# Deep Learning
## Assignment 4

**אלעד גוברמן 209320696 , ניב כהן 318507241**

**הקדמה:**

בפרויקט זה אנו עוסקים במודלים גנרטיביים מסוג Generative Adversarial Network (GAN) ו-Conditioned Generative Adversarial Network (cGAN) שמטרתם לייצר נתונים טבלאיים סינתטיים תוך שמירה על מאפיינים דומים לנתונים המקורים. מודלים אלה מבוססים על רשתות עצביות ומשמשים ליישומים רבים, כגון יצירת נתונים עבור תרחישי למידה או ניתוחים בהם יש מחסור בנתונים אמיתיים. בדוח זה, נסקור את התהליך המלא של בניית המודלים, אימונם, הערכתם וניתוח הביצועים.

**תיאור הדאטה:**

הדאטהסט Adult מכיל 32,561 דוגמאות ו-15 משתנים, כאשר המשתנה התלוי הוא ההכנסה (income) המחולקת לשתי קטגוריות '<=50K' & '>50K'. המשתנים הבלתי תלוי כללו מאפיינים קטגוריאליים ורציפים, כגון גיל, מגדר, מצב משפחתי, מקצוע ועוד.

**תהליך הניתוח המקדים:**

1. **טעינת הנתונים :טעינת** הנתונים בוצעה באמצעות חבילת scipy, הדאטסט הכיל 32,561 דוגמאות ו-15 משתנים. הנתונים נקראו בהצלחה מפורמט ARFF והומרו ל DataFrame-של pandas לצורך עיבוד נוסף.
2. **ניקוי והשלמת ערכים חסרים :**טיפול בערכים חסרים בוצע על ידי החלפתם בערך מצב (mode) של כל עמודה. צעד זה נדרש כדי לוודא שהנתונים יהיו שלמים ולא יפגעו בתהליך האימון של המודלים.
3. **קידוד מאפיינים קטגוריאליים :**נעשה שימוש ב LabelEncoder-לצורך קידוד המשתנים הקטגוריאליים לערכים מספריים. לדוגמה, עמודת ה workclass-הוצגה בקידוד הבא:
   - 'Federal-gov' → 0
   - 'Local-gov' → 1
   - 'Private' → 3

תהליך זה הושלם עבור כל המשתנים הקטגוריאליים, כפי שמוצג בפלט הניתוח המקדים (ניתן לראות בקוד)

4. **נרמול מאפיינים רציפים :**המאפיינים הרציפים, למשל גיל ושעות עבודה בשבוע, נורמלו באמצעות StandardScaler כדי להביאם לסקאלה אחידה בין 0 ל-1. פעולה זו מסייעת למנוע הטיות שנובעות מהבדלי סקאלה בין משתנים.
5. **פיצול הנתונים :**הנתונים פוצלו לסט אימון וסט בדיקה ביחס של 80/20 תוך שימוש בפרמטר stratify כדי להבטיח שיחס הכיתות (הכנסה גבוהה ונמוכה) יישמר בצורה אחידה בין שני הסטים.

**פלטים מרכזיים מהניתוח המקדים:**

- עמודות קטגוריאליות כוללות: ['workclass', 'education', 'marital-status', 'occupation', 'relationship', 'race', 'sex', 'native-country']
- עמודות רציפות כוללות: ['age', 'fnlwgt', 'education-num', 'capital-gain', 'capital-loss', 'hours-per-week']

**תהליך בחירת הייפרפרמטרים:**

בתהליך האימון של המודלים הגנרטיביים, הגדרת היפרפרמטרים היא שלב קריטי המאפשר כוונון מדויק של המודל בהתאם לדאטה ולמטרת האימון. היפרפרמטרים כמו גודל השכבות הנסתרות, גודל הבאצ', קצב הלמידה, והאופטימייזר משפיעים על יכולת המודל ללמוד ולהתכנס בצורה אפקטיבית.

ראשית, הגדרנו את מספר הנוירונים בשכבות הנסתרות (hidden_dims) לערכים של 128 ו-256. הבחירה בערכים אלו נעשתה מתוך איזון בין יכולת המודל ללמוד תבניות מורכבות לבין מניעת תופעת ה-overfitting, שיכולה להתרחש אם מספר הנוירונים יהיה גדול מדי ביחס לגודל הדאטה. בנוסף, גודל השכבות משפיע גם על משך זמן האימון, ולכן נדרשנו לאיזון מדויק.

random_states נבחרו כך שיאפשרו שחזור התוצאות בניסויים שונים. הם מונעים שונות גבוהה בתוצאות הנובעת מפיצולים אקראיים של הנתונים ומשקפים אמינות גבוהה יותר בניתוח הביצועים.

גודל הבאצ' (batch_sizes) נקבע לערכים של 32 ו-64. בחירה זו מאפשרת למודל לעדכן את המשקלים בתדירות גבוהה יחסית תוך שמירה על יציבות בתהליך האופטימיזציה. באצ'ים קטנים מדי עלולים להוביל לרעש באימון, בעוד באצ'ים גדולים מדי עלולים להאט את התהליך ולדרוש משאבים חישוביים גדולים יותר.

קצב הלמידה (learning_rates) נבחר לערכים של 0.0002 ו-0.0005 מתוך מטרה לאפשר למידה מהירה אך יציבה. קצב למידה גבוה מדי עלול למנוע התכנסות של המודל, בעוד קצב נמוך מדי יכול להאריך את זמן האימון מעבר לנדרש.

בכל הנוגע לאופטימייזרים, בחרנו להשתמש ב Adam-וב-RMSprop, כל אחד עם יתרונות משלו. Adam הוא אופטימייזר מתקדם המשלב התאמות אוטומטיות של קצב הלמידה על בסיס ממוצעים של שיפועים קודמים, מה שמאפשר למידה מהירה ואפקטיבית בתרחישים מגוונים. לעומתו RMSprop , תוכנן להתמודד היטב עם בעיות שבהן השיפועים משתנים בתדירות גבוהה, והוא מצטיין במיוחד במצבים שבהם יש צורך ביציבות באימון תוך מניעת קפיצות גדולות מדי בעדכוני המשקלים. השילוב בין שני האופטימייזרים מאפשר לנו לבחון איזון בין מהירות התכנסות ליציבות בתהליך האימון.

לבסוף, השתמשנו בפונקציית ההפסד BCEWithLogitsLoss, המתאימה לבעיות סיווג בינאריות. פונקציה זו מאפשרת למדוד את הדיוק של ההפרדה בין דוגמאות אמיתיות לסינתטיות במהלך האימון. חשוב לציין שפונקציה זו רגישה לחוסר איזון בין הכיתות, ולכן נדרשנו לטפל בנקודה זו במהלך הכנת הנתונים.

**ארכיטקטורת המודל ותהליך האימון**

**תכנון הארכיטקטורה:**

תכננו את ארכיטקטורת מודל ה- GAN וה- cGAN באופן המאפשר ייצור נתונים סינתטיים איכותיים תוך התחשבות במבנה הנתונים הטבלאיים. הארכיטקטורה מתבססת על שימוש בגנרטור ליצירת נתונים ובמבחין להערכתם, תוך שילוב של שכבות ליניאריות, שכבות אקטיבציה, ומנגנוני Dropout להבטחת יציבות האימון ומניעת overfitting.

הגנרטור מורכב מכמה שכבות ליניאריות המעבדות בהדרגה את הרעש הגאוסי שמוזן כקלט וממירות אותו לדוגמאות סינתטיות. בכל שלב נוסף מנגנון אקטיבציה מסוג LeakyReLU לשיפור היכולת של המודל ללמוד תבניות מורכבות, וכן שכבות נורמליזציה (BatchNorm1d) המייצבות את האימון. במודל ה-cGAN, הקלט כולל גם תוויות מותנות המשולבות בתהליך הייצור, מה שמאפשר שליטה על סוג הדוגמאות המתקבלות.

כדי לשפר את ביצועי המודל ולמנוע בעיות התכנסות, אתחלנו את המשקלים של הגנרטור והמבחין באמצעות אתחול מסוג Xavier Normal, המספק ערכים התחלתיים מאוזנים ומונע שיפועים מתפוצצים או נעלמים בתחילת האימון.

המבחין בנוי מסדרת שכבות ליניאריות המצמצמות בהדרגה את המידע ומובילות לשכבה סופית עם פונקציית Sigmoid. שילבנו שכבות Dropout בנקודות מפתח במבחין כדי להקטין את הסיכוי ל-overfitting ולאפשר למודל ללמוד את ההבדלים בין נתונים אמיתיים לסינתטיים בצורה אפקטיבית.

**תהליך האימון:**

תהליך האימון של המודלים מבוסס על דינמיקה של תחרות בין הגנרטור למבחין:

1. **שלב אימון המבחין:**
   - המבחין מקבל כקלט דוגמאות אמיתיות מתוך הדאטה ודוגמאות סינתטיות שנוצרו על ידי הגנרטור.
   - הדוגמאות האמיתיות עוברות תהליך הוספת רעש קל כדי להקשות על המבחין ולמנוע overfitting.
   - בכל 10 אפוקים, מתבצע אימון של המבחין כדי להקל על אימון הגרנטור.
   - מחשבים את ההפסד של המבחין באמצעות BCEWithLogitsLoss אשר משווה בין הפלט לבין תוויות האמת.
   - המשקלים של המבחין מתעדכנים בהתאם לשיפועי ההפסד באמצעות אחד מהאופטימייזרים .
2. **שלב אימון הגנרטור:**
   - הגנרטור מקבל רעש גאוסי כקלט ובמודל ה-cGAN, גם תוויות מותנו ומייצר דוגמאות סינתטיות.
   - הדוגמאות נשלחות למבחין, שמטרתו להבחין ביניהן לבין דוגמאות אמיתיות.
   - ההפסד של הגנרטור מחושב בהתאם ליכולת המבחין לטעות בזיהוי הדוגמאות הסינתטיות כאמיתיות.

- המשקלים של הגנרטור מתעדכנים במטרה לשפר את איכות הדוגמאות הסינתטיות.

3. **מנגנון עצירה מוקדמת:**

- במהלך האימון עקבנו אחר השינויים בהפסד של הגנרטור. אם לא נצפתה ירידה משמעותית בהפסד לאורך מספר איטרציות רצופות, האימון הופסק על מנת לחסוך בזמן ולמנוע overfitting.

**בחירות עיצוביות נוספות:**

- שילוב שכבות BatchNorm1d בגנרטור לצורך ייצוב התהליך.
- שימוש ב LeakyReLU-כמנגנון אקטיבציה לניהול טוב יותר של שיפועים קטנים.
- הוספת רעש קל לדוגמאות האמיתיות במהלך אימון המבחין כדי למנוע למידה פשוטה מדי ולשפר את היכולת הכללית של המודל.
- הגבלת הגרדינטים (Gradient Clipping) במבחין לצורך שליטה טובה יותר בתהליך האימון.

באמצעות תכנון זה, הצלחנו להשיג איזון בין ייצור נתונים סינתטיים מגוונים לבין שמירה על יציבות ודיוק במהלך האימון.

**תוצאות האימון**

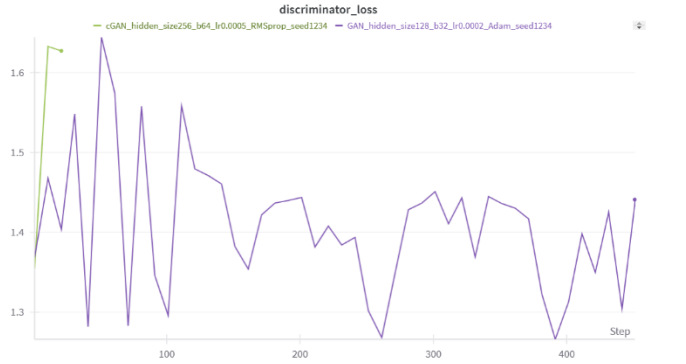
האימון כלל 96 קונפיגורציות שונות של שילובי היפר-פרמטרים (48 עבור כל סוג מודל). נתמקד ב-2 המודלים בעלי הביצועים הטובים ביותר עבור כל ארכיטקטורה(לפי מדד efficiency ratio):

1. Best configuration: **Architecture_cGAN_h256_b64_lr0.0005_RMSprop_seed1234**
   Parameters: {'architecture': 'cGAN', 'hidden_dim': 256, 'batch_size': 64, 'learning_rate': 0.0005, 'optimizer': 'RMSprop', 'criterion': BCEWithLogitsLoss(), 'random state': 1234, 'num epochs': 22, 'real auc': 0.906, 'synthetic auc': 0.751, 'efficacy ratio': 0.82972, 'detection score': 1.0} Best loss: 0.313262

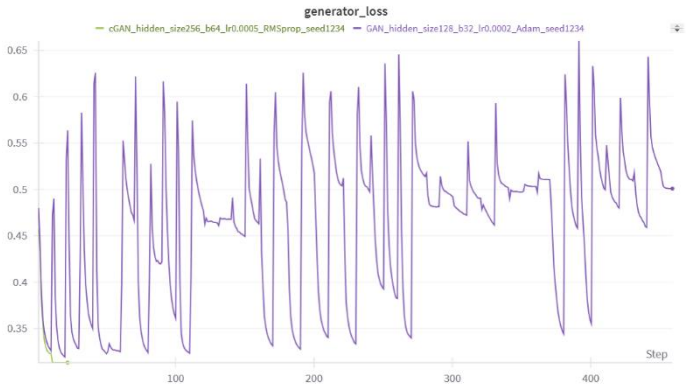
2. Best configuration: **Architecture_GAN_h128_b32_lr0.0002_Adam_seed1234**
   Parameters: {'architecture': 'GAN', 'hidden_dim': 128, 'batch_size': 32, 'learning_rate': 0.0002, 'optimizer': 'Adam', 'criterion': BCEWithLogitsLoss(), 'random state': 1234, 'num epochs': 459, 'real auc': 0.906, 'synthetic auc': 0.689, 'efficacy ratio': 0.76135, 'detection score': 1.0} Best loss: 0.319267

<u>בשלב הראשון</u> ננתח ונשווה בין שני מודלים אלו. לאחר מכן, <u>בשלב השני</u> נשווה בין שלושת ערכי הrandom state עבור כל מודל ונבדוק השפעה שלהם על הביצועים.

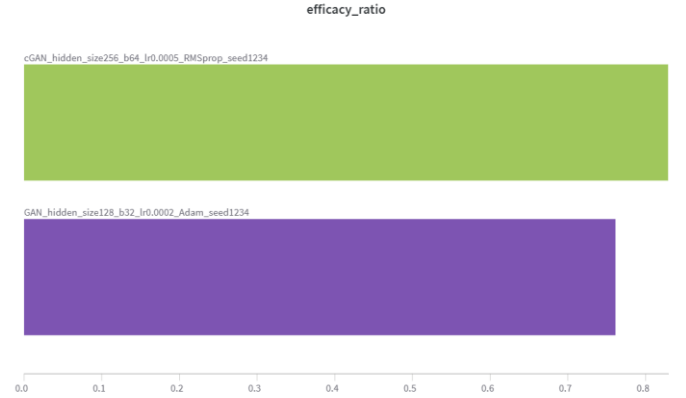
בהתבוננות ראשונית בשני המודלים, ההבדל המשמעותי ביותר הוא במספר האפוקים בין שני המודלים.
מודל ה-cGAN **נעצר אחרי 22 אפוקים**, ככל הנראה בשל התכנסות מהירה לאחר הפעלת מנגנון עצירה מוקדמת ולעומתו מודל ה-GAN **המשיך עד 459 אפוקים,** כך שהאימון שלו היה ארוך בהרבה.




בהסתכלות על גרף loss של ה-Discriminator לאורך האימון בשני המודלים ניתן לראות שהתנודתיות יחסית גבוהה, מה שתואם את ההתנהגות האופיינית ל-GANs שבהם ה-Discriminator וה- Generator מתחרים זה בזה. ב- GAN (הקו הסגול), ניתן לראות ירידות ועליות לאורך כל 459 האפוקים, מה שמעיד על תהליך למידה יותר ארוך וחוסר יציבות.



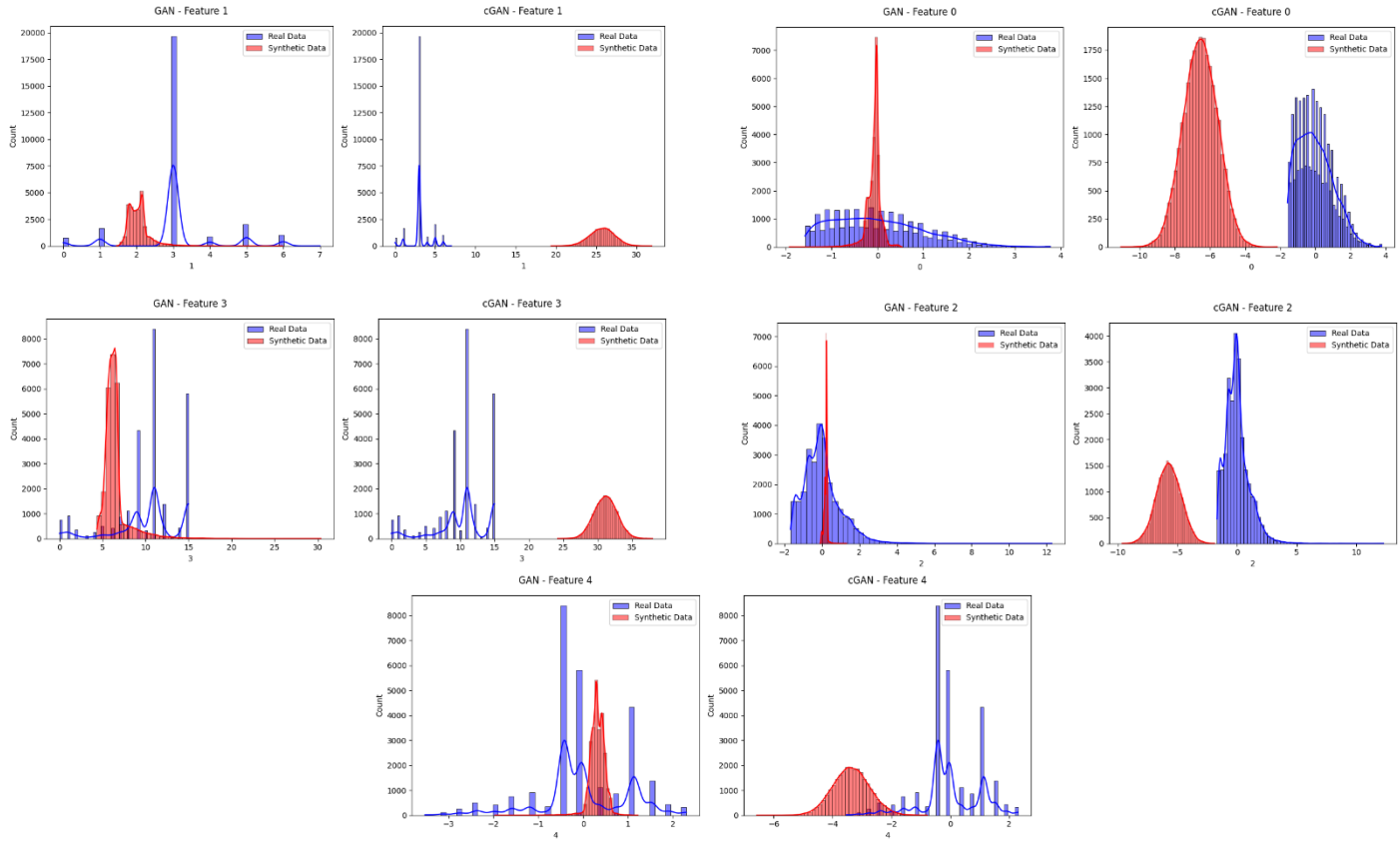

בהוספת גרף loss של ה- Generator ניתן לראות שגם הוא תנודתי מאוד לאורך האפוקים ומחזק את ההתנהגות האופיינית. ב- cGAN רואים ירידה יחסית מהירה והתכנסות סביב הערך 0.313 עד לעצירת האימון. בהשוואה אליו, התהליך של GAN ארוך ומלא בקפיצות חדות. אנו מניחים כי הסיבה לכך היא שה-Discriminator נעשה חזק מדיי לפרקים, מה שגרם ל- Generator להתקשות בלמידה ולהתכנס בצורה יציבה.




מדד ה-Efficacy Ratio בודק עד כמה הנתונים הסינתטיים שנוצרו יכולים להחליף את הנתונים האמיתיים בדאטה. אנו רואים כי מודל cGAN קיבל יחס Efficacy גבוה יותר (0.8297) ממודל GAN (0.76135) משמע הנתונים הסינטטיים של cGAN היו מעט יותר שימושיים לאימון מודל והניבו תוצאות טובות יותר.
שני המודלים קיבלו ערכי Efficacy נמוכים מ-1 , כלומר אין מודל שהצליח לייצר נתונים שמתפקדים טוב לפחות כמו הנתונים האמיתיים והם אינם הצליחו לשמר את המידע הקריטי בצורה מושלמת.

במדד ה-Detection שני המודלים קיבלו AUC=1 כלומר מודל ה-Random Forest הצליח לזהות בצורה מושלמת בין נתונים אמיתיים לסינתטיים. המשמעות היא שהנתונים הסינתטיים שונים באופן ברור מהנתונים האמיתיים, הם אינם מצליחים "להיטמע" בתוך הדאטה ולכן קל לזהות אותם כ"לא אמיתיים". במודל אידיאלי, היינו רוצים AUC קרוב יותר ל-0.5 ,כך שהנתונים הסינתטיים כל כך דומים לאמיתיים, שאי אפשר להבדיל ביניהם בקלות. כמובן שזו תצאה לא אידאלית ולהמשך עבודה עתידי היינו מנסים לשפר את יכולת הג'ינרוט ליצירת נתונים מגוונים יותר ללא התמקדות בתבניות מסוימות ובכך להימנע מבעיית ה-Mode Collapse אותה למדנו בכיתה. כמו כן, בדיקת ערכי היפר-פרמטרים נוספים אולי גם יניבו תוצאות טובות יותר.

בהמשך לכך, בחרנו 5 פיצ'רים מרכזיים מתוך הדאטה. לכל פיצ'ר, ייצרנו היסטוגרמות המשוות בין ההתפלגות של הנתונים האמיתיים (בכחול) לבין הנתונים הסינתטיים (באדום), תוך השוואה בין שני המודלים. מטרת ההשוואה היא להבין ולזהות פערים בין המודלים.




בהשוואה בין המודלים, ניתן לראות כי מודל GAN (צד שמאל בכל גרף) מצליח לשמר במידה מסוימת את מבנה הנתונים האמיתיים, אך עדיין ישנם התפלגויות שאינן זהות לחלוטין. בכמה מהפיצ'רים, הנתונים הסינתטיים שלו מתרכזים סביב טווחים צרים יותר מאשר הנתונים האמיתיים, מה שמעיד על כך שהמודל לא הצליח לשחזר את כלל המגוון שבדאטה האמיתי. מנגד, מודל cGAN (צד ימין בכל גרף) מציג פערים משמעותיים יותר ביחס לנתונים האמיתיים, כאשר בחלק מהפיצ'רים ההתפלגות של הנתונים הסינתטיים נמצאת בטווח אחר לחלוטין, מה שמעיד על חוסר התאמה גבוהה.

לאחר שהשווינו בין שני המודלים הטובים ביותר וניתחנו את איכות הנתונים שהם מייצרים, נבדוק עד כמה המסקנות שלנו יציבות כאשר משנים את ערך ה-Random Seed. ייתכן שמודל מסוים מציג ביצועים טובים עם Seed מסוים, אך משתנה משמעותית כאשר מריצים אותו מחדש. לכן, נשווה את ביצועי שני המודלים תחת שלושה Seeds שונים ונבחן האם קיימת שונות משמעותית או שהתוצאות נותרות עקביות.

**עבור cGAN:**

| Efficacy Ratio | AUC סינתטי | AUC אמיתי | Seed |
|---|---|---|---|
| 0.83 | 0.3496 | 0.89845 | 1234 |
| 0.4798 | 0.43698 | 0.91076 | 42 |
| 0.39508 | 0.75172 | 0.906 | 57 |
| **0.5683** | **0.5121** | **0.90507** | ממוצע |

**עבור Gan:**

| Efficacy Ratio | AUC סינתטי | AUC אמיתי | Seed |
|---|---|---|---|
| 0.76135 | 0.4074 | 0.89845 | 1234 |
| 0.51473 | 0.4688 | 0.91076 | 42 |
| 0.49056 | 0.68978 | 0.906 | 57 |
| **0.58888** | **0.52266** | **0.90507** | ממוצע |

**ניתוח AUC אמיתי –** שני המודלים קיבלו ערכים גבוהים ויציבים יחסית. הממוצע עבור שניהם הוא **0.90507** , ללא סטייה משמעותית בין ה-Seeds.

**ניתוח AUC סינתטי** – המודל cGAN מציג שונות גבוהה יותר בין ה-Seeds. הערכים נעים בין 0.3496 ל-0.75172, פער משמעותי של יותר מ-40% בין הריצות מה שמצביע על חוסר יציבות המודל. לעומתו, המודל GAN מציג יציבות גבוהה יותר עם AUC שנע בטווח 0.4074–0.68978 כאשר השונות בין ה-Seeds קטנה יותר מאשר ב-cGAN.

**ניתוח יחס הEfficacy-** מודל GAN מציג ממוצע Efficacy Ratio גבוה יותר (0.58888) מאשר ה cGAN-(0.5683). עם זאת, ל- cGAN יש שונות קיצונית בין ה-Seeds. המודל GAN גם מציג שונות, אך באופן פחות קיצוני. אפשר להבין מכך שהשונות הגבוהה בערכי ה-Efficacy של ה-cGAN מצביעה על כך שהמודל רגיש יותר ל-Seed המשמש לאימון, כלומר הוא אינו לומד בצורה יציבה. ה-GAN לעומת זאת, מציג תוצאות יציבות יותר, אך עדיין מדובר בערכים נמוכים יחסית, מה שמעיד על כך שהנתונים הסינתטיים אינם מהווים תחליף מושלם לאמיתיים.

**סיכום**

בהשוואה בין שני המודלים, מצאנו כי לכל אחד מהם יתרונות וחסרונות. ה-GAN הצליח לשמר חלק מהתכונות הסטטיסטיות של הנתונים האמיתיים בצורה טובה, אך הראה שונות רחבה בהתפלגויות של הנתונים הסינתטיים. לעומתו, ה-cGAN, הציג חוסר יציבות משמעותי בין ריצות שונות ויצר נתונים שהתפלגותם שונה מהותית מהנתונים האמיתיים. מניתוח ה-Seeds נראה כי ביצועי ה-cGAN מושפעים מבחירתו באופן משמעותי, מה שמעיד על רגישות גבוהה לתנאי האימון. במדד ה-Detection, שני המודלים קיבלו AUC של 1, מה שמצביע על כך שהנתונים הסינתטיים מובדלים מאוד מהנתונים האמיתיים. המודלים לא הצליחו לשמר את המבנה האמיתי של הדאטה. בנוסף, יחס ה-Efficacy היה נמוך עבור שני המודלים, מה שאומר שהנתונים הסינתטיים שלהם אינם מהווים תחליף - השימוש בהם עלול להוביל לירידה בביצועי המודלים. מאחר והמודלים סובלים מחוסר יציבות, שפורים אפשריים לעבוד עתידית יהיו בעזרת גישות מתקדמות יותר לייצוב האימון, כגון שימוש בשיטות למניעת Mode Collapse או שילוב מנגנוני רגולריזציה מתקדמים והתאמות בפרמטרי האופטימיזציה.