

# ממ"נ 21 – תומר ריפין – 322230608

## שאלה 1 - הגדרת הבעיה והכנת הנתונים

### מטרת כריית המידע

מטרת כריית המידע בפרויקט הנוכחי היא לחזות לאיזה אדם יש סיכוי גבוה לחייב בשbez באמצעות פרמטרים כגון: מין, גיל, מחלות עבר,عيישן ופרמטרים נוספים.

הנחה: הנתונים נאספו באופן מהימן על ידי החוקרים.

### הגדרת הנתונים

נעוז במידע אשר ניתן ב-kaggle על כל אחת מהעמודות ובניתוח מידע בסיסי על הנתונים עם הספרה Pandas בפייתון. בנוסף, נעוז בחיפושי אינטרנט ובמאמרם המצורפים כדי ללמוד על ייחדות המידע של חלק מהעמודות.

ונכל לראות בטבלה למטה סיכום על כל עמודה שהשתמשנו בה בניתוח זה, העובדה החשובה ביותר שניתן לראות מניתוח זה היא שההתפלגות בעמודת המטרה stroke שלנו אינה אחידה. אחד האנשים שקיבלו שbez קטן משמעותית מ אחוז האנשים שלא (~5% בערך).

סה"ב רשומות לפני הנקוי - 5110

שם עמודה	תיאור התוכנה	יחידות מידע	סוג הנתונים	תחומי ערכאים	נתונים חסרים
מין	Gender	Male Female Other	קטגורית		אין
גיל	Age	שנים	Nominal	[0.08-82]	אין
לחץ דם	Hypertension	האם יש להם יותר	Binari	0,1	אין
heart_disease	heart_disease	האם חוו מחלת לב	Binari	0,1	אין
ever_married	ever_married	האם היה נשוי בעבר	Binari	Yes, No	אין
work_type	work_type	סוג העבודה בה	Private Self-employed Govt_job Children Never_worked	קטגורית	אין
Residence_type	Residence_type	סוג שכונת המגורים שבה התגורר האדם	Urban Rural	קטגורית	אין

אין	[55.12, 271.74]	נומרי	$\frac{mg}{dL}$	רמת הגלוקוז הממוצעת	avg_glucose_level
יש – חסרים ערבים אצל 201 אנשים	[10.3, 97.6]	נומרי	$BMI = \frac{weight}{height^2} = \frac{kg}{m^2}$	מדד שטרתו لتאר את האם משקל אדם תקין	Bmi
יש – עם סטטוס 夷居ן לא ידען		קטגוריו	formerly smoked, never smoked, smokes, Unknown	האם האם עישן או מעשן בעברו	smoking_status
אין	0, 1	בינארי		האם קיבל שbez	Stroke

## שלבי ה-KDD

**מטרת ברית המדע:** מטרת ברית המדע בפרויקט הנוכחי היא לחזות לאיזה אדם יש סיכוי לחלהות בשbez באמצעות פרמטרים כגון: מין, גיל, מחלות עבר, עישון ופרמטרים נוספים.

### איסוף ושמירת הנתונים:

הנתונים הנדרשים לתחילה בנית מידע התקבלו מ- [Kaggle](#) במלואם ולכן אין צורך באינטגרציה. הנתונים עצם מקורם במחקר של חברת [McKinsey & Company](#). במחקר המקורי של החברה היו במאגר 29,072 רשומות אבל רק בערך 30% התפרסם באתר ונגיש לנו. הנתונים שכנו נגושים לנו הם נתונים על 5110 אינדיבידואלים שגרים בנגלדש. בנוסף, לפני הפרסום לאתר החוקרים ערכו מעט את הנתונים המקוריים מהמקור אצל [McKinsey & Company](#).

### ניקוי הנתונים ועיבוד מוקדם:

בשלב זה נחפש נתונים רועשים: נודא שאין נתונים חסרים או ריקים. נוכל לטפל ברשומות בלבד על ידי השמה של הממוצע או של הערך הנפוץ ביותר במקומם. בנוסף, נודא שאין חריגות קייזניות בנתונים או נתונים לא הגיוניים, מכיוון שיש לנו כמות גדולה יחסית של נתונים, נוכל למחוק רשומות עם נתונים חריגים בצורה קייזנית אך נדרש להיות זהירים ולא למחוק יותר מדי רשומות של אנשים שחלו בשbez כי יש יחסית מעט בלבד. לכן, נוכל להעזר בשיטה נוספת – *deleting* שטרתה לבצע דיסקרטיזציה של ערכים רציפים יחסית לקבוצות וכך "להחלק" רשומות חריגות מדי. כדי לזהות האם יש לנו רשומות עם נתונים חריגים מדי, נעזר במידע נוסף מהאינטרנט. עם רשומות כפולות נתמודד בכך שנזהה אותן ונמחק אותן.

### טרנספורמציה הנתונים ורടוקציה:

בשלב זה, נבחן האם אנחנו צריכים לבצע טרנספורמציה למדע שלנו. נדרש לעיתים לבצע פעולה

כזו כדי להתאים את הנתונים לכריה. פעולות טרנספורמציה יכולות להיות נורמליזציה או דיסקרטיזציה.

#### **בחירה שיטות וכליים לביצוע ברית מידע:**

אשתמש ב-`python` בספריות `Pandas`, `matplotlib` לייצוג הנתונים בצורה גרפית ולסידור המידע. בנוסף, אעזר בספריה `sklearn` אשר מימושה את האלגוריתמים הפלונציאלים (עצים החלטה, יערות אקראיים וכו') לשיפור לטובות תהליכי ברית המידע.

#### **ברית המידע:**

ישום האלגוריתמים של ברית הנתונים שנבחרו בשלבים קודמים על הנתונים כדי לחוץ דפוסים או מגמות.

#### **סקירת התוצאות וניתוח:**

בשלב זה נבחן את התוצאות של המודל שיצרנו. נבדוק דיווק, פשוטות, רלוונטיות וכו'. במקרה שתוצאות המודל לא יספקו, נחזור על השלבים שרשמנו לעלה שוב, על מנת לשפר את המודל ולהגיע לתוצאה מספקת.

#### **הסקת מסקנות:**

ניצג את המודל שהתקבל מהתהליך, ונוכל לבחון כל רשותה חדשה שנתקבל ונוכל לחזות את הסיכוי של המטופל החדש לחטופ שבע.

#### **סקירת חלופות לברית מידע**

במה שרשעיפים א' וב', נבעצ סקירה של 4 חלופות שונות לברית מידע. נתיחס בכל אחת מהן ליתרונות וחסרונות שלה בהקשר הבעיה הנתונה.

##### **1. Random Forest**

Random Forest משלב עצים החלטה מרובים כדי לשפר את הדיווק ולהפחית התאמת יתרה. במהלך תהליכי ייצור המודל, מאמנים עצים החלטה רבים על תת קבוצה אקראיית של הנתונים והתקנות. כאשר כדי לקבל את ההחלטה בסוף מחפשים את התוצאה שהתקבלה על ידי רוב העצים האקראיים.

##### **יתרונות:**

a. **דיווק:** משפר את הדיווק על פני עץ החלטה יחיד

b. **רובוטיות:** המודל פחות סביר שיבצע Overfit לעומת עץ החלטה יחיד מפאת שהוא مستمر על ממוצע של עצים החלטה רבים

##### **חסרונות:**

c. **יקר מבחינה חישובית:** יוצר כל העצים וחישוב התוצאה על כל אחד מהם מבביד מאוד על החישוב

d. **לא טוב עם נתונים לא מאוזנים:** יכול להיות מושה כלפי קבוצת הרוב נתונים לא מאוזנים.

2. **עץ החלטה 3D) מבוסס Information Gain –** ID3 הינו אלגוריתם רקורסיבי אשר מייצר עצי החלטה ויציר פיצולים בעץ על בסיס קритריון הנבחנה. הקритריון הנבחר אצלנו הוא Information Gain (רוח אינפורמיטיבי) – תחילת מחושבת האנתרופופיה של עמודות המידע המבוקשת לSieog/וחיזוי, ובכל רמה בעץ נבחנת התכונה בעלת האנתרופופיה המותניתה הנמוכבה ביותר, כלומר זו שתביא לרוח אינפורמיטיבי המירבי לSieog/וחיזוי משתנה המטרה בהתאם לידע הערך של תכונת הפיצול. אלגוריתם זה רץ בצורה רקורסיבית עד לתנאי העציה שבו לכל הרשומות בעלה יש Sieog זהה או שאין יותר תכונות לפצל על פיהן. לאלגוריתם זה קיים האילוץ שהוא עובד רק עם ערכים קטגוריאליים, כלומר דרוש דיסקרטיזציה של הנתונים.

**יתרונות:**

- a. **פשטות:** אלגוריתם פשוט ליישום ופרשנות.
- b.  **מהירות:** חישובים מהירים.
- c. **לא דרוש הנחת לנאריות:** בניגוד לרגרסיה לנארית, באלגוריתם זה, אין צורך בהנחה של קשר לנארו בין המשתנים.

**חסרונות:**

- a. **נטיה ל-*overfitting*:** קל להגיע באמצעות אלגוריתם זה למצב של overfitting ממעט רשומות ונתחנים או עמודה עם אופציות רבות שלא ניתנת לדיסקרטיזציה אשר מקבלות تعدוף על ידי אלגוריתם זה.
- b. **לא מתמודד היטב עם נתונים חסרים**

3. **עץ החלטה C4.5 עם קритריון פיצול gain-ratio –** עץ החלטה זה מנסה להתגבר על הבעיה באלגוריתם ID3 עם gain information – הנטיה לבחירה של תכונות בעלות ערכים רבים על ידי בחירת קритריון אחר לפיצול. ס�מן Gain-ratio הוא קритריון שמנסה "להעניש" תכונות מרובות ערכים על ידי חלוקה של gain information שלhn ב-information של הפיצול.

**יתרונות:**

- a. **טוב לתכונות מרובות ערכים:** עשוי להביא תוצאות טובות יותר מאלגוריתמים אחרים למאגרי מידע עם תכונות מרובות ערכים שונים.
- b. **גמישות:** מתאים לעבודה עם תכונות קטגוריאליות ורציפות כאחד. באמצעות גזימה העץ יודע להפחית התאמת יתר ויודע להתמודד עם ערכים חסרים.

**חסרונות:**

- a. **ענפים לא מאוזנים:** המدد המשמש לפיצול מוגה לבחירת תכונות שambilאות לחולקה לא מאוזנת של ענפים. בעץ הסופי לעיתים רבות יש ענפים שימושית קצרים מכל השאר.

4. **עץ החלטה CART עם קритריון פיצול gini-index** – תוצאת האלגוריתם הינה עץ החלטה ביןארי בלבד. לאחר סיום הריצה מבצעים גיזום על העץ לטובת דיקן שלו.

Gini-index הינו קритריון הפיצול בשיטה זו, gini-index מוכיח מנח שבל התבוננות רציפות וכי יש כמה ובמה נקודות פיצול לכל תבונה. האלגוריתם בוחר את הפיצול שיביא לרמת איזון הנמוכה ביותר. אלגוריתם זה מעדיף פיצול לשתי קבוצות שהיו יחסית שוות בגודלן.

**יתרונות:**

- a. מתאים במיוחד לתוצאות בוליאניות
- b. גיזום: לאחר בניית העץ המלא, האלגוריתם יוצר קבוצה של תת-עצים גדולים ובודח בעץ עם פונקציית העלות-סיבוכיות המינימלית על מנת לצמצם overfit. וכן שיטה זו לרוב נתנות עצי החלטה קומפקטיים ומדוקרים יותר.

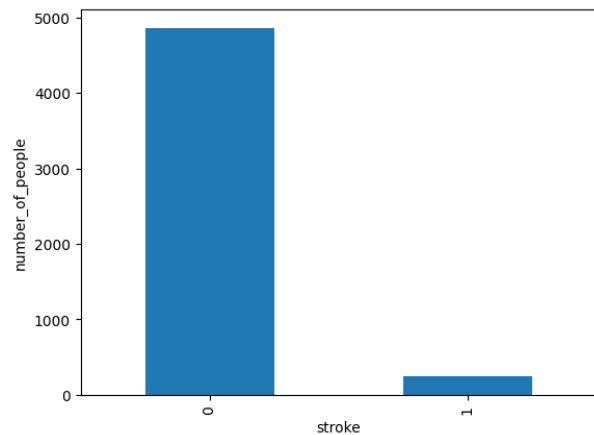
**חסרונות:**

- a. **יעילות:** זמן ריצת האלגוריתם גדול משמעותית משאר האלגוריתמים המוצעים.

## הכנות הנתונים וט依ובם

### תצוגה גרפית, סידור וניקוי הנתונים:

נשים לב שיש לנו מעט מאוד אנשים שקיבלו שbez – עובדה זו תנחה אותי בהמשך ניקוי הנתונים בכר שאנסה להמנע במה שאפשר ממחיקת שורות של אנשים שקיבלו שbez כדי להמנע מאיבוד מידע יקר. להלן היסטוגרמָה של העמודה שbez במאגר המידע:



כדי לקבל תחושה טובה יותר של הנתונים, ניצר היסטוגרמות של העמודות השונות ביחד עם עמודת המטרה:



נוכל לשים לב לכמה תופעות מעניינות שניכו אותן בהמשך –

1. אפשר לראות שיש קורלציה חיובית חזקה בין הגיל לבין שבע.
2. התפלגות התכונה avg\_glucose\_level (מעתה AGL) נראית לא נורמלית

נמשיך בסידור של הנתונים בצורה שתהיה נוחה יותר לעבוד איתם בהמשך:

- . את העמודה married ever, נחליף לבינארית בר ש-Yes יהיה 0 ו-No יהיה 1.
- . את העמודה gender נחליף גם לבינארית בר ש-Male Female יהיה 0 ו-Female יהיה 1. את השורה היחידה שיש בה Other נמחק כי היא יחידה ואוטו ייחיד לא קיבל שbez ולכן אנחנו לא מבדים נתונים קרייטיים.
- . את העמודה Residence\_type נחליף גם לבינארית בר ש-Rural Urban יהיה 0 ו-Urban יהיה 1.
- . נמחק את העמודה p? כי הנתונים בה אינם נתונים מידע על הסיכון של האדם לקבל שbez.

#### **טיפול בערכים חסרים -**

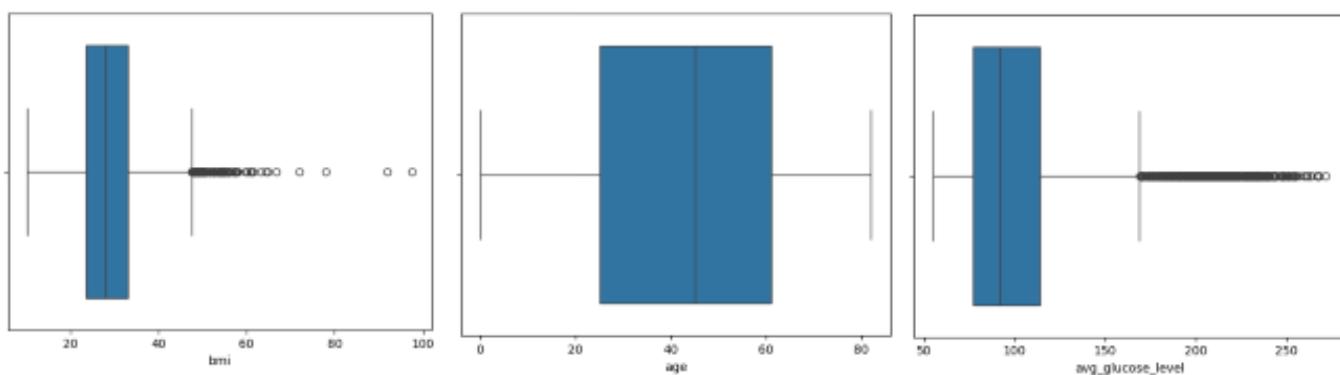
נטפל בערכים החסרים – כמעט בכל התכונות אין לנו ערכים חסרים, יש לנו בעמודת ה-BMI ובעמודת הסטטוס עישון (Unknown).

- . בעמודת ה-BMI אמלא את כל הערכים החסרים בערבי חייזן ה-BMI.
- . בעמודת smoking-status נחליף את כל הערכים שהם Unknown וגם <age> או <work\_type> == children להיות never smokes. אני יצא מהנחה שרוב מוחלט של הילדים הקטנים בימינו לא עושים או עושים בכמות זניחה. לאחר פעולה זו צמצמנו את במות ה-Unknown מ-1544 ל-109!

#### **טיפול בערכים קיצוניים -**

נחפש ערכי קיצון שיכולים להיות לא הגיוניים. נשתמש ב-BoxPlot כדי לקבל תחושה ויזואלית של האם בעמודות הנומריות יש ערכים חריגים.

ובכל לראות שבעמודת הגיל, אין לנו ערכים שנראים חריגים מדי אבל בעמודות ה-BMI וה-AGL יש לנו זנב ימני יחסית ארוך של ערכים שחוודים כחריגים (שמוחושבים על ידי ערכים שהם מעל  $Q_3 + 1.5 \cdot IQR$ ). מעך הרבעון השלישי כאשר  $Q_3 = Q_1 + IQR$ ). לפי הוקפדה ומאמרים נוספים, אנשים עם רמת סוכר ממוצעת גבוהה הם ברמת סיכון גבוהה לסכנת יש גם סכנות בריאותיות ב-AGL גובה אר נתוניים אלו אינם לא הגיוניים לחילופין. לכן, אני מעדיף כמה שפחות למחוק שורות שיכולות לאבד מידע חשוב וamaha רק ערכים קיצוניים מאוד ( $80 > BMI > 225$  ו- $AGL > 200$ ).



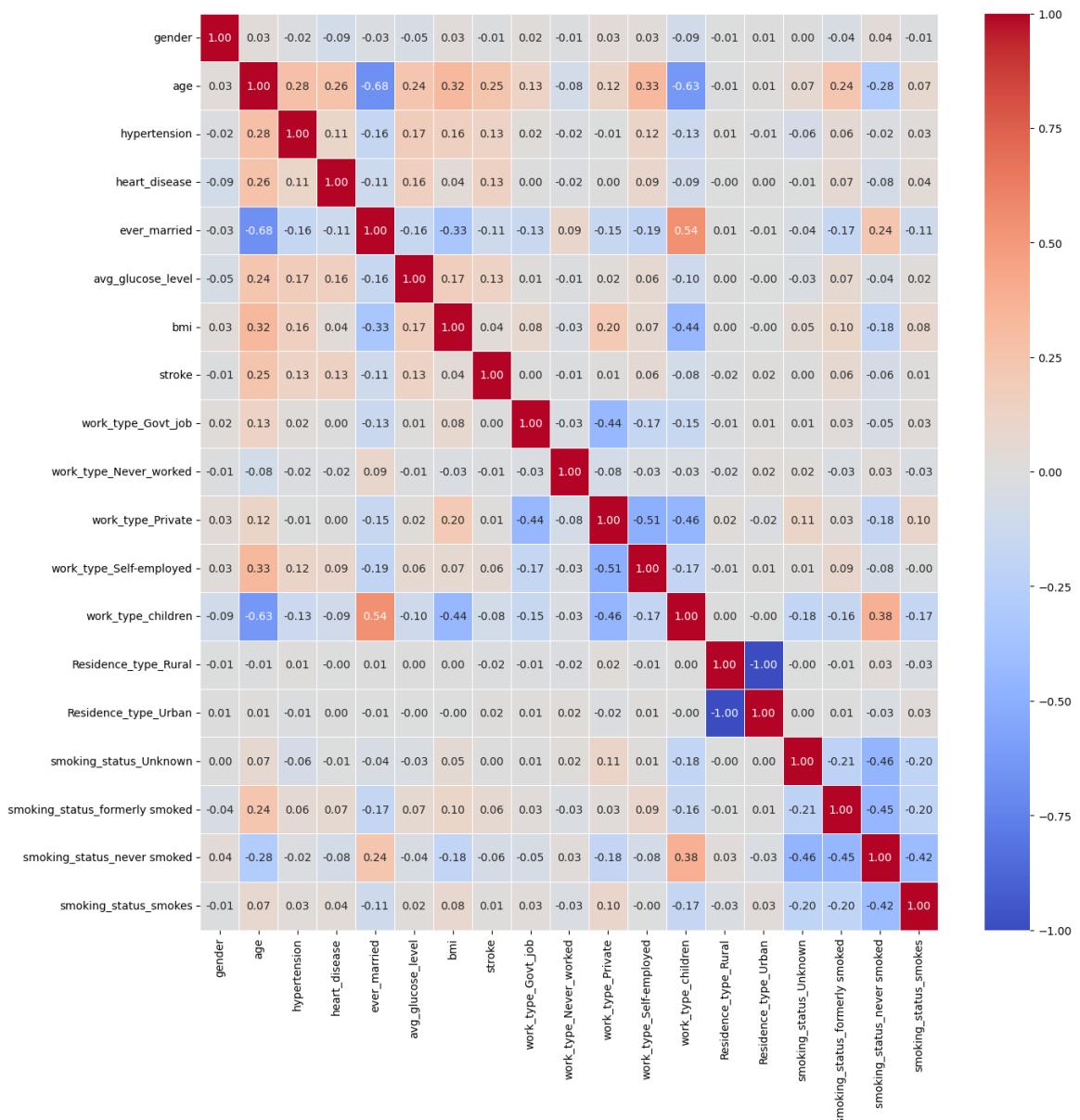
לאחר מחיקת הערכים הקיצוניים, נשארתי עם 4892 שורות במאגר.

## - pd.get\_dummies()

מכיוון שההימוש הפיזוני של RandomForest ו-DecisionTree לא יודע להתמודד עם ערכים מילוליים. השתמשתי בפונקציה של חבילת הפיזון Pandas אשר לוקחת עמודות קטגוריאליות שמכילות יותר משתי אופציות וממירה אותן לעמודות בינהו של האם העמודה בקטgorיה המסויימת או לא.

## מציאת קורלציה בין תכונות -

כדי להבין טוב יותר את הנתונים, ואת היחסים בין העמודות, יצרתי heatmap אשר מראה את הקורלציה בין כל אחת מהעמודות:



ניתוח הגרף:

1. יש קורלציה שלילית בין age ו-ever\_married וגם עם ever\_type\_children. זה כמפורט  
הגיוני בהתאם לתנאים שקשורים לגיל הנבדק (רוב הנבדקים שלא התחתנו היו צעירים).
2. עבר עמודת היעד – stroke ניתן לראות במה דברים:
  - a. כמו שראינו בהיסטוגרמות הקשר חזק ל-stroke זה כמובן הגיל.
  - b. ישנן עוד כמה עמודות עם קשר לא-זניח (מעל 0.1):
    - i. Hypertension
    - ii. Heart\_disease
    - iii. AGL

אני בכוונה מתעלם מ-ever\_married כי כמו שראינו יש קורלציה חזקה בין age לעמודה זו בצורה שכנהאה משפיעה על הקשר שלה ל-stroke.

>Show זה יכולה לעזור לנו להבין יותר את הקשרים בין העמודות השונות ולנסות לחזות על אילו תכונות יتبסס המודל שנבנה. נוכל להסיק שכנהאה בעז החלטה, השורש כמעט תמיד יהיה age כי תכונה עם קורלציה חזקה מאוד. אפשר גם לחשב האם אנחנו רוצים להוריד מהמודל תכונות עם קורלציה חזקה ל-age כמו ever\_married כדי למנוע מהמודל שיבחר להתבסס ארוך רק על הגיל של הנבדקים וכך להתעלם במידע נוסף. אציוں לאחר מכן יש למחוק את העמודות האלה לפני הלמידה של המודל, לא קיבלי שינויים משמעותיים בתוצאות וכן החלטתי להשאיר אותם.

### דיסקרטיזציה לתכונות נומריות -

אני עומד להשתמש בשיטות סיווג מבוססות עצי החלטה שיעזרו לעבוד טוב עם נתונים קטגוריאליים וכן אבצע דיסקרטיזציה לנדרנים הנומריים, הדיסקרטיזציה אף תוכל לעזור להקטין אף יותר את השפעה של הערכים הקיצוניים שעוזרו נותרו. ניסיתי 3 שיטות שונות לדיסקרטיזציה –

1. דיסקרטיזציה מבוססת מידע פומבי – לקחתי שיטות חלוקה נפוצות על פי מחקרים עבר ובר  
מצרו הקטגוריות הבאות:

```

def categorize_AGL(x):
    if 0 <= x < 72:
        return "Low"
    if 72 <= x < 108:
        return "Normal"
    if 108 <= x < 126:
        return "Prediabetes"
    if 126 <= x:
        return "Diabetes"

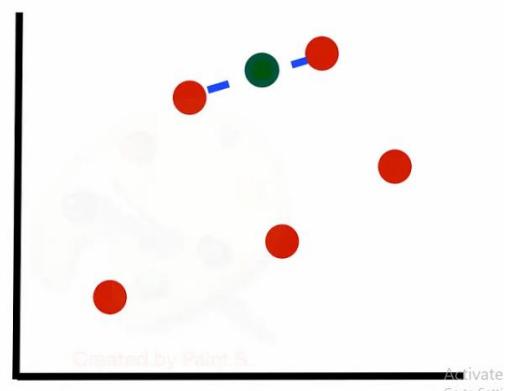
def categorize_age(x):
    if 0 <= x < 10:
        return "Children"
    if 10 <= x < 18:
        return "Teens"
    if 18 <= x < 60:
        return "Adults"
    if 60 <= x:
        return "Elderly"

def categorize_BMI(x):
    if x < 16.5:
        return "Underweight (Severe)"
    if 16.5 <= x < 16.9:
        return "Underweight (moderate)"
    if 16.9 <= x < 18.4:
        return "Underweight (mild)"
    if 18.4 <= x < 25:
        return "Normal"
    if 25 <= x < 30:
        return "Overweight"
    if 30 <= x < 35:
        return "Obese (1)"
    if 35 <= x < 40:
        return "Obese (2)"
    if 40 <= x:
        return "Obese (3)"
```

2. גישת רוחב שווה – חילקתי את התכונות השונות לפי סלים ברוחב שווה כאשר יוצרתי 5 סלים שונים, זה נתן לי את התוצאות הטובות ביותר.
  3. גישת עומק שווה – גם פה, חלוקה ל-5 תאים נתנה לי את התוצאות הטובות ביותר.
- לבסוף, שיטת עומק שווה (Equal Frequency) נתנה לי את התוצאות הטובות ביותר ולבן הלבתי אותה.

### **Synthetic Minority Over-sampling لטובת איזון עמודת המטרה על ידי Oversampling - TEchnique (SMOTE)**

כבר בתחילת העבודה על הנתונים, ניתן היה לראות שההתפלגות של עמודת המטרה אינה מאוזנת. אנחנו רוצים שהמודל שלנו ידע לחזות חול שbez וכרגע יש לו דוגמה קטנה מאוד ללמידה ממנה. נרצה ליצר עוד שורות באופן שלא יוצר over-fitting ולבן השתמש באלגוריתם SMOTE. אלגוריתם זה מייצר באופן סינטטי שורות נוספות – שורה סינטטית מיוצרת על ידי בחירה רנדומלית של שורה מה-minority class ומיציאת השכן הקרוב ביותר שלה, לאחר מכן מגדילים מספר בין 0 ל-1 ובאמצעותו מייצרים דוגמה סינטטית שהיא יושבת על הווקטור בין השורה שמצאנו לשורה הקרובה ביותר שלה.



בתמונה זו<sup>1</sup> – לקחנו את הדוגמה השמאלית העליונה ביותר, חיפשנו את השכן הקרוב שלה וייצרנו דוגמה סינטטית חדשה בינה.

השתמשתי באלגוריתם SMOTE עד שמספר הדוגמאות של אנשים שקיבלו שbz יהיה זהה במספר הדוגמאות לאלו שלא.

חשוב מאד להשתמש ב-SMOTE רק על נתונים האימון, אחרת יכול להיווצר מצב של data-leakage והמודל שלנו יהיה לא מדויק עבור נתונים אמת. בנוסף, חשוב לבצע את ה-SMOTE **לפניהם** שמבצעים החלוקות לנתונים כמו דיסקרטיזציה, כדי שהדוגמאות הסינטטיות שנוצרות לאחר ריצת האלגוריתם באה שיוצרו מגוונות וקרובות למקו.

---

<sup>1</sup> <https://medium.com/@corymaklin/synthetic-minority-over-sampling-technique-smote-7d419696b88c>

**טבלה מסכמת של הנתונים לאחר סידור ונקיי -**

שם עמודה	תיאור התוכנה	סוג הנתונים	תחומי ערכים	נתוני מחרם	איך טופלה
<b>Gender</b>	מין	בינהריה	0,1	אין	
<b>Age</b>	גיל	נומרי	0,1,2,3,4	אין	בוצעה דיסקרטיזציה שווה עומק
<b>Hypertension</b>	האם יש להם יתר לחץ דם	בינהריה	0,1	אין	
<b>heart_disease</b>	האם חוו מחלת לב	בינהריה	0,1	אין	
<b>ever_married</b>	האם היה נשוי בעבר	בינהריה	0,1	אין	הומר מ-Yes ו-No ל-0 ו-1.
<b>Work_type_self-employd</b>	האם האדם הוא מעסיק עצמאי	בינהריה	0,1	אין	
<b>Work_type_Govt_job</b>	האם האדם הוא עובד ממשלתי	בינהריה	0,1	אין	
<b>Work_type_Children</b>	האם האדם הוא ילך	בינהריה	0,1	אין	
<b>Work_type_never_worked</b>	האם האדם מעולם לא עבד	בינהריה	0,1	אין	
<b>Residence_type</b>	סוג שכונת המגורים שבה התגורר האדם	בינהריה	0,1	אין	
<b>avg_glucose_level</b>	רמת הגלוקוז הממוצעת	נומרי	0,1,2,3,4	אין	נמחקו ערכים קיצוניים, בוצעה דיסקרטיזציה שווה עומק
<b>Bmi</b>	מדד שטרתו לתאר את האם משקל אדם תקין	נומרי	0,1,2,3,4	אין	נמחקו ערכים קיצוניים, ערכים חסרים מולאו בחציו, בוצעה דיסקרטיזציה שווה עומק
<b>smoking_status_formaly_smoked</b>	האם האדם עישן בעברו	בינהריה	0,1	אין	

		אין	0,1	בינהריה	האם האדם מעולם לא עישן	<b>smoking_status_never_smoked</b>
		אין	0,1	בינהריה	האם האדם מעשן	<b>smoking_status_smokes</b>
נשתמש ב- Smote כדי לייצר איזון בנתונים לאחר הפייטול לנתחי אימון ולמידה	אין	0, 1		בינהריה	האם קיבל שbez	<b>Stroke</b>

לאחר ניקוי וסידור הנתונים נשארו לי רק 909 שורות עם סטטוס עישן לא ידוע, החלטתי להוריד  
עמוודה זו כי היא לא מוסיפה לנו מידע מתאים.

לאחר ביצוע ה-SMOTE תוך כדי ה-*h-ohion*-Fold cross validation 10-10 הוי במאגר הלמידה 8750 שורות,  
מתוכן חצי עם 1 == stroke וחצי עם 0 ==stroke באשר רוב השורות לאנשים שיש להם שbez יוצרו  
בצורה סינטטית על ידי האלגוריתם.

## שאלה 2 – סיווג

בחירת השיטות לסיווג הנתוניים

בחرت בשתי שיטות לסיווג הנתוניים:

1. עץ החלטה מבוסס Gini-index
2. Random Forest

ניסיתי את כל השיטות שתיארתי כ-4 חלופות אפשריות לסיווג לביצוע ברית המידע, בחירת מתחום בשתי שיטות אלו כי הן החזרו לי את התוצאות הטובות ביותר והןabic מתאימות לביצוע פעולה הסיווג. Gini-index הוא מודד פיצול המתאים בעיקר לחיזוי תכונות בינאריות וראיתי שפיצול על Information-gain נטה לחסר דיק שנבע מהעדפה בעץ לתכונות רבות משתנים כמו - smoking ו-work-type. בנוסף, שיטת הסיווג RandomForest הכי עמידה בפני Overfitting מבחן status מבחן work-type. שאר האלגוריתמים שמתבססים על עץ החלטה אחד ייחיד.

**פסאודו קוד לכל אחת מהשיטות שנבחרו**

**- Random Forest**

```
# Initialize the RandomForest with parameters
```

```
Initialize RandomForest(num_trees, max_depth, min_samples_split)
```

```
# Fit the RandomForest model
```

```
Function fit(X, y):
```

```
    For each tree in num_trees:
```

```
        X_sample, y_sample = random_sample(X, y)
```

```
        # Train a decision tree
```

```
        tree = train_decision_tree(X_sample, y_sample, max_depth, min_samples_split)
```

```
        # Add the trained tree to the list of trees
```

```
        Add tree to trees
```

```
# Predict using the RandomForest model
```

Function predict(X):

```
# Collect predictions from all trees  
tree_predictions = collect_predictions(trees, X)  
  
# Majority vote to determine final prediction  
final_predictions = majority_vote(tree_predictions)  
  
Return final_predictions
```

### פואודו קוד לעץ החלטה מושג Gini-index

# Initialize the DecisionTree with parameters

```
Initialize DecisionTree(max_depth, min_samples_split)
```

# Fit the DecisionTree model

Function fit(X, y):

```
tree = build_tree(X, y, depth=0)
```

```
Return tree
```

# Build the tree recursively

Function build\_tree(X, y, depth):

```
If stopping_criteria_met(X, y, depth):
```

```
    Return create_leaf(y)
```

```
best_split = find_best_split(X, y)
```

```
left_X, left_y, right_X, right_y = split_data(X, y, best_split)
```

```
left_subtree = build_tree(left_X, left_y, depth + 1)
```

```
right_subtree = build_tree(right_X, right_y, depth + 1)
```

```
Return create_node(best_split, left_subtree, right_subtree)
```

```
# Calculate Gini index
```

```
Function calculate_gini_index(X, y, feature, value):
```

```
    left_y, right_y = split_labels(y, feature, value)
```

```
    gini_left = gini_impurity(left_y)
```

```
    gini_right = gini_impurity(right_y)
```

```
    gini = (gini_left * len(left_y) + gini_right * len(right_y)) / len(y)
```

```
Return gini
```

```
# Calculate Gini impurity
```

```
Function gini_impurity(y):
```

```
    proportions = calculate_proportions(y)
```

```
    gini = 1 - sum(proportions^2)
```

```
Return gini
```

```
# Check stopping criteria
```

```
Function stopping_criteria_met(X, y, depth):
```

```
    If depth >= max_depth or len(y) < min_samples_split or all_labels_same(y):
```

```
        Return True
```

```
    Return False
```

```
# Cart Pruning Algorithm (from Mark Last presentation)
```

1. Initialize the list of optimal trees with the maximal tree
2. Initialize a=0
3. Increase a until the tree ceases to be optimal

4. Find a new sub-tree, which is optimal with a new value of a
5. Add the new sub-tree to the list of optimal trees
6. If the new sub-tree has more than one terminal node, go to Step3. Otherwise, Stop

## תוצאות הנזותחים ומידת הדיקוק של כל שיטה

מכיוון שהוא לנו יחסית מעט שורות במאגר עם אנשים שיש להם שבע, ומכיון שלוחות בדיק גבוה מחולות זה קשה, עוד לפני שראיתי את התוצאות הראשונות הנחתי שההתוצאות **בחיזי אנשים שיש להם שבע לא יהיו טובות**.

ואכן, בתוצאות הראשונות שהמודלים החיזו לי, בשזה הגיע לחיזי ראיתי שהמודלים בקושי הצליכו להבדיל בין חולים שקיבלו שבע לאלו שלא ופספסו הרבה חולים.

ניסיתי לחשב מה עולה יותר – FP (זיהוי שגוי של אדם כפוטנציאלי גבוה לשבע) או FN (פספוס ואין התרעה בפני אדם שעלול לחטוף שבע). קראתי על מקרי שבע ועל טיפול מניעתי וראיתי שבמקרים שבהם יש חשש לשבע, אין צורך בבדיקות מחמירה וקרهة אלא לרוב ממליצים להתחיל לחוות אורח חיים בריאות וריאה או לעשות פעילות ספורטיבית וטיפול זה יחסית אפקטיבי ובמחיר מועט גם לחולה וגם לבית החולים. לעומת זאת, הטיפול לאדם שקיבל שבע הוא יקר, החיים יכולים לההרס, נדרש ליווי קבוע בבית החולים, תרופות וטיפולים יקרים ורבות החולים אינם מתואימים בצורה מלאה מהתקנית. מפאת הסיבות הללו, היה לי קרייטי שהיה לי כמה שפחות FN (אנשים שעומדים לחולות בשבע ולא הזהרתי אותם) ופחות היה אכפת לי מאותי FP גבוהים יותר. האינטינקט שלי אמר לי שכדי לומר להגדר למודלים threshold נמוך יותר מ-0.5 כדי לחזות האם יש לאדם שבע או לא. בעזרתו הפונקציה המובנית predict\_proba בדקתי מה ההסתברות של ה-sample להיות 0 או 1 והגדרתי שזה 0 רק עם ההסתברות קטנה מה-threshold אחרית זה 1. ובחרתי threshold-ים קטנים מ-0.5 כדי להעלות את הסיכוי ל-1 לדוגמה:

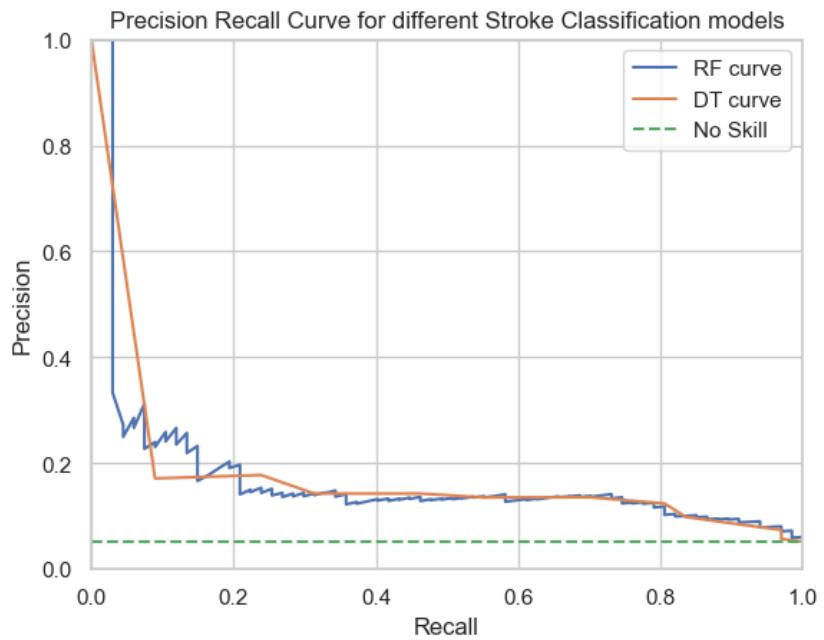
```
threshold = 0.45
y_proba = clf.predict_proba(X_test)[:,1]
y_pred = (y_proba >= threshold).astype("int")
```

כדי לראות מה ה-threshold הטוב ביותר, השתמשתי ב-*precision-recall-curve*, גרף אשר משמש כדי לראות את היחס tradeoff בין recall ל-precision על פני threshold-ים שונים.

اذכיר –

$$recall = \frac{TP}{TP+FN}, precision = \frac{TP}{TP+FP}$$

לבן אשתדל למצוא threshold שיתן לי recall כמה שייותר גבוה (סימן לכמות FN מועטה) ועדין ישמור לי על Precision סביר.

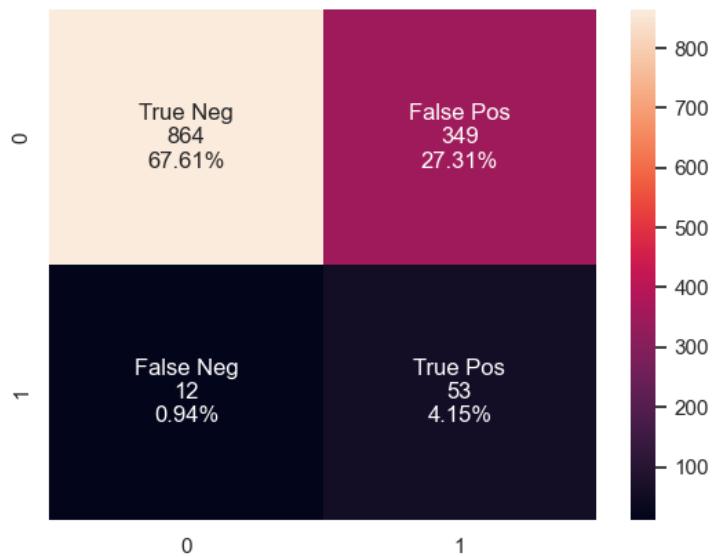


ובכל לראות בנקל מהגרף שכמעט לא משנה מה ה-threshold שאשים, ה-hreshold שלי יהיה נמוך מאד. ולכן ביעוני ל-pdthreshold נמוך מאד (0.15) שיתן לי גובה ו-precision סביר. יחסית אבל אז בשסתובלי על ה-xmatrix confusion ראייתי שהכבות FP שלי יותר מדי גובהה. מכיוון שהנתונים מאד לא מאוזנים, על כל שיפור של חיזוי מדויק על במה נבדקים בודדים שחולים בשbatch אני צריך לדוח על מאות נבדקים נוספים שיש להם פוטנציאל לשbatch למורות זהה לא נכון. לכן, שיחקתי עוד עם הpdthreshold עד שהגעתי לאחד שהייתי מרוצה ממנו ונתן לי את היחס הטוב ביותר שנראה לי סביר. ה-hreshold היה 0.45, זה אכן קטן מ-0.5 במו שצפיתי אבל לא אגרסיבי כמו ששאלוי ה-precision-recall curve יציע.

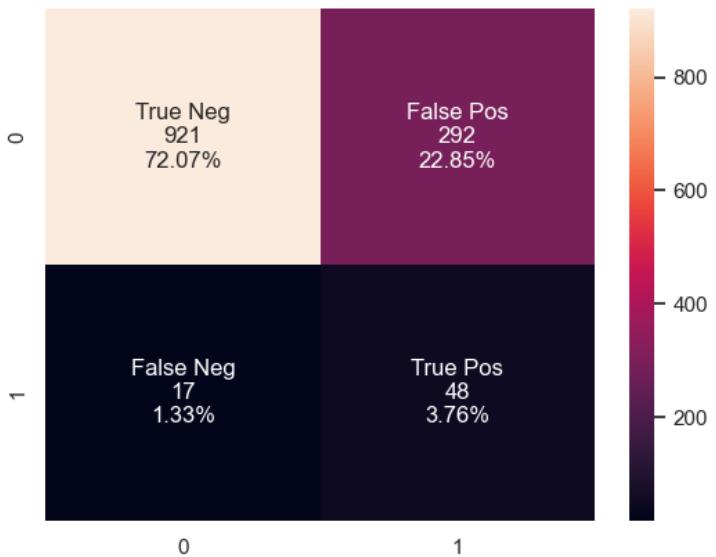
בידוע, יש עוד Settings חוץ מ-pdthreshold שאפשר להוסיף לעצמי החלטה ול-Random Forest. כדי למנוע Overfitting אני הגדרתי לעצמי ההחלטה `max_depth=6 min_samples_leaf = 400` ועבור ה-Random Forest רק `min_samples_leaf=200`.

להלן ה-xmatrix confusion שעשיתי עבור ה-hreshold המתקבל בחלוקת train-test של שני שליש – שלישי לשני המסוגים השונים (כasher לפני הלמידה הורץ SMOTE לאיזון הנתונים על מגאר הלמידה).

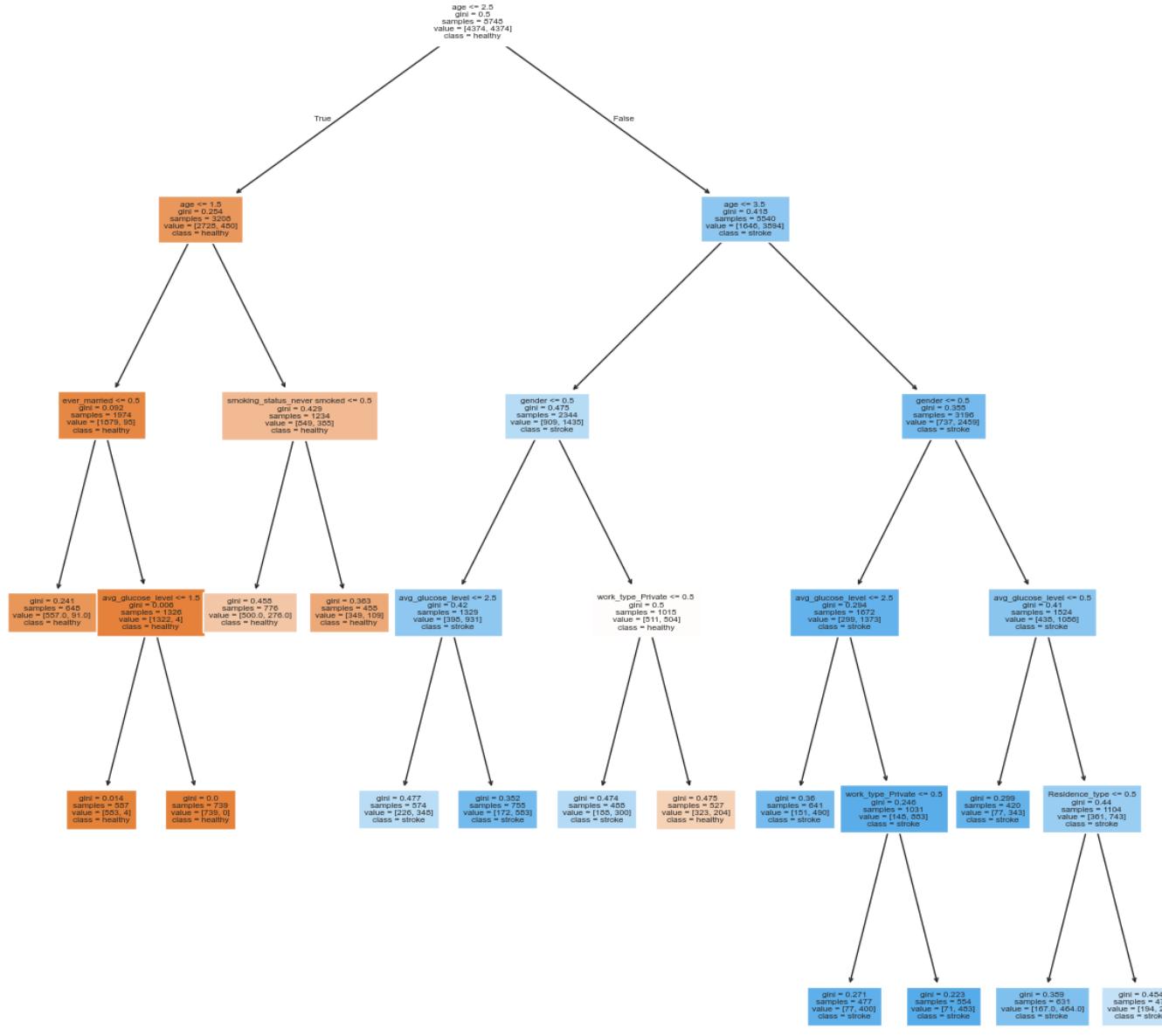
Confusion Matrix - RandomForestClassifier



Confusion Matrix - DecisionTreeClassifier



## אוסף את היזואליציה של ה-Gini-index מבסס DecisionTree שיצרת!

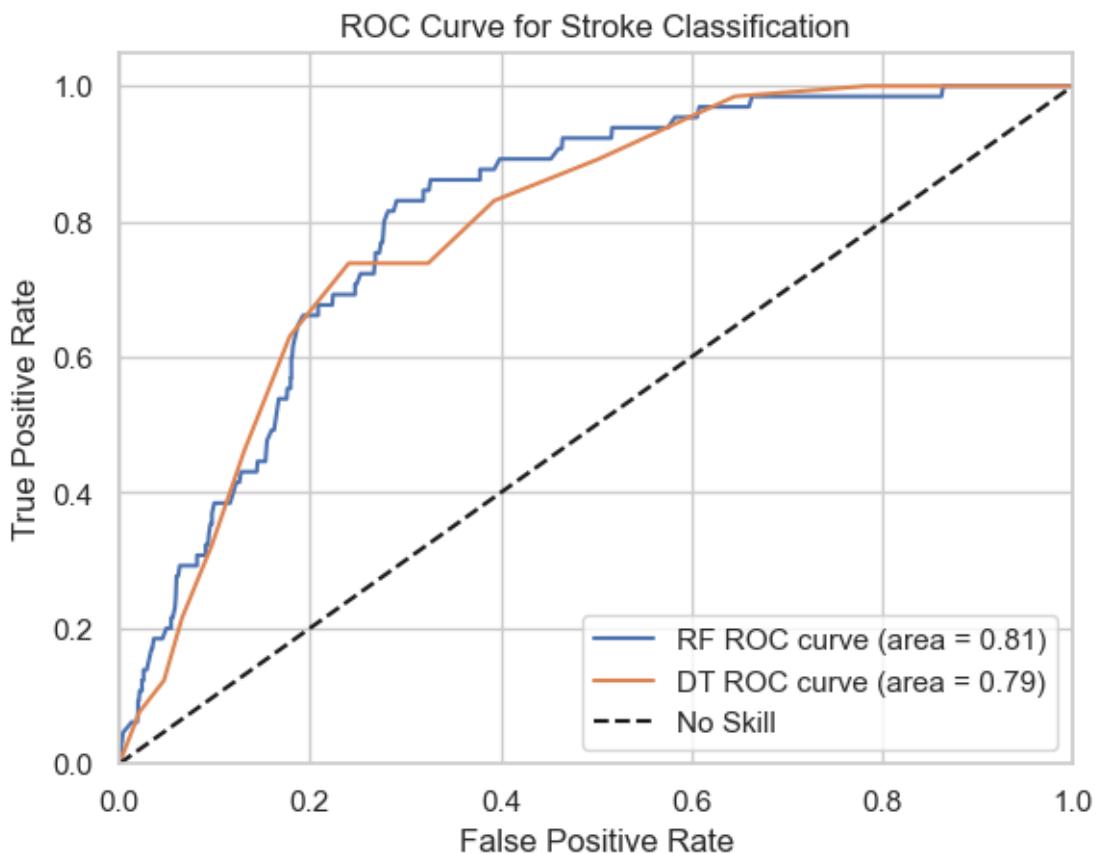


כמו שציפינו בהתחלה, ל-Age יש השפעה גדולה מאוד על הסיווג, היא מופיעה כאן בשלושה פיצולים שונים של העז! בנוסף, AGL שראינו שגם לה יש הרבה קורלציה באופן יחסית עם stroke מתקבלת מהNODE 13.

ל-RandomForest היה קשה יותר לעשות ייזואליציה ולבסוף לא הצליחו את זה.

## ניתוח השוואתי של התוצאות

:ROC Curve



שני המטודולוגיים נוטנים Area Under Curve קרוב ל-0.8 שזו נחשבת תוצאה טובת אך לא מצוינת. מכיוון שמדובר ב>Data Imbalanced, ובמלה האנשים עם stroke ב-test נמוכה מאוד ויכולת להיות עם שוכנות גבוהה שתשפייע משמעותית על התוצאות בחלוקת של שני שליש שלישי בעטתי fold 10 cross validation כדי להתגבר על הרעש שיכול להווצר מכך ולדיק את ההערכה וההשוואה בין המודלים.

ב-fold cross validation 10 אי אפשר ליצור confusion matrix וליצור ROC curve אחד ויחיד, אלא צריך ליצור 10 כאלה או לעשות גרפים שיציגו את הממוצע שלהם. חשוב מאוד לציין שגם כאן כדי לאזן את הנתונים ולתת למודל יותר דגימות להסתאמן עליון, לפני כל הרצה של הסיגוג יוצרו דגימות סינטטיות על גבי נתונים הלמידה (לאחר החלוקה ל-train-i-test) באמצעות אלגוריתם SMOTE שהוזכר לעיל.

בחורתי במקומם להציג השוואה רק בין ממוצע התוצאה שלהם:

Metric	RandomForest	CART gini-index
Accuracy	0.757	0.691
Recall	0.767	0.692
ROC AUC	0.824	0.815

Precision

0.134

0.129

## סיכום ומסקנות

אפשר לראות בטבלת ההשוואה ש-RandomForest טוב יותר מערץ ההחלטה במעט בכל פרמטר. אני מאמין שהסיבה לכך היא בעיקר בזכות של RandomForest להטבש על תוצאות רבות מהרבה עצים אקראים וכך להמנע מתופעת ה-Overfitting שעצ החלטה ייחיד יכול לשבול ממנה בקלות. בנוסף, כל עץ ב-RandomForest לומד על feature-ים שונים וכן הרבה פעמים המודל מצליח להשתמש בתנאים רבים יותר כדי לקבל החלטה ולתפס את המשמעות מאחוריו כל התנאים בניגוד לעצ החלטה ייחיד שבспособו של דבר מtabss על סט התכונות המוצמצם שאצלו בעז.

למרות ש-RandomForest טוב יותר מ-CART כדי לחזות שבע בינוי זה עם תנאים אלו, עדין הוא מודל שלא משיג תוצאות טובות במיוחד. יש לו הרבה מאוד FP ו-Precision נמוך מאוד. ו-Recall ו-Accuracy של מתחת ל-0.8 היא גם תוצאה נמוכה מאוד. חיזוי שבע זה קשה מאוד, גוף האדם הוא מכונה מורכבת מאוד שתלויה בהמון פרמטרים שונים, פנימיים וחיצוניים וכן גם מאוד בלתי צפופה. כמו כן, במקרים הנבדקים שהוא להם שבע מהם יכולים ללמוד על המשותף לחולי שבע היה נמוך מאוד ונאלצנו להשתמש בטכניקות של יצירת דגימות סינטטיות ו-Oversampling כדי להתגבר על כך, פתרון לא אידיאלי שפגע בהצלחת המודל שלנו לכל הדעות.

אני חושב שnitin היה לשפר משמעותית את יכולות המודל שלנו בכמה אופנים –

1. העשרה מאגר הנתונים: אני מאמין שם הינו מקבלים את כל מאגר הנתונים של McKinsey & Company שכולל כמעט 30 אלף רשומות של נבדקים הינו יכולים לקבל עוד שורות רבות ונתחום על אנשים שהוא להם שבע כדיחזק את יכולת הלמידה שלנו על הגורמים לשבע. בנוסף, אם היה אפשר להוסיף עוד עמודות עם תנאים על הנבדקים כמו תוצאות בחינה גופנית, קוגניטיבית וגנטית אולי היו עוד פרמטרים לפיהם הינו יכולים לנסות לאפיין את המשותף לסיכון גבוהה לשבע.

2. שימוש טוב יותר בפרמטרים של CART ושל RandomForest ב-sklearn: אני חושב שיכולים עם משחק נסוף והבנה עמוקה יותר של משמעויות הפרמטרים של המסוגים בפייתן להשיג תוצאות מעט טובות יותר.

3. שימוש באlgorigthmic סיווג נוספים ללא נלמדו בקבוצה של (אך אין נמצאים בהרצאות של מארק): אלגוריתמים כמו AdaBoost, XGBoost ואlgorigthmic חיזוק גרדיאנט נוספים שלא נלמדו בקורס אמרורים להיות טובים יותר מ-RandomForest ופוטנציאלית יכולו לחזות בצורה טובה ממנה נתונים שלנו.

לסיכום, מאוד נהנתי מהפרויקט, אני מאמין מאוד בלמידה דרך הידיים. עד בה בא"פ זה היה הפרויקט שאני מרגיש שדחף אותי הכח הרבה ללמידה עצמאית, קריאה באינטרנט וניסוי וטעייה.

תודה רבה ומוקוה שנהנתם :)