): המודל עובד טוב, כמעט ללא טעויות.

עבור מחלקה 1 (לקו בשבץ): המודל נכשל - זיהה רק מקרה אחד מתוך 42.

המודל לא זיהה את רוב האנשים אשר לקו בשבץ מוחי.

<u>דו"ח סיווג:</u>

1. דיוק כללי accuracy = 95.8%

מקרים בהם אנשים ללא שבץ (932 מתוך 982).המודל חזה תמיד שאין שבץ ולכן קיבלנו אחוזי דיוק גבוהים, אך בפועל המודל נכשל במשימה החשובה שהיא לזהות שבץ.

2. מדדי **Precision, Recall, F1 לפי מחלקות:**

מחלקה 0 (לא שבץ):

Precision = 0.958: מתוך כל מי שהמודל ניבא עליהם "לא שבץ", כ־95.8% באמת לא חלו.

Recall = 0.991: מתוך כל האנשים שבאמת לא חלו (940), המודל זיהה 932 ולא פספס כמעט אף אחד (רק 8 טעויות).

F1 = 0.974: הערך גבוה מאוד כלומר המודל מזהה טוב מקרים של "לא שבץ".

מחלקה 1 (שבץ):

Precision = 0.111: מתוך כל המקרים שהמודל ניבא "שבץ", רק כ־11% באמת היו חולים.

Recall = 0.024: מתוך 42 אנשים שחלו באמת, המודל מצא רק 1.

F1 = 0.039: המודל נכשל לחלוטין בזיהוי חולי שבץ.

בס"ד

3. מדד ROC AUC Score = 0.0.76

המודל כן יודע במידה מסוימת להבדיל בין מי שבסיכון ומי שלא, אבל התוצאה בפועל היא שהמודל תמיד חוזה "לא שבץ".

מסקנה:

המודל במצבו הנוכחי מזהה מצוין מי לא חולה, אבל מתעלם לחלוטין מהחולים האמיתיים (מזהה רק 1 מתוך 42) - ולכן רשת זו לא טובה לתופעה אותה אנו חוקרות.

בס"ד

<u>מחברת שלישית - בניית רשת נוירונים (שכבה חבויה אחת עם רגולריזציה):</u>

1. **טעינת הדאטה לאחר הניקוי (שנעשה במחברת בראשונה)**
2. **הינדוס ובחירת פיצ'רים**
3. **חלוקת הדאטה לשתי קבוצות test,train**

Train - מהווה 80% מהנתונים.

Test - מהווה 20% מהנתונים.

עם stratify לשימור יחס המחלקות.

4. **הכנת דאטה (קידוד משתנים קטגוריים וסטנדרטיזציה)**

חילקנו את הדאטה לשלושה סוגים -

- משתנים קטגוריים - smoking_status, work_type, gender. על פיצ'רים אלו ביצענו one hot encoding.
- משתנים בינארים - על פיצ'רים אלו לא ביצעו קידוד. על חלק מהפיצ'רים ביצענו המרה אשר משמעותה מפורטת בטבלה. את ההמרה עשינו בעזרת פונקציית map, פונקציה אשר ממפה ערכים בינארים במקום הערכים הקיימים.

פיצ'ר	0	1
residence_type	Urban	Rural
ever_married	No	Yes
heart_disease	-	-
hypertension	-	-

- משתנים מספריים - bmi, avg_glucose_level, age. על פיצ'רים אלו ביצענו סטנדרטיזציה (הורדת הממוצע וחלוקה בסטיית התקן).

חיברנו את כל סוגי הפיצ'רים למערך אחד רציף בעזרת np.hstack: הפיצ'רים הקטגוריאליים שעברו קידוד One-Hot, הפיצ'רים הבינאריים שהומרו ל־0/1 והפיצ'רים המספריים שעברו נרמול. בצורה זו יצרנו סט נתונים מאוחד (X_train_ready ו־X_test_ready) שבו כל המאפיינים מוצגים בערכים מספריים בלבד ובפורמט אחיד, כך שהמודל יכול לקבל אותם כקלט ישיר לאימון ולבחינה.

5. **בניית והרצת המודל**

<u>בניית המודל - הרצה 3:</u>

קלט: מספר הפיצ'רים לאחר קידוד וסקיילינג.

שכבות חבויות:שכבה חבויה אחת עם 320 נוירונים ופונקציית אקטיבציה relu. הוספת dropout - בכל אפוק באימון מאפסים אקראית 15% מהנוירונים בשכבה, במטרה למנוע overfitting ותלות בנוירון בודד. משום שמספר הנתונים בהם לקו בשבץ מוחי הוא קטן מאוד בחרנו ב 15% ובכך לא נאבד דגימות אלו.

שכבת פלט: נוירון אחד (כמספר הנתונים אותם רוצים לחזות) ופונקציית אקטיבציה sigmoid משום שמדובר בבעיית קלסיפיקציה.

אופטימייזר: adam

פונקציית הפסד: binary_crossentropy

מדד ביצועים: accuracy, confusion matrix

<u>אימון המודל - הרצה 3:</u>

האימון מתבצע על הנתונים בקבוצת ה - X_train_ready, y_train.

הגדרנו שהמודל ירוץ 30 פעמים (epochs) ושבכל עדכון משקולות משתמשים ב 32 דגימות (batch_size).

הגדרנו ש 20% מהנתונים יוקצו לבדוק את ביצועי המודל במהלך האימון.

בעקבות תוצאות רשת הנוריונים הקודמת החלטנו לעשות earlyStopping אחרי 5 אפוקים ברציפות בהם אין שיפור, כך שהרשת תפסיק אוטומטית, ובכך חסכנו זמן חישוב ומנענו overfitting.

עצירה מוקדמת עשינו עבור הערך של val_loss (השגיאה על סט הולידציה אחרי כל אפוק).

בחרנו בקריטריון זה משום שאם הוא מפסיק להשתפר, כנראה שהמודל התחיל לשנן את ה-Train. אחרי העצירה, המודל מחזיר את המשקולות מהאפוק שבו התקבל val_loss הנמוך ביותר.

כך נמנע מצב שבו המודל נשאר עם משקולות פחות טובות מהשיא שלו.

<u>הערכת הביצועים של המודל - הרצה 3:</u>

הערכת רשת הנוירונים על קבוצת הנתונים ב test, הרשת לא נחשפה לנתונים הללו. אנחנו בודקות מה אחוזי ההצלחה של המודל ומה אחוזי הכישלון לפי פונקציית השגיאה.

ניתן לראות כי בכל epochs של המודל אחוזי ההצלחה היו גבוהים מאוד (מעל 96% הצלחה).

מצב זה נוצר בעקבות העובדה שהדאטה אינו מאוזן ולכן הגיוני שהמודל יצליח לחזות את הסיכוי לחזות את הסיכוי ללקות בשבץ באחוזים גבוהים.

6. ויזואליזציה

בהערכת הביצועים של המודל (לפי הערכת הצלחה ושגיאה) קיבלנו את הפלט הבא:

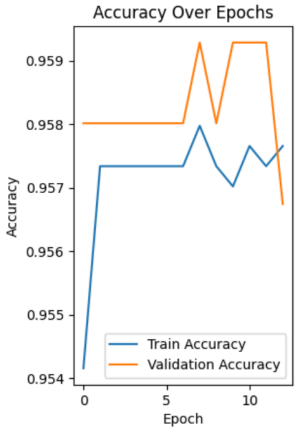
```
# Evaluate the trained model on the test dataset and display accuracy
loss, accuracy = model.evaluate(X_test_ready, y_test, verbose=0)
print(f"Test Accuracy: {accuracy:.2f}")
```

```
Test Accuracy: 0.96
```

המודל הצליח לחזות את הסיכוי ללקות בשבץ מוחי ב 96% מהמקרים.

בס"ד

גרף הצלחה:

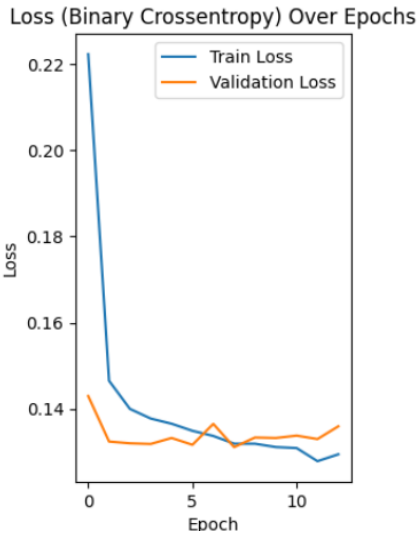


הגרף של ה Train Accuracy מתחיל נמוך באפוק הראשון וקופץ מהר ל 95%. ניתן לראות שסביב האפוק ה6 הגרף מתחיל לרדת ולעלות שוב פעם סביב האפוק ה11.

הגרף של ה Validation Accuracy מתחיל יחסית גבוה, נשאר סטטי עד האפוק ה6 ומשם חלה עלייה משמעותית וירידה חדה סביב האפוק ה11.

כלומר ישנה **התכנסות מהירה,** כבר אחרי 2-3 אפוקים המודל הגיע לביצועים המקסימליים שלו על ה־Validation. זה אומר שהרשת למדה את רוב הדפוסים החשובים מהר מאוד. בנוסף ישנו **פער קטן בין Train ל־Validation** זה מראה שיש **Overfitting קל** - המודל למד טוב יותר על על ה־Train מאשר על הדאטה החדש ב test. **יציבות ב־Validation Accuracy** ניתן לראות כי בערך לאחר האפוק ה 5 המודל **לא משתפר משמעותית.**

גרף שגיאה:



גרף ה Train Loss מתחיל גבוה ואז יורד בצורה חדה כבר סביב האפוקים הראשונים. אחר כך הוא ממשיך לרדת בצורה יציבה מה שמעיד על כל שהמודל ממשיך ללמוד ולשפר את עצמו על ה־Train. גרף ה Validation Loss מתחיל נמוך יחסית, יורד מעט באפוקים הראשונים ואז מתייצב ואף מתחיל לטפס קלות סביב האפוק ה 10 בערך.

כלומר ישנה התכנסות מהירה, בדיוק כמו בגרף ה Accuracy, גם כאן המודל "תפס" מהר את

הדפוסים החשובים, ואחרי האפוק 5 בערך כבר לא משתפר על Validation.
בסוף האימון של המודל קיים פער בין ה Train ל־Validation הפער הזה מראה שיש Overfitting
קל - המודל מצליח לזכור טוב יותר את הדאטה של האימון מאשר את הדאטה החדש.

מתוך גרפים אלו ניתן לראות כי שכבה אחת של נוריונים ושיטות רגולריזציה שונות (עצירה מוקדמת ו dropout) לא מספיקות במקרה של דאטה לא מאוזן ועל כן יש לאזן את הדאטה בעזרת שיטות שונות כמו SMOTE.

בהערכת הביצועים של המודל (לפי מטריצת הבלבול) קיבלנו את הפלט הבא:

```
Classification Report:
              precision    recall  f1-score   support

           0      0.957     0.999     0.978       940
           1      0.000     0.000     0.000        42

    accuracy                          0.956       982
   macro avg      0.479     0.499     0.489       982
weighted avg      0.916     0.956     0.936       982

Confusion Matrix:
 [[939   1]
 [ 42   0]]
ROC AUC Score: 0.7949594731509625
```

<u>מטריצת הבלבול:</u>

939 מקרים של "לא שבץ" (0) שנחזו נכון כ"לא שבץ".

1 מקרים של "לא שבץ" שחזו בטעות כ"שבץ".

42 מקרים של "שבץ" שחזו בטעות כ"לא שבץ".

0 מקרים של "שבץ" שחזו נכון כ"שבץ".

מסקנה - עבור מחלקה 0 (לא לקו בשבץ): המודל עובד טוב, כמעט ללא טעויות.
עבור מחלקה 1 (לקו בשבץ): המודל נכשל - לא זיהה אף מקרה אמיתי של שבץ!

<u>דו"ח סיווג:</u>

1. דיוק כללי accuracy = 95.6%
מקרים בהם אנשים ללא שבץ (939 מתוך 982). המודל חזה תמיד שאין שבץ ולכן קיבלנו אחוזי דיוק גבוהים, אך בפועל המודל נכשל במשימה החשובה שהיא לזהות שבץ.
2. מדדי **Precision, Recall, F1 לפי מחלקות:**

מחלקה 0 (לא שבץ):

Precision = 0.957: מתוך כל מי שהמודל ניבא עליהם "לא שבץ", כ־95.7% באמת לא חלו.
Recall = 0.999: מתוך כל האנשים שבאמת לא חלו (940), המודל זיהה 939 ולא פספס כמעט אף אחד.
F1 = 0.978: הערך גבוה מאוד כלומר המודל מזהה טוב מקרים של "לא שבץ".

בס"ד

מחלקה 1 (שבץ):

Precision = 0.000: מתוך כל המקרים שהמודל ניבא "שבץ".

Recall = 0.000: מתוך 42 אנשים שחלו באמת, המודל לא זיהה אף אחד.

F1 = 0.000: המודל נכשל לחלוטין בזיהוי חולי שבץ.

3. מדד ROC AUC Score =0.795

המודל כן יודע במידה מסוימת להבדיל בין מי שבסיכון ומי שלא, אבל התוצאה בפועל היא שהמודל תמיד חוזה "לא שבץ".

מסקנה:

המודל במצבו הנוכחי מזהה מצוין מי לא חולה, אבל מתעלם לחלוטין מהחולים האמיתיים (לא מזהה בכלל) - ולכן רשת זו לא טובה לתופעה אותה אנו חוקרות.

SMOTE

כמענה לבעיית חוסר האיזון בנתונים בחרנו להשתמש ב **SMOTE**:

שיטת למידת מכונה מפוקחת לטיפול בבעיה של חוסר איזון בין מחלקות. מספר הדוגמאות שנוספו הן כמספר הנתונים שיש בקבוצת הרוב.

במקום לשכפל דוגמאות קיימות ממחלקת המיעוט, SMOTE מייצרת דוגמאות חדשות באופן סינתטי באופן הבא:

1. בחירת דגימה ממחלקת המיעוט.
2. איתור **5 שכנים קרובים** (k=5). (דווקא 5 כי זה ברירת המחדל)
3. בחירת שכן אחד באקראי.
4. יצירת דגימה סינתטית חדשה על קו המחבר בין הדגימה לשכן.

<u>נדגיש כי השימוש ב־SMOTE בוצע רק על קבוצת ה־Train</u>, במטרה לשמור על קבוצת ה־Test כמייצגת את הנתונים האמיתיים בשטח. בדרך זו המודל נבדק על נתונים מקוריים בלבד, ללא דגימות סינתטיות, וכך מתקבלת הערכה אמינה של הביצועים.

בס"ד

<u>מחברת רביעית - בניית רשת נוירונים (עם SMOTE):</u>

1. **טעינת הדאטה לאחר הניקוי (שנעשה במחברת בראשונה)**
2. **הינדוס ובחירת פיצ'רים**
3. **חלוקת הדאטה לשתי קבוצות test,train**

Train - מהווה 80% מהנתונים.

Test - מהווה 20% מהנתונים.

עם stratify לשימור יחס המחלקות.

4. **הכנת דאטה (קידוד משתנים קטגוריים וסטנדרטיזציה)**

חילקנו את הדאטה לשלושה סוגים -

- משתנים קטגוריים - smoking_status, work_type, gender. על פיצ'רים אלו ביצענו one hot encoding.
- משתנים בינארים - על פיצ'רים אלו לא ביצעו קידוד. על חלק מהפיצ'רים ביצענו המרה אשר משמעותה מפורטת בטבלה. את ההמרה עשינו בעזרת פונקציית map, פונקציה אשר ממפה ערכים בינארים במקום הערכים הקיימים.

פיצ'ר	0	1
residence_type	Urban	Rural
ever_married	No	Yes
heart_disease	-	-
hypertension	-	-

- משתנים מספריים - bmi, avg_glucose_level, age. על פיצ'רים אלו ביצענו סטנדרטיזציה (הורדת הממוצע וחלוקה בסטיית התקן).

חיברנו את כל סוגי הפיצ'רים למערך אחד רציף בעזרת np.hstack: הפיצ'רים הקטגוריאליים שעברו קידוד One-Hot, הפיצ'רים הבינאריים שהומרו ל־0/1 והפיצ'רים המספריים שעברו נרמול. בצורה זו יצרנו סט נתונים מאוחד (X_train_ready ו־X_test_ready) שבו כל המאפיינים מוצגים בערכים מספריים בלבד ובפורמט אחיד, כך שהמודל יכול לקבל אותם כקלט ישיר לאימון ולבחינה.

5. **בניית והרצת המודל**

<u>בניית המודל - הרצה 4:</u>

קלט: מספר הפיצ'רים לאחר קידוד, סקיילינג וSMOTE.

שכבות חבויות: שכבה חבויה אחת עם 320 נוירונים ופונקציית אקטיבציה relu. הוספת dropout - בכל אפוק באימון מאפסים אקראית 15% מהנוירונים בשכבה, במטרה למנוע overfitting ותלות בנוירון בודד. משום שמספר הנתונים בהם לקו בשבץ מוחי הוא קטן מאוד בחרנו ב 15% ובכך לא נאבד דגימות אלו.

שכבת פלט: נוירון אחד (כמספר הנתונים אותם רוצים לחזות) ופונקציית אקטיבציה sigmoid משום שמדובר בבעיית קלסיפיקציה.

אופטימייזר: adam

פונקציית הפסד: binary_crossentropy

מדד ביצועים: accuracy, confusion matrix

<u>אימון המודל - הרצה 4:</u>

האימון מתבצע על הנתונים בקבוצת ה - X_train_bal, y_train_bal.

הגדרנו שהמודל ירוץ 30 פעמים (epochs) ושבכל עדכון משקולות משתמשים ב 32 דגימות (batch_size).

הגדרנו ש 20% מהנתונים יוקצו לבדוק את ביצועי המודל במהלך האימון.

בעקבות תוצאות רשת הנוריונים הקודמת החלטנו לעשות earlyStopping אחרי 5 אפוקים ברציפות בהם אין שיפור, כך שהרשת תפסיק אוטומטית, ובכך חסכנו זמן חישוב ומנענו overfitting.

עצירה מוקדמת עשינו עבור הערך של val_loss (השגיאה על סט הולידציה אחרי כל אפוק).

בחרנו בקריטריון זה משום שאם הוא מפסיק להשתפר, כנראה שהמודל התחיל לשנן את ה-Train. אחרי העצירה, המודל מחזיר את המשקולות מהאפוק שבו התקבל val_loss הנמוך ביותר.

כך נמנע מצב שבו המודל נשאר עם משקולות פחות טובות מהשיא שלו.

<u>הערכת הביצועים של המודל - הרצה 4:</u>

הערכת רשת הנוירונים על קבוצת הנתונים ב test, הרשת לא נחשפה לנתונים הללו. אנחנו בודקות מה אחוזי ההצלחה של המודל ומה אחוזי הכישלון לפי פונקציית השגיאה.

ניתן לראות כי בכל epochs של המודל אחוזי ההצלחה היו גבוהים מאוד (מעל 73% הצלחה).

6. ויזואליזציה

בהערכת הביצועים של המודל (לפי הערכת הצלחה ושגיאה) קיבלנו את הפלט הבא:

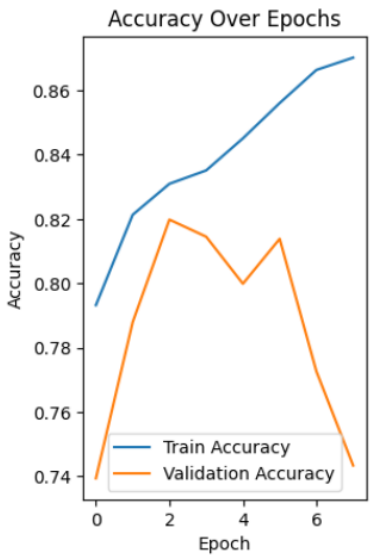
```
# Evaluate the trained model on the test dataset and display accuracy
loss, accuracy = model.evaluate(X_test_ready, y_test, verbose=0)
print(f"Test Accuracy: {accuracy:.2f}")
```

```
Test Accuracy: 0.82
```

המודל הצליח לחזות את הסיכוי ללקות בשבץ מוחי ב 82% מהמקרים.

בס"ד

<u>גרף הצלחה:</u>



גרף ה Train Accuracy הולך ועולה באופן יציב כמעט בכל אפוק עד שהוא מגיע לאזור ה87%.
גרף ה Validation Accuracy מתחיל נמוך, עולה עד אפוק 2 ומתחיל לרדת בצורה חדה עד הסוף.
כלומר הגרף מציג תופעה של overfitting. המודל מצליח לזכור טוב יותר את דאטה בקבוצת האימון, ולכן train accuracy עולה.אבל על הדאטה החדש (validation) הוא כבר לא מצליח לשפר ביצועים, ואפילו מתדרדר, סימן שהמודל מתחיל להתאים את עצמו יותר מדי לרעש של ה־train.

בס"ד

<u>גרף שגיאה:</u>

גרף ה Train Loss הולך ויורד, סימן לכך שהמודל לומד את הדאטה בקבוצת האימון טוב יותר בכל אפוק.
גרף ה Validation Loss בהתחלה יורד עד סביב האפוק השני ואז מתחיל לעלות.
כלומר הגרף מתאר תופעה של overfitting, בזמן ש־Train Loss ממשיך לרדת, גרף ה Validation Loss עולה. זוהי הנקודה שבה המודל מתחיל לשנן פרטים ספציפיים של train ולא מצליח ללמוד טוב ל־val.

ראינו בגרפים שיש overfitting לכן בחרנו להשתמש באחת מכמה השיטות לטיפול ולמניעת בעיה זו והיא הפיכת המודל למורכב פחות כלומר להוריד את מספר הנוירונים ברשת / או להגדיל dropout.

בהערכת הביצועים של המודל (לפי מטריצת הבלבול) קיבלנו את הפלט הבא:

```
Classification Report:
              precision    recall  f1-score   support

           0      0.977     0.831     0.898       940
           1      0.131     0.571     0.213        42

    accuracy                          0.820       982
   macro avg      0.554     0.701     0.556       982
weighted avg      0.941     0.820     0.869       982

Confusion Matrix:
 [[781 159]
 [ 18  24]]
ROC AUC Score: 0.7976697061803445
```

בס"ד

<u>מטריצת הבלבול:</u>

781 מקרים של "לא שבץ" (0) שנחזו נכון כ"לא שבץ".

159 מקרים של "לא שבץ" שחזו בטעות כ"שבץ".

18 מקרים של "שבץ" שחזו בטעות כ"לא שבץ".

24 מקרים של "שבץ" שחזו נכון כ"שבץ".

מסקנה - עבור מחלקה 0 (לא לקו בשבץ): המודל עובד טוב, אך עדיין יש 159 התראות שווא.
עבור מחלקה 1 (לקו בשבץ): המודל מזהה חלק מהחולים (24 מתוך 42) אך עדיין מפספס 18 מהם.

<u>דו"ח סיווג:</u>

1. דיוק כללי accuracy = 82%

2. מדדי **Precision, Recall, F1 לפי מחלקות:**

מחלקה 0 (לא שבץ):

Precision = 0.977: מתוך כל מי שהמודל ניבא עליהם "לא שבץ", כ־97.7% באמת לא חלו.
Recall = 0.831: מתוך כל האנשים שבאמת לא חלו (940), המודל זיהה 781 מקרים.
F1 = 0.898.

מחלקה 1 (שבץ):

Precision = 0.131: מתוך כל המקרים שהמודל ניבא "שבץ", רק 13.1% באמת חולים.
כלומר הרבה התראות שווא.
Recall = 0.571: מתוך 42 אנשים שחלו באמת, המודל זיהה 24 חולים.
F1 =0.213.

3. מדד ROC AUC Score =0.798
מדד זה מעיד על כך שהמודל יודע להבחין טוב יותר בין שתי הקבוצות - אפילו עם הדאטה הלא מאוזן.

מסקנה:

המודל מצליח לזהות בצורה טובה יחסית את האנשים שלא לקו בשבץ, למרות שעדיין קיימות טעויות של "אזעקות שווא".
בנוסף, המודל מצליח לזהות חלק משמעותי מהאנשים שלקו בשבץ (מעל מחצית מהם). עם זאת, הדיוק בזיהוי חולים עדיין נמוך, כלומר חלק גדול מהתחזיות של "שבץ" הן התראות שווא.

באופן כללי, זהו המודל הראשון שבנינו שמציג איזון ראשוני: הוא לא מתעלם מהחולים, אבל עדיין זקוק לשיפור ניכר כדי להגיע לרמה שבה אפשר יהיה להסתמך עליו בכלי חיזוי רפואי אמיתי.

בס"ד

רגרסיה לוגיסטית:

מודל סטטיסטי שמטרתו לחזות הסתברות להשתייכות של דגימה לקטגוריה מסוימת.
משמשת בעיקר לסיווג (classification), לא לחיזוי ערך רציף.
רגרסיה לוגיסטית היא מן מקרה פרטי של רשת נוירונים:

- רשת עם שכבת קלט אחת, שכבת פלט אחת ובלי שכבות חבויות.
- פונקציית האקטיבציה בשכבת הפלט היא Sigmoid (לבעיה בינארית) או Softmax (לריבוי קטגוריות).
- רגרסיה לוגיסטית היא ה"נוירון" הבסיסי ביותר ברשת נוירונים.

<u>מחברת חמישית (בעיה) + מחברת שישית (פתרון) - רגרסיה לוגיסטית:</u>

חיפשנו דרך להבין מי הם הפיצ'רים בעלי ההשפעה הרבה ביותר על הסיכוי ללקות בשבץ מוחי.

שלבים בבניית המודל:

1. טעינת הקובץ הנקי.
2. הכנת הדאטה - כמו במחברת 1 - קידוד וסקליינג.
3. חלוקה ל train ו test.
4. שימוש ברגרסיה לוגיסטית:

הגדרנו את מספר החזרות המקסימלי שהמודל יבצע על מנת למצוא את הפתרון הטוב ביותר (בחרנו ב-1,000 , במספר גבוה כי הדאטה שלנו לא מאוזן ורצינו לבדוק מההתחלה מקרה קצה).
נרצה לחזות את פיצ'רים בעלי המשקל הגבוה ביותר - המשפיעים ביותר - על הסיכוי ללקות בשבץ מוחי.

5. מטריצת הבלבול ודוח סיווג:

```
Classification Report:
              precision    recall  f1-score   support

           0      0.957     1.000     0.978       940
           1      0.000     0.000     0.000        42

    accuracy                          0.957       982
   macro avg      0.479     0.500     0.489       982
weighted avg      0.916     0.957     0.936       982

Confusion Matrix:
 [[940   0]
 [ 42   0]]
ROC AUC Score: 0.8136524822695036
```

בס"ד

<u>מטריצת הבלבול:</u>

940 מקרים של "לא שבץ" (0) שנחזו נכון כ"לא שבץ".

0 מקרים של "לא שבץ" שחזו בטעות כ"שבץ".

42 מקרים של "שבץ" שחזו בטעות כ"לא שבץ".

0 מקרים של "שבץ" שחזו נכון כ"שבץ".

מסקנה - עבור מחלקה 0 (לא לקו בשבץ): המודל עובד טוב, ללא התראות שווא.

עבור מחלקה 1 (לקו בשבץ): המודל נכשל - הוא מפספס את כל החולים.

<u>דו"ח סיווג:</u>

1. דיוק כללי accuracy = 95.7%

2. מדדי **Precision, Recall, F1 לפי מחלקות:**

מחלקה 0 (לא שבץ):

Precision = 0.957: מתוך כל מי שהמודל ניבא עליהם "לא שבץ", כ־95.7% באמת לא חלו.

Recall = 1.000: מתוך כל האנשים שבאמת לא חלו (940), המודל זיהה את כל המקרים.

F1 = 0.978.

מחלקה 1 (שבץ):

Precision = 0.000: המודל לא חזה אף חולה.

Recall = 0.000: מתוך 42 אנשים שחלו באמת, המודל לא זיהה אף חולה.

F1 =0.000.

3. מדד ROC AUC Score =0.814

מדד זה מעיד שלמרות העובדה שהמודל לא מצליח לזהות מקרים חיוביים הוא עדיין יכול לסווג בין הקבוצות השונות.

נציין שבמחברת זו המטרה היא לבחון מהם הפיצ'רים בעלי המשקל הרב ביותר, המשפיעים ביותר על הסיכוי ללקות בשבץ מוחי.

בס"ד

הסבר על פיצ'רים נבחרים:

המספרים המייצגים את הפיצ'רים שקיבלנו הם המקדמים של הרגרסיה הלוגיסטית:

פיצ'ר עם **משקל חיובי גדול** = מעלה את הסיכון לשבץ.

פיצ'ר עם **משקל שלילי** = מוריד את הסיכון.

ככל שהמשקל רחוק מ- 0 - כך השפעת הפיצ'ר חזקה יותר.

```
▲ Features that increase stroke risk:
age                                 1.658722
hypertension                        0.655611
work_type_children                  0.315130
smoking_status_smokes               0.253713
heart_disease                       0.237036
avg_glucose_level                   0.207183
work_type_Private                   0.072334
bmi                                 0.016949
smoking_status_formerly smoked      0.010707
dtype: float64

▼ Features that decrease stroke risk:
work_type_Self-employed        -0.288544
ever_married                   -0.268836
smoking_status_Unknown         -0.248120
work_type_Govt_job             -0.137303
smoking_status_never smoked    -0.103591
gender_Male                    -0.071256
work_type_Never_worked         -0.048908
gender_Other                   -0.012927
Residence_type                 -0.008990
gender_Female                  -0.003109
dtype: float64
```

<u>פיצ'רים המגבירים את הסיכוי (מקדם חיובי):</u>

1. גיל: ככל שהגיל גבוה יותר, הסיכוי ללקות בשבץ עולה משמעותית. זהו הפיצ'ר המשפיע ביותר.
2. לחץ דם: ככל שלחץ הדם גבוה יותר, הסיכוי ללקות בשבץ עולה.
3. במקרה בו האדם הוא ילד (לא מועסק), הסיכוי ללקות בשבץ מוחי עולה - נתון שנראה חריג, **נבחן אותו בהמשך.**
4. אדם מעשן: נמצא בסיכון גבוה יותר ללקות בשבץ מוחי.
5. אדם עם מחלות רקע (גלוקוז גבוה ומחלות לב): נמצא בסיכון גבוה יותר ללקות בשבץ מוחי.
6. עבודה בשוק הפרטי: מעלה במעט את הסיכוי ללקות בשבץ מוחי.
7. הBMI: אדם בעל עודף משקל בעל סיכוי גבוה יותר ללקות בשבץ.
8. אדם שעישן בעבר: הסיכוי ללקות בשבץ מוחי עולה מעט.

בס"ד

<u>פיצ'רים המפחיתים את הסיכוי (מקדם חיובי):</u>

1. סוג עבודה עצמאי: מפחית את הסיכוי ללקות בשבץ מוחי.
2. אדם נשוי / היה נשוי בעבר: מפחית את הסיכוי ללקות בשבץ מוחי.
3. אנשים ללא נתון על סטטוס העישון בעלי סיכון מופחת ללקות בשבץ מוחי (זה דומה לסטטוס 'לא מעשן').
4. עבודה במגזר הממשלתי
5. אנשים שלא עישנו מעולם
6. גברים: בעלי סיכוי מופחת ללקות בשבץ מוחי - לעומת נשים.
7. אנשים שלא עבדו מעולם
8. מגדר - אחר
9. מקום מגורים כמעט ולא משפיע
10. נשים

ניתן לראות כי ישנם רק 9 פיצ'רים אשר מגבירים את הסיכוי לחלות בשבץ מוחי.

<u>בעקבות המקרה השלישי בפיצ'רים המגבירים החלטנו לבדוק מדוע פיצ'ר זה הוא בעל משקל רב.</u>

בהתחלה בדקנו מה טווח הגילאים של האנשים שסימנו work_type=children במטרה לבדוק שהם אכן ילדים.

לאחר מכן בדקנו כמה מתוכם אכן לקו בשבץ מוחי על מנת לוודא את אמינות הנתונים.

ראינו כי הנתונים אכן תקינים וגילאי הילדים בין 1-16 והסיכוי ללקות בשבץ מוחי הוא נמוך מאוד.

בשביל לפתור את הבעיה החלטנו להוציא את הקטגוריה של ילדים בחלק של ה work_type.

שלבים בביצוע השינוי:

1. הוספנו את השורות הבאות: שינינו את הקידוד של המשתנים הקטגוריים

```
X_train_cat_df = pd.DataFrame(X_train_cat, columns=ohe.get_feature_names_out(cat_cols), index=X_train.index)
X_test_cat_df = pd.DataFrame(X_test_cat, columns=ohe.get_feature_names_out(cat_cols), index=X_test.index)
```

```
if 'work_type_children' in X_train_cat_df.columns:
    X_train_cat_df.drop(columns=['work_type_children'], inplace=True)
if 'work_type_children' in X_test_cat_df.columns:
    X_test_cat_df.drop(columns=['work_type_children'], inplace=True)
```

בחלק הזה של הקוד מקודדים את כל המשתנים הקטגוריים, שומרים על כל הקטגוריות, אבל מוחקים באופן יזום רק את הקטגוריה work_type_children - כדי למנוע השפעה שגויה על המודל.

2. הוספנו SMOTE: המטרה היא לאזן את המודל, ביצענו את הSMOTE רק על קבוצת הtrain במטרה שהנתונים בקבוצת הtest יהיו הנתונים המקוריים והמודל יחזה את הסיכוי ללקות בשבץ על הדאטה המקורי שלא עבר שינוי והתאמה.
3. הוספנו את השורה:

y_pred = log_reg.predict(X_test_ready)

השורה מחזירה את התחזיות של המודל לגבי מי צפוי ללקות בשבץ (1) ומי לא (0) בtest.

מטרת שורה זו היא שנוכל להשוות בין התחזיות לבין התוויות האמיתיות.

בס"ד

דו"ח הסיווג לאחר השינויים:

```
Classification Report:
              precision    recall  f1-score   support

           0      0.980     0.770     0.862       940
           1      0.111     0.643     0.189        42

    accuracy                          0.765       982
   macro avg      0.545     0.707     0.526       982
weighted avg      0.943     0.765     0.834       982

Confusion Matrix:
 [[724 216]
 [ 15  27]]
ROC AUC Score: 0.8113475177304964
```

<u>מטריצת הבלבול:</u>

724 מקרים של "לא שבץ" (0) שנחזו נכון כ"לא שבץ".

216 מקרים של "לא שבץ" שחזו בטעות כ"שבץ".

15 מקרים של "שבץ" שחזו בטעות כ"לא שבץ".

27 מקרים של "שבץ" שחזו נכון כ"שבץ".

מסקנה - עבור מחלקה 0 (לא לקו בשבץ): המודל עובד טוב בזיהוי אנשים בריאים.

עבור מחלקה 1 (לקו בשבץ): המודל מצליח לזהות 27 חולים מתוך 42. .

<u>דו"ח סיווג:</u>

1. דיוק כללי accuracy = 76.5%
2. מדדי **Precision, Recall, F1 לפי מחלקות:**

מחלקה 0 (לא שבץ):

Precision = 0.980: מתוך כל מי שהמודל ניבא עליהם "לא שבץ", כ־98% באמת לא חלו.

Recall = 0.770: מתוך כל האנשים שבאמת לא חלו (940), המודל זיהה 724 מהמקרים.

F1 = 0.862.

מחלקה 1 (שבץ):

Precision = 0.111: המודל מזהה רק 11.1% מהמקרים של 'שבץ'.

Recall = 0.643: מתוך 42 אנשים שחלו באמת, המודל זיהה 27.

F1 = 0.189.

3. מדד ROC AUC Score = 0.811

המודל יודע להבדיל בין חולים לבריאים יחסית טוב, אך עדיין הוא מנחש אנשים בריאים כחולים - מצב זה עדיף על פני החמצת אנשים חולים וסיווגם כבריאים (ברפואה).

בס"ד

בגרסה הקודמת, הפיצ'ר work_type_children קיבל מקדם חיובי גבוה יחסית במודל, מה שמראה על תרומתו להחלטה. לאחר הסרת הפיצ'ר, המודל "פיצה" על המידע דרך פיצ'רים אחרים וההסתברויות והחיזויים כמעט שלא השתנו. לכן מטריצת הבלבול נותרה זהה למעט הבדל של חיזוי יחיד, ללא שינוי בביצועים הכוללים.

ניתן לראות שלאחר השינוי יש משקל גבוה יותר למקדמי הפיצ'רים ברגרסיה הלוגיסטית.

```
▲ Features that increase stroke risk:
age                                2.156918
hypertension                       0.776008
smoking_status_smokes              0.323882
avg_glucose_level                  0.208180
heart_disease                      0.197460
smoking_status_formerly smoked     0.159194
gender_Female                      0.074242
bmi                                0.048451
Residence_type                     0.041340
dtype: float64

▼ Features that decrease stroke risk:
work_type_Govt_job            -1.209302
work_type_Self-employed       -1.183666
work_type_Private             -0.666253
smoking_status_Unknown        -0.403305
ever_married                  -0.291472
smoking_status_never smoked   -0.249611
gender_Male                   -0.191424
work_type_Never_worked        -0.135706
gender_Other                  -0.052658
dtype: float64
```

<u>מסקנות על הדאטה בזכות שימוש ברגרסיה לוגיסטית:</u>

- **הגיל** הוא הגורם המשמעותי ביותר בסיכוי ללקות בשבץ.
- הרגלי עישון, לחץ דם ורמת גלוקוז גבוהים מהווים גורם משמעותי נוסף לסיכוי ללקות בשבץ מוחי.
- חלק ממשתני הרקע (סוג עבודה, מגורים, מצב משפחתי) משפיעים אך בעוצמה פחותה.
- נתון נוסף ומעניין הוא כי סוג מקום העבודה מפחית את הסיכוי ללקות בשבץ מוחי אך בעוצמות שונות.

בס"ד

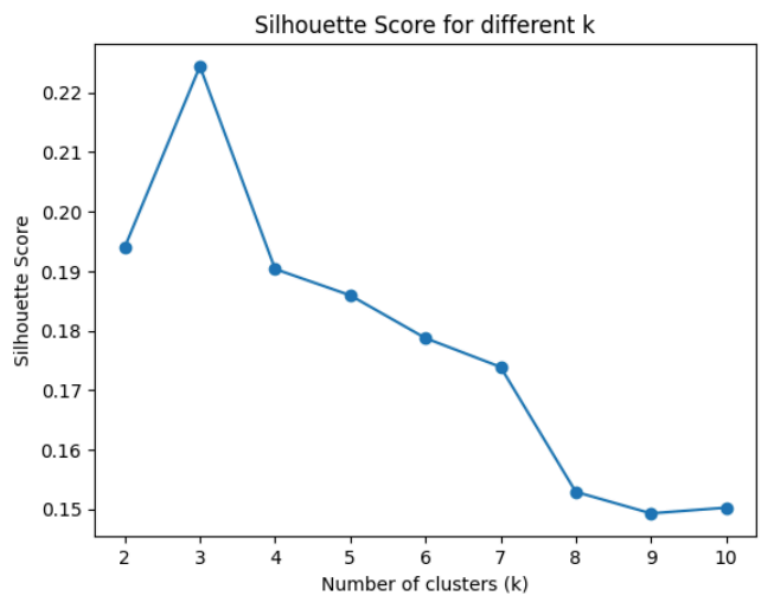
מחברת שביעית - למידה לא מפוקחת:

לפני הפעלת האלגוריתמים על הדאטה עשינו את השלבים הבאים:

טעינת הדאטה הנקי, הינדוס, בחירת פיצ'רים וקידוד הדאטה.

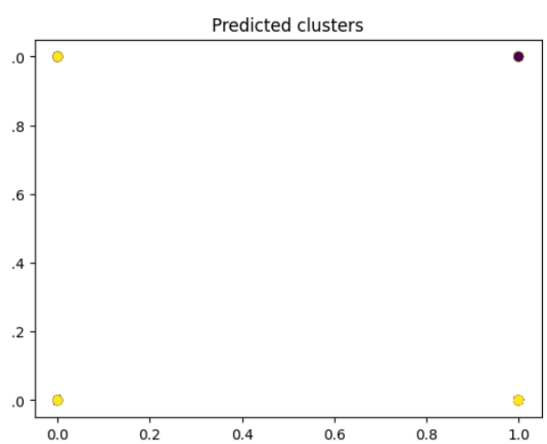
על מנת לבחור את מס' האשכולות הטוב ביותר לדאטה השתמשנו במדד silhouette.

```
sil_scores = []
k_values = range(2, 11)

for k in k_values:
    kmeans = KMeans(n_clusters=k, random_state=42)
    labels = kmeans.fit_predict(X_all)
    score = silhouette_score(X_all, labels)
    sil_scores.append(score)

plt.plot(k_values, sil_scores, marker='o')
plt.xlabel("Number of clusters (k)")
plt.ylabel("Silhouette Score")
plt.title("Silhouette Score for different k")
plt.show()
```



לאחר הרצת קטע הקוד הנ"ל חיפשנו את הערך הגבוה ביותר של מדד סילואט - לפי מדד זה הוא 3, כלומר מספר האשכולות הטוב ביותר הוא 3.

לא ניתן לבצע אשכול על הדאטה הזה משום שהוא מורכב מיותר משני מימדים. אלגוריתם kmeans עובד על שניים או שלושה מימדים ולכן לא ניתן לבצע אשכול ולהציג אותו ויזואלית בגרף של שני מימדים.



בס"ד

יש לבצע אלגוריתם PCA ורק לאחר מכן לעשות kmeans.

בחרנו לא לחפש את מספר המימדים אלא לנתח מה הם הפיצ'רים שהכי תרמו לכל רכיב PCA.
לשם כך ביצענו את השלבים הבאים:

1. שמירת שמות כל הפיצ'רים - כדי לקשר בין הפיצ'רים המקוריים לבין ה PCA.
2. טעינת המשקלים של הפיצ'רים ברכיבים - כל ערך מעיד על עד כמה פיצ'ר מסויים תורם ל PCA1, PCA2.
   PCA1 - זה הראשי, הערכים העצמיים הגדולים ביותר (השונויות).
   PCA2 - ציר משני, ציר מאונך לציר הראשי.
3. בניית DataFrame שמסדר את התרומה של כל פיצ'ר לצירים.
4. מיון לפי גודל התרומה, סידור הפיצ'רים לפי השפעתם פעם ראשונה כאשר PC1 הוא הרכבי הראשני ופעם שנייה ש PC2 הוא הרכבי הראשי:
   <u>טבלה ראשונה:</u>

Top 10 Features Contributing to PC1

	PC1	PC2	abs_PC1	abs_PC2
age	0.63	-0.18	0.63	0.18
bmi	0.54	-0.31	0.54	0.31
avg_glucose_level	0.40	0.90	0.40	0.90
ever_married	0.24	-0.10	0.24	0.10
work_type_children	-0.18	0.11	0.18	0.11
smoking_status_Unknown	-0.17	0.09	0.17	0.09
work_type_Private	0.08	-0.09	0.08	0.09
smoking_status_never smoked	0.07	-0.05	0.07	0.05
work_type_Self-employed	0.07	-0.01	0.07	0.01
smoking_status_formerly smoked	0.07	-0.01	0.07	0.01

כל שורה בטבלה היא פיצ'ר מקורי.
בעמודות PC1 וPC2 מוצגים המקדמים, כלומר המשקל הלינארי של הפיצ'ר באותו הרכיב.
בעמודות **abs_PC1** ו־**abs_PC2** מוצגים הערך המוחלט של התרומה.

ערך גבוה = הפיצ'ר בעל ההשפעה הרבה ביותר על הרכיב.
ערך קרוב ל0 = הפיצ'ר כמעט לא משפיע.
סימון + / - = כיוון התרומה.

רכיב ראשון PC1:
גיל: המשתנה המשמעותי ביותר ברכיב PC1.
הBMI, גלוקוז, סטטוס זוגי נשוי / היה נשוי בעבר: תורמים משמעותית.

- המשמעות: PC1 הוא בעיקר ציר בריאותי

בס"ד

רכיב שני PC2:

הBMI, הגיל והסטטוס הזוגי נשוי / היה נשוי בעבר בעלי תרומה שלילית.

ילדים וסטטוס עישון לא ידוע בעלי תרומה חיובית.

- המשמעות: PC2 מפריד יותר בין הרגלים וסגנון חיים.

<u>טבלה שניה:</u>

Top 10 Features Contributing to PC2

	PC1	PC2	abs_PC1	abs_PC2
avg_glucose_level	0.40	0.90	0.40	0.90
bmi	0.54	-0.31	0.54	0.31
age	0.63	-0.18	0.63	0.18
work_type_children	-0.18	0.11	0.18	0.11
ever_married	0.24	-0.10	0.24	0.10
gender_Female	0.01	-0.09	0.01	0.09
gender_Male	-0.01	0.09	0.01	0.09
work_type_Private	0.08	-0.09	0.08	0.09
smoking_status_Unknown	-0.17	0.09	0.17	0.09
smoking_status_never smoked	0.07	-0.05	0.07	0.05

רכיב ראשון PC1:

גלוקוז: המשתנה המשמעותי ביותר ברכיב PC1.

הBMI, גיל, סטטוס זוגי נשוי / היה נשוי בעבר: תורמים משמעותית.

- המשמעות: PC1 הוא בעיקר ציר בריאותי.

רכיב שני PC2:

גלוקוז: המשתנה המשמעותי ביותר ברכיב PC2.

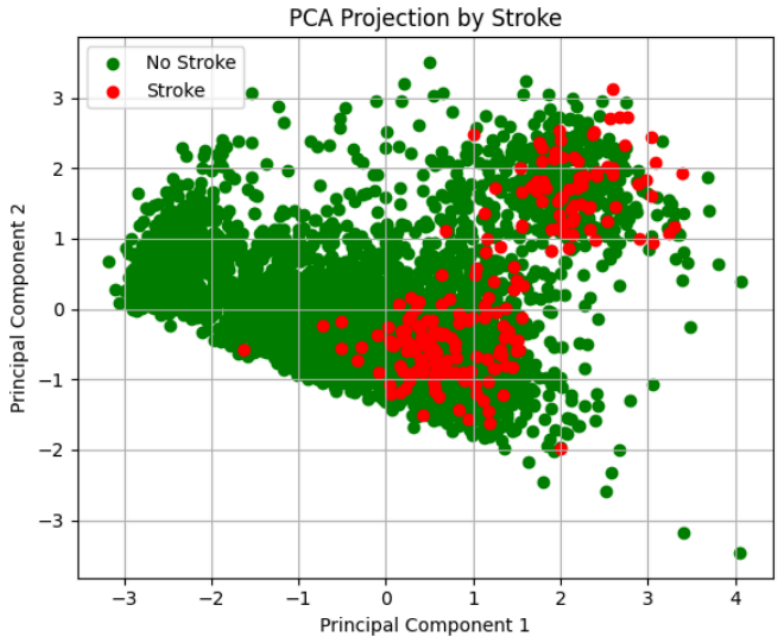
הBMI, גיל, ילדים, סטטוס זוגי נשוי / היה נשוי בעבר: תורמים משמעותית.

- המשמעות: PC2 מפריד יותר בין הרגלים וסגנון חיים.

ניתן לראות שישנם פיצ'רים המופיעים בשני הרכיבים, כל פעם עם מקדם שונה, ההסבר לכך הוא שכל רכיב מסביר שונות אחרת של הפיצ'ר.

בס"ד

הצגת הדאטה לפי PCA:

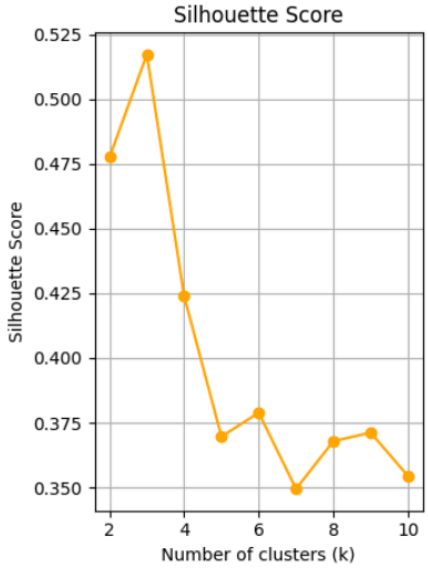


תצוגת גרף זו אינה משקפת מידע קונקרטי על הדאטה, למעט פילוח הדאטה לפי לקה בשבץ / לא לקה בשבץ. לא ניתן להסיק מהם הפיצ'רים המשפיעים על הסיכוי ללקות בשבץ מוחי.

ביצוע אשכול על הדאטה לאחר PCA:

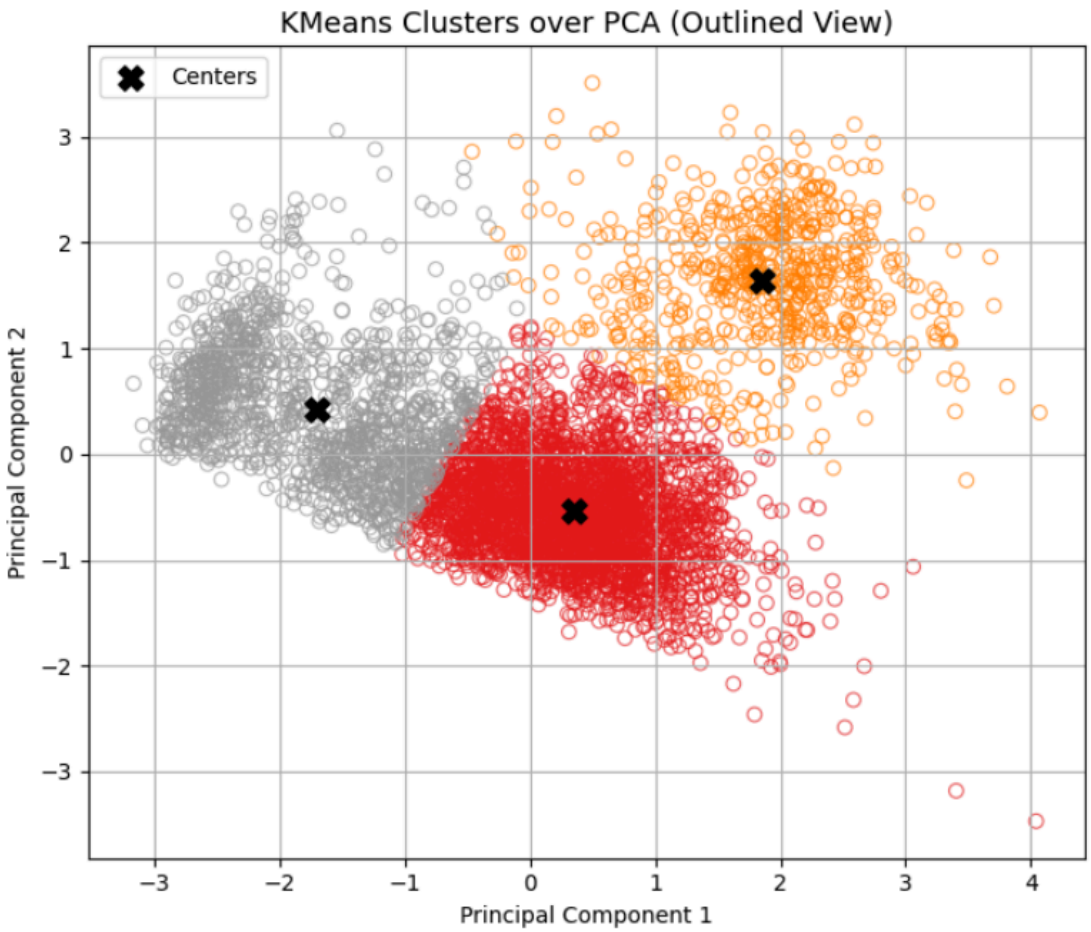
בשביל לדעת מה מספר האשכולות הטוב ביותר השתמשנו במדד Silhouette. לפי מדד זה מספר האשכולות הטוב ביותר הוא 3.

חשוב לציין שהאשכול מתבצע על הדאטה לאחר הורדת המימד - אחרי PCA נשארים רק עם המידע "המועיל", KMeans מנסה לגלות קבוצות פנימיות בתוך המרחב.



בס"ד

פרשנות גרף האשכול לאחר הורדת המימד:



ניתן לראות חלוקה ברורה בין שלושת האשכולות – סימן ש־KMeans הצליח לזהות מבנים פנימיים טובים.

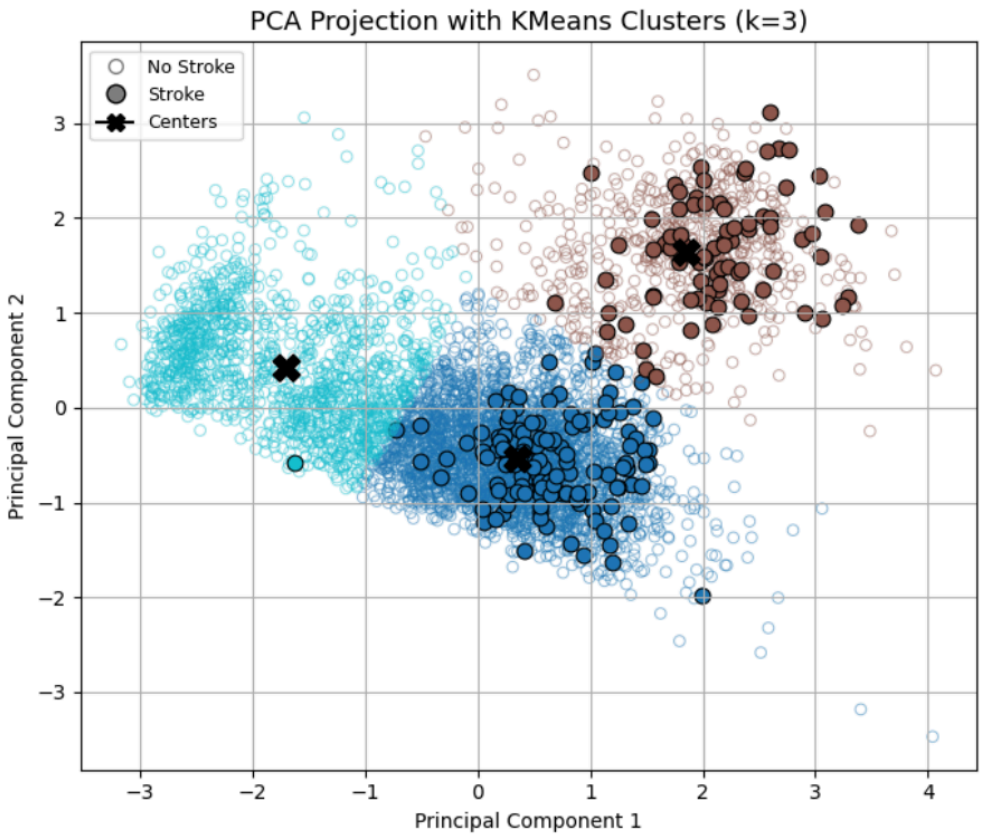
אחרי שהבנו כי רכיב PC1 מייצג מצב רפואי כללי (גיל, סוכר, BMI) ו-PC2 משקף פרופיל דמוגרפי תעסוקתי נוכל לפרש את הגרף באופן הבא:

צבע	מיקום	פרשנות
אפור	PC1 נמוך, PC2 סביב 0	פרופיל כללי, ללא מאפיינים בולטים
אדום	PC1 בינוני, PC2 שלילי	אוכלוסייה צעירה ובריאה יותר יחסית, עם מצב רפואי טוב יותר
כתום	PC1 גבוה, PC2 חיובי	אוכלוסייה במצב בריאותי ירוד

בס"ד

פרשנות הגרף PCA Projection with KMeans Clusters (k=3):

הגרף מציג את הנתונים לאחר הפחתת מימדים (PCA) וחלוקה ל־3 קלאסטרים באמצעות KMeans. כל אשכול מסומן בצבע שונה והנקודות מסמנות מטופלים, כשהן מקודדות לפי מצב שבץ:



עיגול ריק = לא שבץ | עיגול מלא עם מסגרת שחורה = שבץ | X שחור = מרכז האשכול

מה ניתן לראות לפי הצבעים:

- <u>אשכול טורקיז:</u>

  אשכול גדול המאופיין ברובו המוחלט במטופלים ללא שבץ. שיעור חולי השבץ כאן נמוך יחסית, מה שמעיד על אוכלוסייה עם פרופיל בריאותי טוב יותר או מאפיינים דמוגרפים טובים.

- <u>אשכול כחול כהה:</u>

  אשכול מעורב, עם שילוב של בריאים וחולי שבץ. ניתן לראות חפיפה בין הקבוצות, מה שמרמז על פרופיל "ביניים" - אוכלוסייה בעלת חלק מגורמי הסיכון אך לא כולם.

- <u>אשכול חום:</u>

  זהו האשכול עם השכיחות הגבוהה ביותר של חולי שבץ. רואים כאן ריכוז משמעותי של נקודות מלאות (Stroke). ככל הנראה מדובר באוכלוסייה עם פרופיל רפואי פחות טוב - גיל מבוגר יותר, BMI גבוה ורמות גלוקוז גבוהות, כפי שמשתקף מהקשר ל־PC1.

בס"ד

לאחר תצוגת גרף זה החלטנו להציג את הנתונים בצורה מספרית, ברורה ונוחה לפרשנות:

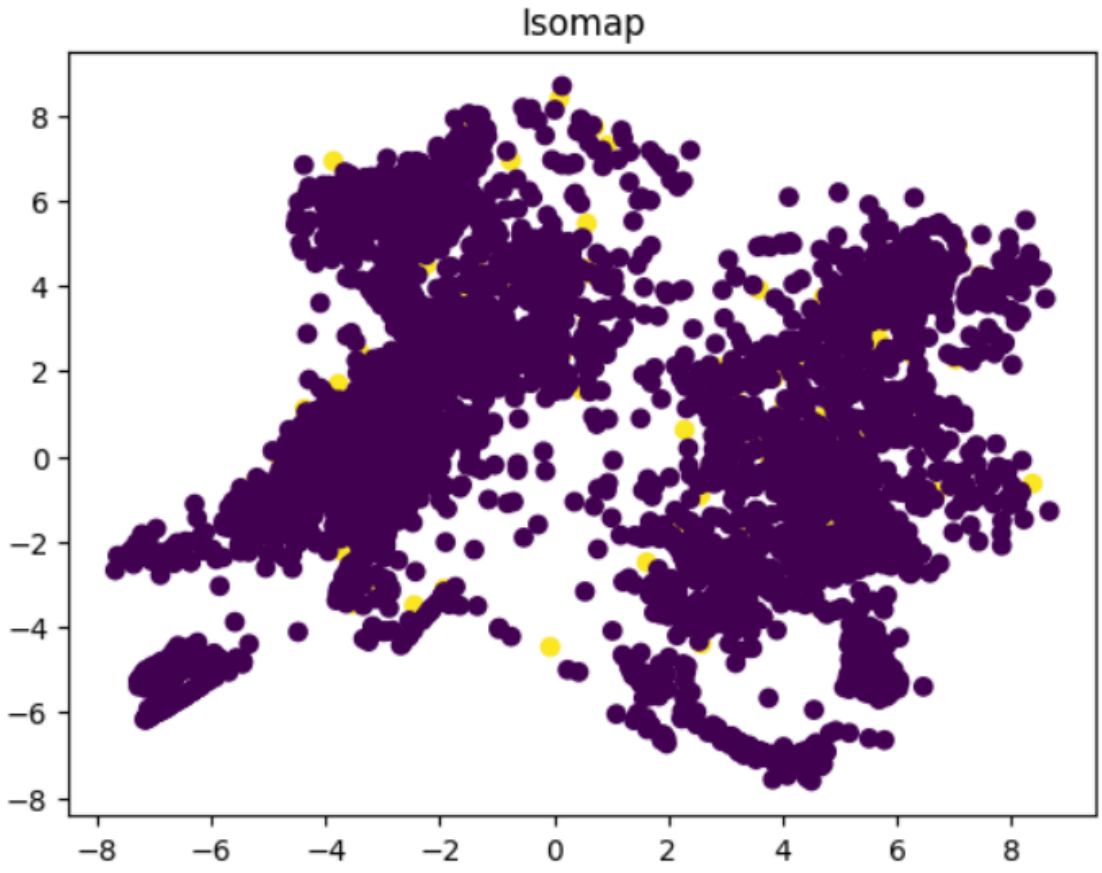
	Cluster	Number of samples	Stroke	% Stroke in Cluster	% Total Strokes
0	Brown	2967	128	4.31	61.24
1	Blue	634	80	12.62	38.28
2	Turquoise	1308	1	0.08	0.48

בס"ד

מחברת שמינית - גרפים בלמידה לא מפוקחת

לפני הפעלת האלגוריתמים על הדאטה עשינו את השלבים הבאים:
טעינת הדאטה הנקי, הינדוס, בחירת פיצ'רים וקידוד הדאטה.

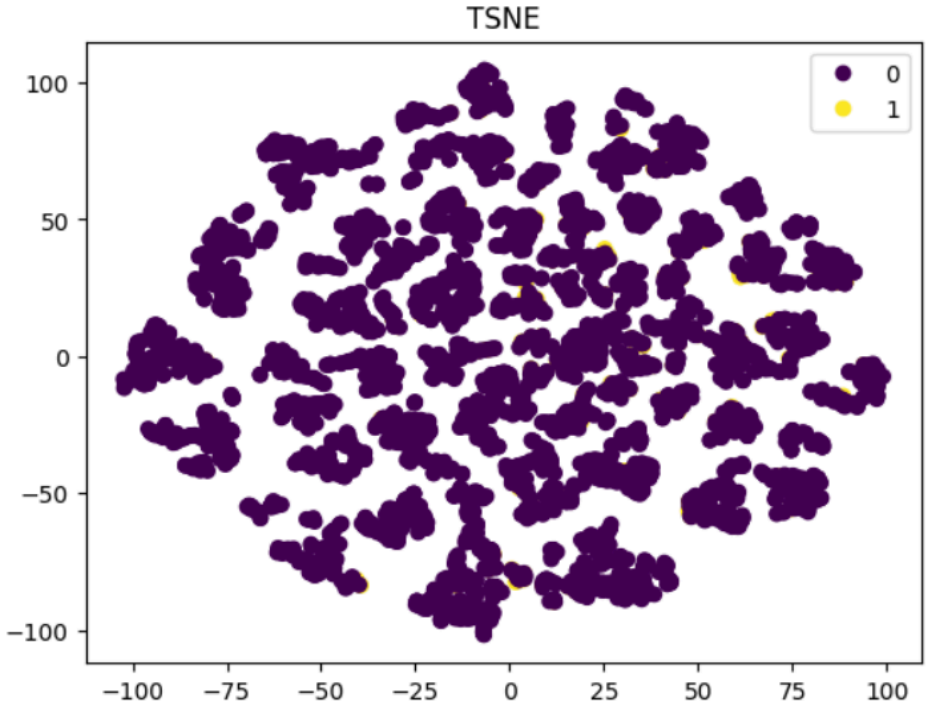
Isomap:



הצירים (X ו־Y) הם שני הרכיבים הראשיים שהתקבלו מ־Isomap.
צבע סגול = לא לוקה שבץ,
צבע צהוב = לוקה בשבץ.
ניתן לראות כי הנקודות אינן מפוזרות בצורה אקראית, אלא מחולקות ל2 אשכולות.
ניתן לראות כי בצידו הימני של הגרף ישנן יותר נקודות צהובות, כלומר יותר אנשים שלקו בשבץ.

בס"ד

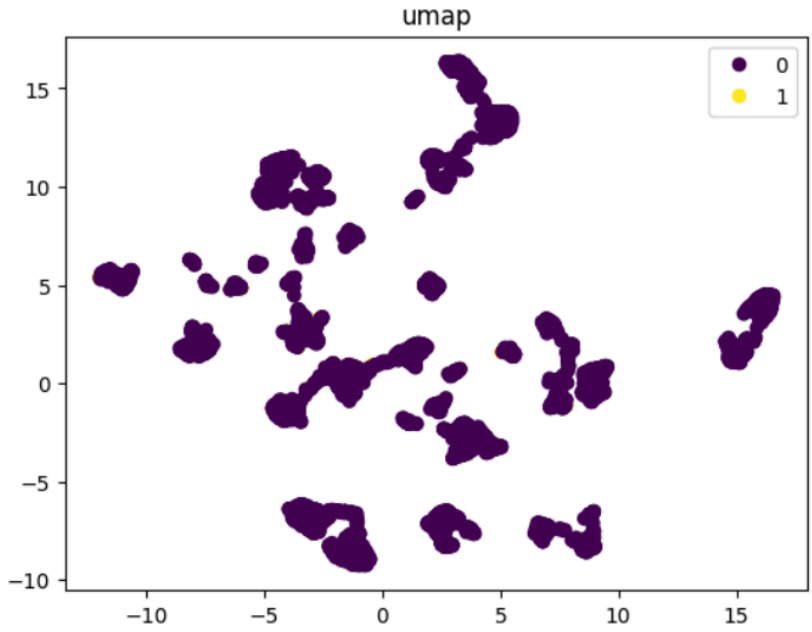
TSNE:



הצירים בגרף הם צירים אותם המודל בנה כדי לשמר את יחסי הקרבה בין הנקודות.
הגרף מראה הרבה אשכולות קטנים בצבע סגול ולעומת זאת קשה מאוד להבחין בנקודות הצהובות, כלומר אין אשכול ייעודי ללוקים בשבץ מוחי, הם נטמעים בתוך שאר האוכלוסיה.

בס"ד

UMAP:



ניתן לראות שהדאטה מתכנס לאשכולות ברורים, אך לוקי השבץ אינם מתכנסים לאשכול ייעודי, אלא משולבים באשכולות האחרים (האנשים שאינם לקו בשבץ) - מה שמעיד על קושי בזיהוי לוקי השבץ.

לסיכום:

לאחר השוואה של שלושת השיטות השונות להצגת הדאטה ניתן לראות ששיטת הumap היא הטובה ביותר משום שהיא מפצלת את הדאטה לאשכולות קטנים וברורים המעידים על קשרים משותפים.