עיבוד אותות אקראיים

עבודה 2 – סיווג סוכרת באמצעות מסווגי KNN,RF,SVM והשוואה ביניהם.

**תוכן עניינים:**

[1.רשימת איורים: 3](#_Toc128760494)

[2.הקדמה ומוטיבציה 4](#_Toc128760495)

[3.הבנת בסיס הנתונים (dataset) – EDA והכנתו לאימון המודלים 5](#_Toc128760496)

[4.הערכת מודלים וביצועים וקריטריון הטיב 9](#_Toc128760497)

[4.1 מטריצת בלבול – Confusion Matrix 9](#_Toc128760498)

[4.2 עקומת ROC ו-AUC 10](#_Toc128760499)

[4.3 קריטריון הטיב 11](#_Toc128760500)

[5.מודלי סיווג 12](#_Toc128760501)

[**5.1.מודל** SVM 12](#_Toc128760502)

[**5.2.מודל** Random Forest 17](#_Toc128760503)

[Decision Tree 5.2.1 17](#_Toc128760504)

[Random Forest 5.2.2 19](#_Toc128760505)

[5.3.מודל KNN 20](#_Toc128760506)

[6.ניסויים 22](#_Toc128760507)

[6.1.מודל knn 22](#_Toc128760508)

[6.2.מודל SVM 24](#_Toc128760509)

[6.3.מודל Random Forest 25](#_Toc128760510)

[7.סיכום ומסקנות 27](#_Toc128760511)

[8.נספח 28](#_Toc128760512)

# 

# 1.רשימת איורים:

[איור 1- הצגת עמודות ראשונות של ה- dataset 5](file:///C:\Users\shayb\Desktop\עיבוד%20אותות%20אקראיים%20-%20עבודת%20מחשב%202חדש.docx#_Toc128760030)

[איור 2- התפלגות כל feature 6](file:///C:\Users\shayb\Desktop\עיבוד%20אותות%20אקראיים%20-%20עבודת%20מחשב%202חדש.docx#_Toc128760031)

[איור 3 - התפלגות כל feature מנורמל 7](#_Toc128760032)

[איור 4 - מפת חום של התכונות 8](#_Toc128760033)

[איור 5 - מטריצת בלבול או בשמה הלועזי confusion matrix 10](file:///C:\Users\shayb\Desktop\עיבוד%20אותות%20אקראיים%20-%20עבודת%20מחשב%202חדש.docx#_Toc128760034)

[איור 6 - גרף ROC 11](file:///C:\Users\shayb\Desktop\עיבוד%20אותות%20אקראיים%20-%20עבודת%20מחשב%202חדש.docx#_Toc128760035)

[איור 7 - הצגת ROC עבור thresholds שונים 12](file:///C:\Users\shayb\Desktop\עיבוד%20אותות%20אקראיים%20-%20עבודת%20מחשב%202חדש.docx#_Toc128760036)

[איור 8- מגוון אופציות לhyperplane לעומת האופטימלי 13](file:///C:\Users\shayb\Desktop\עיבוד%20אותות%20אקראיים%20-%20עבודת%20מחשב%202חדש.docx#_Toc128760037)

[איור 9 - דוגמא לSVM עבור מרחב תלת מימדי 13](#_Toc128760038)

[איור 10 - דוגמא לsvm בתלות בפרמטר C 15](#_Toc128760039)

[איור 11 - data שלא מופרד בצורה ליניארית 16](file:///C:\Users\shayb\Desktop\עיבוד%20אותות%20אקראיים%20-%20עבודת%20מחשב%202חדש.docx#_Toc128760040)

[איור 12- דוגמא לחלוקת הציר בהתאם לשאלות בעץ ההחלטה 18](#_Toc128760041)

[איור 13 - דוגמא לתהליך של ה RF 20](file:///C:\Users\shayb\Desktop\עיבוד%20אותות%20אקראיים%20-%20עבודת%20מחשב%202חדש.docx#_Toc128760042)

[איור 14 - דוגמא לתהליך ה knn עבור k=5 ו classes=3 21](file:///C:\Users\shayb\Desktop\עיבוד%20אותות%20אקראיים%20-%20עבודת%20מחשב%202חדש.docx#_Toc128760043)

[איור 15 - השפעת k על מרחב הסיווג 22](file:///C:\Users\shayb\Desktop\עיבוד%20אותות%20אקראיים%20-%20עבודת%20מחשב%202חדש.docx#_Toc128760044)

[איור 16 – confusion matrix עבור knn ללא בחירת k אופטימלי 23](file:///C:\Users\shayb\Desktop\עיבוד%20אותות%20אקראיים%20-%20עבודת%20מחשב%202חדש.docx#_Toc128760045)

[איור 17 - גרף הroc עבור k ברירת מחדל 24](file:///C:\Users\shayb\Desktop\עיבוד%20אותות%20אקראיים%20-%20עבודת%20מחשב%202חדש.docx#_Toc128760046)

[איור 18- confusion matrix עבור k אופטימלי ל accuracy 24](file:///C:\Users\shayb\Desktop\עיבוד%20אותות%20אקראיים%20-%20עבודת%20מחשב%202חדש.docx#_Toc128760047)

[איור 19 - גרף הroc עבור k אופטימלי 25](file:///C:\Users\shayb\Desktop\עיבוד%20אותות%20אקראיים%20-%20עבודת%20מחשב%202חדש.docx#_Toc128760048)

[איור 20 - confusion matrix עבור מודל svm אופטימלי 25](file:///C:\Users\shayb\Desktop\עיבוד%20אותות%20אקראיים%20-%20עבודת%20מחשב%202חדש.docx#_Toc128760049)

[איור 21 - confusion matrix עבור מודל random forest אופטימלי 26](file:///C:\Users\shayb\Desktop\עיבוד%20אותות%20אקראיים%20-%20עבודת%20מחשב%202חדש.docx#_Toc128760050)

[איור 22- גרף הroc עבור svm אופטימלי 26](file:///C:\Users\shayb\Desktop\עיבוד%20אותות%20אקראיים%20-%20עבודת%20מחשב%202חדש.docx#_Toc128760051)

[איור 23 - גרף הroc עבור random foret אופטימלי 27](file:///C:\Users\shayb\Desktop\עיבוד%20אותות%20אקראיים%20-%20עבודת%20מחשב%202חדש.docx#_Toc128760052)

[איור 24- חשיבות כל feature ביצירת העץ 27](file:///C:\Users\shayb\Desktop\עיבוד%20אותות%20אקראיים%20-%20עבודת%20מחשב%202חדש.docx#_Toc128760053)

# 2.הקדמה ומוטיבציה

סוכרת הינה מחלה כרונית המשפיעה על רמת הסוכר בדם. המחלה עשויה לגרום למחלות לב, כליות, כלי דם ועיניים. גילוי מוקדם וטיפול יעיל בסוכרת חיוניים במניעת סיבוכים ובשיפור תוצאות המטופל. כדי להשיג זאת, אנשי מקצוע רפואיים דורשים כלים מדויקים לאבחון וסיווג הסוכרת. גישה מבטיחה אחת היא השימוש באלגוריתמים של למידת מכונה, שיכולים לנתח מערכי נתונים גדולים ולספק תוצאות סיווג מדויקות.

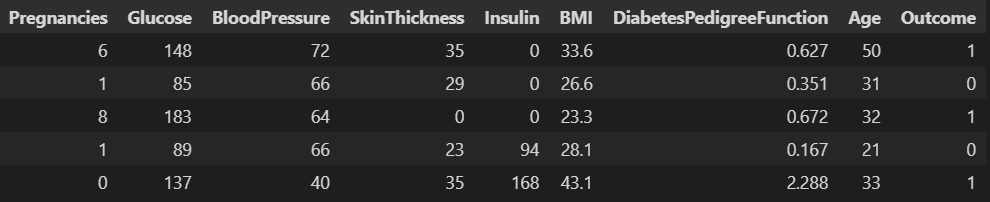
במסגרת הפרויקט בצענו לימוד וחקר של שלושה אלגוריתמי למידת מכונה מפוקחת, נלמד ונכיר את המודלים, נתנסה באימון המודלים ובחירת הפרמטרים השונים בהם המשפיעים על ביצועי המודלים לאחר מכן נערוך השוואה בין שלושה מסווגי Supervised Machine Learning אשר כל אחד מהם מהווה מודל שבעזרתו ניתן לבצע חיזוי של סוכרת. את האימון ואת החיזוי נבצע על dataset שנקרא "Pima Indians diabetes ". קובץ זה מכיל מאפיינים שאדם יכול להוציא מבדיקת דם פשוטה, ולקבל אינדיקציה ראשונית לגבי האם הוא חיובי או שלילי למחלה.

בעבודה זו, נתחיל בביצוע EDA מקיף להבנת הdataset כאשר נבחן את ההתפלגויות של כל מאפיין ומה מידת השפעתו, מתוך הEDA נראה מה ניתן לעשות כדי להפוך את ה data למתאים יותר למסווגים מהסוגים שבחרנו. לאחר מכן- נאמן שלושה מודלים שונים, כאשר נבחן כל אחד מהם בנפרד ונציג גרפים רלוונטיים עבור כל אחד מהמודלים ולבסוף נכריע מי המודל המתאים ביותר לטיפול בבעיה שלנו.

מטרת פרויקט זה היא להשוות את הביצועים של שלושה מסווגים עבור סיווג סוכרת. על ידי ניתוח של כל מסווג, נוכל לקבוע מי המתאים ביותר לסיווג סוכרת.

# 3.הבנת בסיס הנתונים (dataset) – EDA והכנתו לאימון המודלים

בשלב הראשון, נבדוק מהם המאפיינים מהם מורכב בסיס הנתונים שלנו:



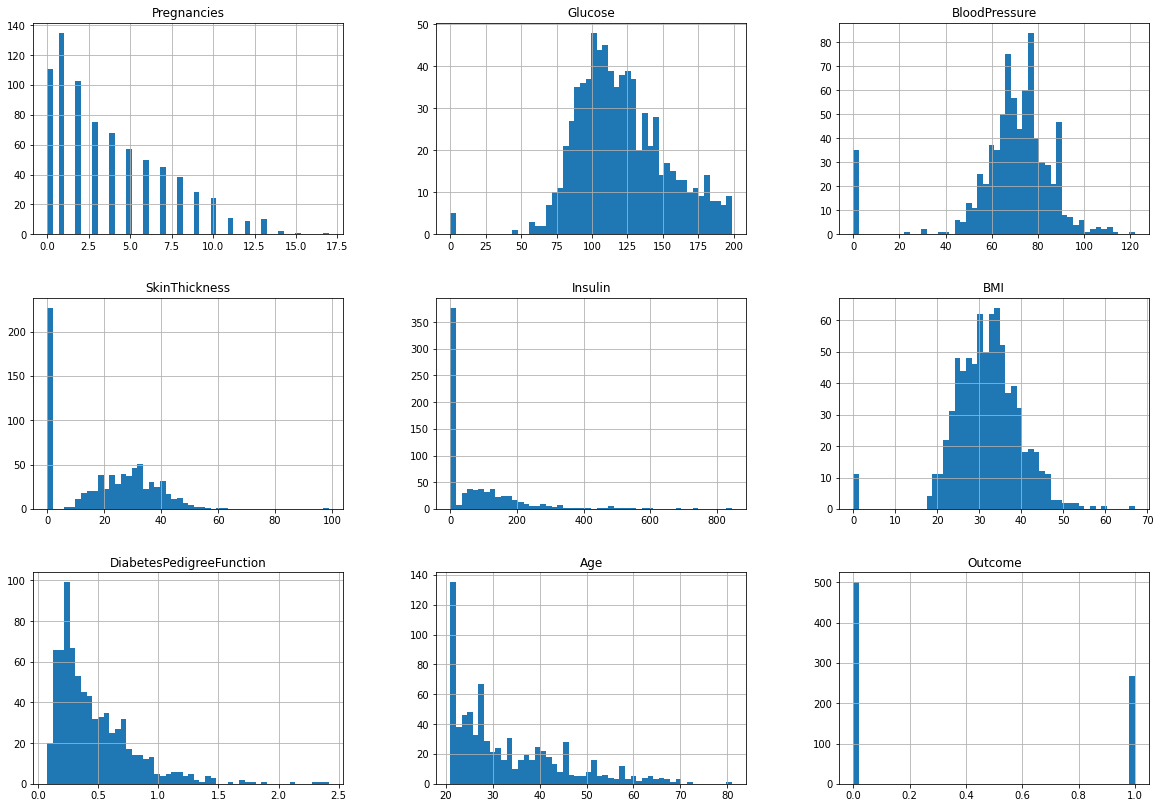
איור 1- הצגת עמודות ראשונות של ה- dataset

מאיור 1, ניתן לראות כיצד נראה הdataset מאילו מאפיינים הוא מורכב ומהם הערכים שניתנים לכל מאפיין כאשר ידוע לנו כי הוא מכיל כ – 768 מטופלים, כאשר העמודה הימנית היא העמודה שאומרת לנו האם מטופל הוא חולה (1) או לא חולה (0). כעת נרצה לראות האם הdataset שלנו מאוזן, כלומר האם יש כמות שווה של חולי סכרת לעומת לא חולי סכרת, בדיקה זו חשובה מאחר ומאוד קשה לאמן data לא מאוזן.

טבלה 1- כמות חולים בסכרת לעומת לא חולים

|  |  |
| --- | --- |
| **חולה – 1** | **לא חולה – 0** |
| 268 | 500 |

ניתן לראות כי יש בערך חולים לעומת  *של בריאים ולכן אמנם הבסיס נתונים לא מאוזן לחלוטין אך מאוזן בצורה סבירה על מנת לעבוד אתו איך שהוא ללא עבודות איזון מקדימות.* כעת נחפש האם יש מידע שלא הוזן כראוי או שלא הוזן בכלל - דבר שיכול למנוע מהמודל להתאמן בכלל , בצענו בדיקה שבה חפשנו Nan- Not a Number ולשמחתנו לא מצאנו דבר. לאחר מכן חפשנו outliers – נתונים הרחוקים בצורה משמעותית משאר הנתונים נתונים אלה משפיעים מאוד על אימון המסווג מאחר ואלו הם נתונים בודדים שמשפיעים רבות על התנהגות המודל כולו ויכולים לגרום ל overfitting ולבסוף למודל פחות טוב.



איור 2- התפלגות כל feature

בבחינת כל הסטוגרמה ראינו משהו מוזר, כיצד יכול להיות לאדם **BMI** או **BloodPressure** **Insulin**, **SkinThickness** השווה ל-0.

הגענו למסקנה כי אלו בדיקות שלא נלקחו ולכן במקום להשאיר Nan החליטו להציב 0. Outliers אלה יכולים להשפיע על ביצועי המודל שכן הם בהחלט מבלבלים את המודל בשלב האימון שהרי בבחינת test או evaluation של המודל אנחנו ככל הנראה לא ניקח בחשבון מקרה קצה כזה במיוחד כאשר ניתן לראות שרק חלק מהמאפיינים הם כאלה, כלומר הנחנו שלא היה מראש כוונה לאמן את המודל גם כאשר חסרים מאפיינים מסוימים.

ולכן ניצבנו בפני דילמה כיצד לנהוג בשלב זה על מנת להימנע מהבעיה הזו כאשר היו לנו 2 אופציות:

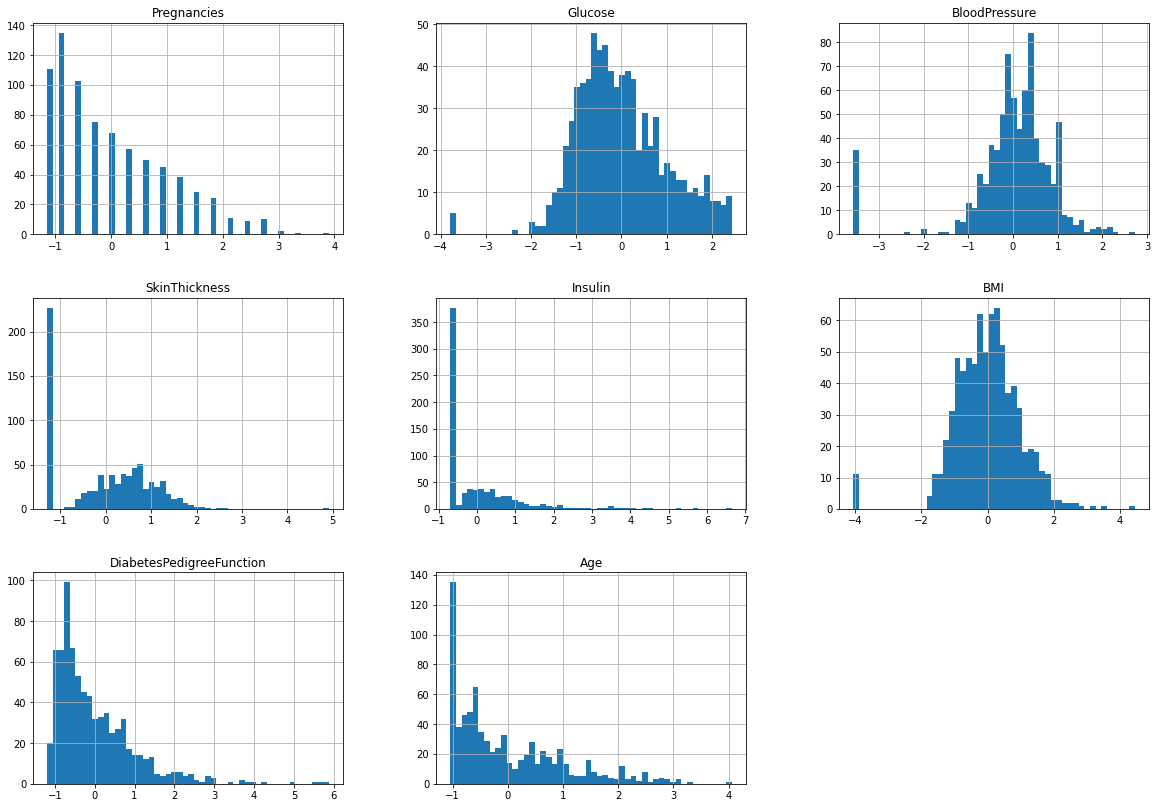
1. השארת האפסים ונירמול כולל של כל מבנה המידע מה שיגרום להם להשפיע פחות.
2. החלפת הערכים בערך מומצא שמתקרב לממוצע הערכים של אותה התכונה במבנה המידע. כלומר לסקור את ההתפלגות של כל התכונות השונות מ-0 בעמודה זו ולייצר התפלגות נורמלית בעלת תוחלת ושונות ואז להגריל לתוך האפסים ערכים מתוך ההתפלגות הזו.

לבסוף החלטנו באופציה 1 שכוללת נרמול מבנה המידע הכולל. בחרנו לנרמל את המידע באמצעות טכניקה הנקראת z normalization . טכניקה זו נועדה לנרמל מידע מספרי למבנה מידע בעל תוחלת 0 ושונות של 1 :

כאשר :

x – הערך של התכונה הספציפית

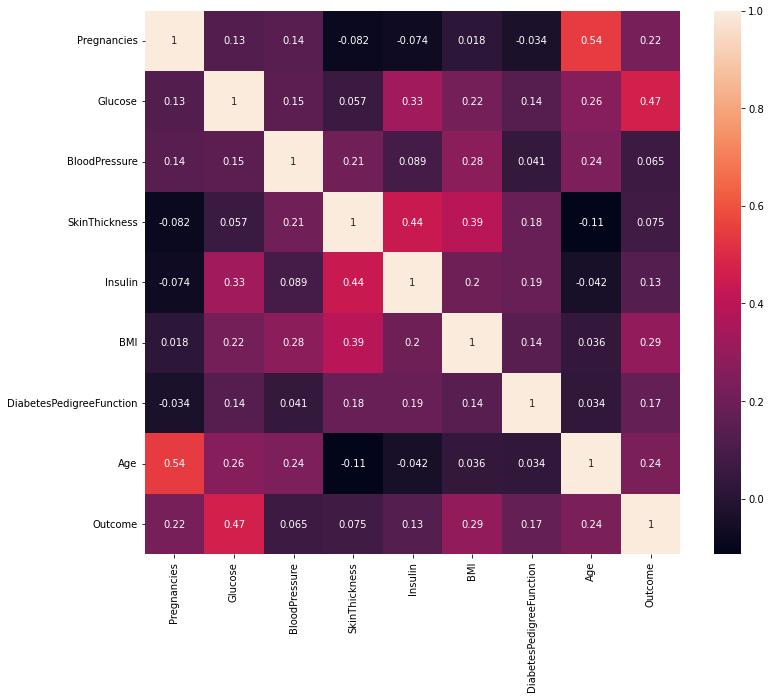
ֹ – התוחלת של כלל הערכים של התכונה הספציפית

- השונות של כלל הערכים של התכונה הספציפית

איור 3 - התפלגות כל feature מנורמל

כפי שניתן לראות, כמובן שלא נרמלנו את עמודת ה-  **.** מאחר ואנחנו לא מקבלים testset נפרד, נבצע חלוקה של ה dataset שלנו ל train, test כאשר כמובן שאת חלק הtest נשאיר "מוחבא" עד לסיום אימון המודל ולבסוף נבצע את הבדיקה עליו, **כלומר הוא עובר נורמליזציה בהתאם לתוחלת והשונות של ה train מאחר ובתיאוריה אנחנו לא מכירים אותו.** חלקנו את ה dataset לפי קריטריון של 30-70 . כלומר, 70 אחוז מהמבנה מידע מועבר לצורך אימון המודל ו30 אחוז ממנו עובר לבדיקות בלבד.

לבסוף נציג מפת חום של מבנה המידע כדי לראות את הקורלציה בין כל feature במבנה המידע:



איור 4 - מפת חום של התכונות

כלומר ניתן לראות שעבור שיש קורלציה יחסית גדולה עבור ה outcome לבין תכונת הglucose . כל שורה מייצגת את הקורלציה בין כל אחד מהתכונות לבין שאר התכונות. ניתן לראות שהמטריצה סימטרית כלומר ניתן להסתכל על השורות בכל פעם.

נסכם את הצעדים שבצענו על מנת להכין את ה data שלנו לפני עבודה על המודלים:

1. קביעת seed אשר מגדיר כי בכל פעם שנריץ את התוכנית המידע יפוצל בדיוק באותו סדר. שלב זה בוצע מאחר ואם לא נגדיר seed נקבל בכל הרצה של המודל חלוקה אחרת ולא נוכל לקבוע האם המודלים שלנו התאמנו והשיפור בזיהוי חל בעקבות שיפור הפרמטרים בהם או שהוא נקבע בעקבות החלוקה השונה של מבנה המידע . במילים אחרות קיבוע ה seed מבטיח לנו תנאי התחלה שווים בכל פעם ומבטיח לנו שאם חל שיפור בזיהוי מודל זה בגלל שהוא באמת השתפר ולא כי יצא במקרה שכעת הוא מסווג על data ״נוח יותר״.
2. בדיקה וניקוי של מבנה המידע כך שכל הערכים הם מספריים ואין מקום שבו אין ערך.
3. זיהוי outliers וביצוע נירמול על מנת לגרום להם להשפעה נמוכה יותר ולכך שהמודל לא יתבלבל ולא יבצע עליהם over fitting (מצב שבו המודל מתאים את עצמו יותר מידי לסט האימון ועבור דוגמאות אימון מביא תוצאות מצוינות אך עבור סט ואלידציה או בדיקה מביא תוצאות נמוכות הרבה יותר.)
4. פיצול מבנה המידע לסט אימון ולסט בדיקה כך שהמודל מתאמן אך ורק על סט האימון וסט הבדיקה שעליו נבדוק את המודל, זהו מבנה מידע שהמודל לא ראה מעולם וממנו נשלוף את כל הקריטריוני טיב שלפיהם נחליט האם המודל ביצע עבודה טובה שמא לא.

# 4.הערכת מודלים וביצועים וקריטריון הטיב

## 4.1 מטריצת בלבול – Confusion Matrix

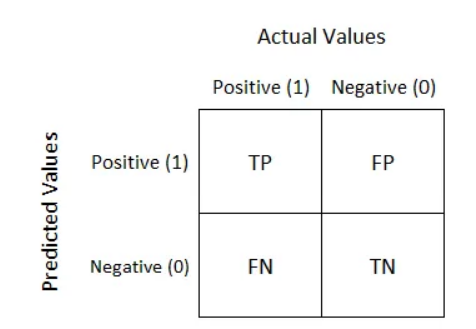
מטריצת "בלבול" הינה מטריצה שבעזרתה מעריכים ביצועים של מודל סיווג – קלסיפיקציה בעיקר קלסיפיקציה בינארית כלומר 2 מחלקות, זו מטריצה של 2x2 כאשר היא מחולקת באופן הבא :

True Positive – סווגנו דוגמא כחיובית והיא אכן חיובית

True Negative – סווגנו דוגמא כשלילית והיא אכן שלילית

False Positive – סווגנו דוגמא כחיובית והיא שלילית

False Negative – סווגנו דוגמא שלילית והיא חיובית



איור 5 - מטריצת בלבול או בשמה הלועזי confusion matrix

מתוך התוצאות האלה אנחנו נוכל לקבל מדדים שיוכלו לעזור לנו להגיד כמה מסווג מסוים מתאים למקרה שלנו.

נגדיר כמה מדדים שנלקחים מתוך מטריצת הבלבול :

מדד זה מתאר כמה מתוך הקריאות החיוביות אכן נכונות מדד זה הינו מדד ל "רגישות" כמות הדוגמאות החיוביות.

מדד זה מתאר את כמות הקריאות השליליות הנכונות (סווג שלילי ואכן שלילי) לעומת כל הקריאות השליליות.

מדד זה מתאר כמה מתוך כל הדוגמאות שסווגנו כחיוביות הם אכן חיוביות.

מדד זה מתאר את הדיוק של הדוגמאות שסיווגנו למול כל הדוגמות, כלומר כמה דוגמאות סווגנו נכון מבין כל הדוגמאות.

מדד זה נקבע על מנת להשוות בין מודלים שיש לאחד מהם precision נמוך וrecall גבוה ולשני הפוך. מדד זה מייצג מעין ממוצע בין 2 המדדים האלו כך שיהיה ניתן להשוות בין 2 מסווגים בצורה כללית יותר.

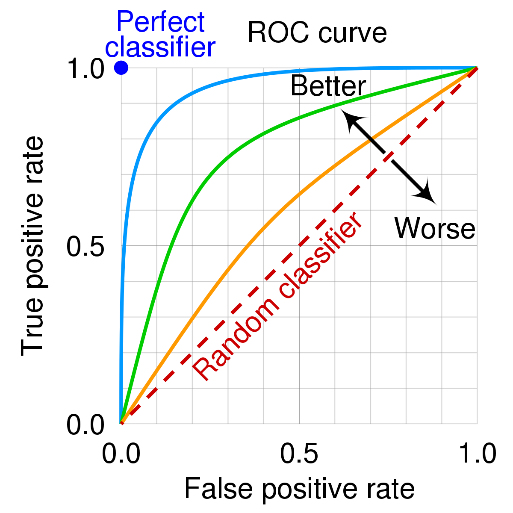
במידה ונרצה לסווג בעיה של "חיים ומוות" למשל זיהוי של מחלה מסוימת, נעדיף לקבל יותר קריאות שווא מאשר פספוס של קריאות אמת במצב כזה ניתן משקל רב יותר בהערכת המודל למדד

מדד זה מתאר כמה מתוך הקריאות החיוביות שסווגו בתור ״שקר״ אכן נכונות. באמצעות (6,1) ניתן לייצר עקומת ROC עליה נפרט בהמשך.

מדד זה הינו קשר בין נוסחאות 6,1 ומגדירה בעצם מדד נוסף על מנת לחשב את הרגישות המשותפת של כל class .

## 4.2 עקומת ROC ו-AUC

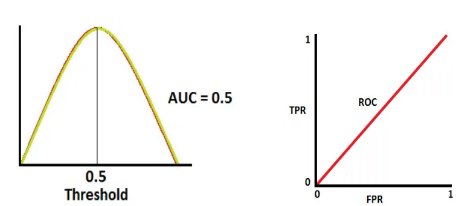
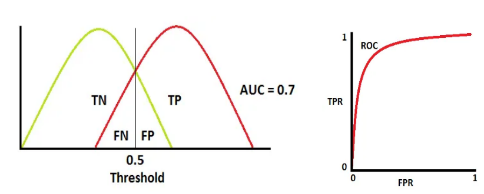
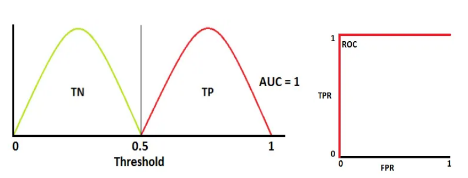
עקומת ROC הינה העקומה המקשרת בין מדד TPR – Recall לבין FPR עבור ערכי thresholds שונים. בבעיות קלסיפיקציה(בעיה בינארית בפרט) נסתמך על עקומה זו על מנת לאפיין את המסווג. עקומה זו מיוצרת על ידי חיבור של נקודות שונות שנוצרות עקב thresholds שונים. הקטנת ה threshold מסווג יותר דוגמאות כחיוביות ולכן מגדיל הן את הFP והן את הTP.



איור 6 - גרף ROC

מאיור 6 ניתן גם לראות את הקשר בין (7) לבין נוסחאות (1,6) כאשר ככל שמעלים את ה tpr אכן מתקרבים לנקודה האופטימלית וכאשר הולכים שמאלה כלומר מקטינים את ה fpr גם כן מתקרבים לנקודה האופטימלית.

**AUC** – השטח שמתחת לגרף הROC, מדד זה מייצג כמה המודל מוצלח בזיהוי מוצלח בין ה classes השונים. ככל שמודל טוב יותר, ה-AUC מתקרב ל1. נדגים כיצד עבור thresholds שונים מקבלים נקודות אחרות על הגרף נניח והdataset- שלנו מתפלג בצורה הבאה :



איור 7 - הצגת ROC עבור thresholds שונים

ניתן לראות כי עבור התפלגויות שונות ו threshold קבוע נקבל גרפים שונים וכן יהיה עבור אותה התפלגות וערכי threshold שונים.

## 4.3 קריטריון הטיב

מאחר ואנחנו מבצעים בדיקות עבור מסווגי למידת מכונה ובפרט מסווגים בינארים מדד ההערכה במקובל הינו הצגת מטריצת בלבול כאשר מתוך טבלה זו נוכל להסיק את כל המדדים הרלוונטים. כפי שציינו מאחר ואנחנו מתעסקים בסיווג בין חולי סכרת ל-לא חולים ומאחר וזו בעיה מסוכנת נשים דגש על שיטת הערכה ש״מעדיפה״ לטעות בקריאות שווא מאשר לפספס קריאת אמת משמע Recall .

כמובן שנתחשב בכך שבהצעת העבודה סכמנו על קריטריון טיב לפיו נבדוק את המודל שהוא הצגת מטריצת הבלבול, מדד Gmeanמדד f score ועקומת roc.

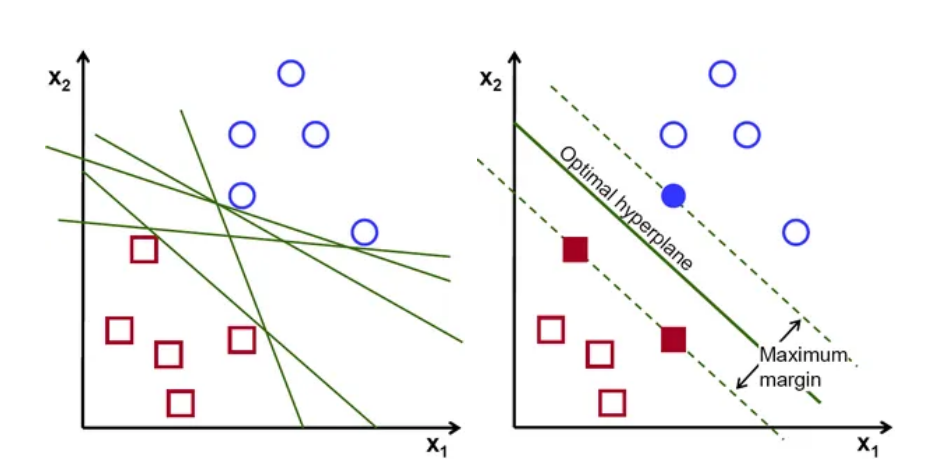
לבסוף נבצע השוואה בין ציוני המודלים השונים.

# 5.מודלי סיווג

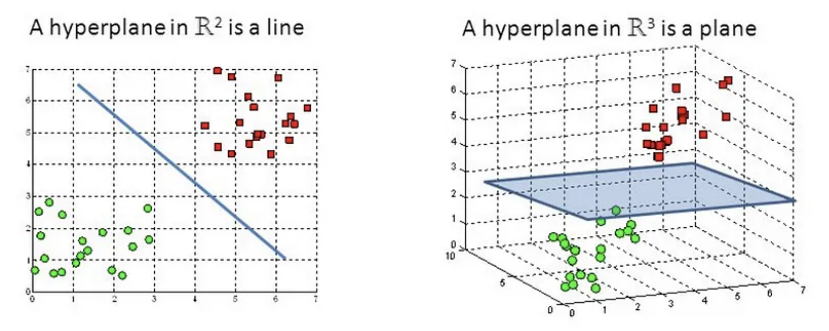
## **5.1.מודל** SVM

מכונת תמך וקטורי – SVM הוא טכניקה של למידה מונחית המשמשת לניתוח נתונים לסיווג ורגרסיה. עבור מודל זה, דוגמאות האימון מיוצגות כווקטורים במרחב ליניארי. עבור בעיות סיווג, בשלב האימון מתאימים מסווג שמפריד נכון ככל הניתן בין 2 ה classes. האלגוריתם תחילה מוצא את ה"הפרדה" – hyperplane אשר מפריד בין הדוגמאות השייכות לclass 0 ואלו השייכות ל class 1 .

ל SVM כמה סוגים כאשר ביניהם המסווג הליניארי אשר לוקח את הhyperplane עם המרווח הגדול ככל האפשר בינו לבין הדוגמאות הקרובות לו ביותר בין 2 ה classes . הנקודות – features אשר "קובעים" את הmargin (ראה איור 7) נקראים supporting vectors.



איור 8- מגוון אופציות לhyperplane לעומת האופטימלי



איור 9 - דוגמא לSVM עבור מרחב תלת מימדי

חשוב לציין כי מימד ה hyperplane נגזר ממימד הfeatures. עבור נקודה-דוגמא חדשה (test data) האלגוריתם יזהה האם היא ממוקמת בתוך הקו המגדיר class מסוים או מחוץ לה.

**נפרט על המתמטיקה:**

המטרה היא להפריד את וקטורי המידע באמצעות hyperplane כאשר המרחק בינו לבין הנקודות הקרובות ביותר מקסימלי. (כמתואר באיור 8 בצד ימין).

נגדיר פונקציה הנקראת decision role function:

unknown new vector

- perpendicular vector to hyperplane

כלומר אנחנו מסווגים נקודה כאשר אם היא נמצאת בצד ימין של ה נקבל ערך גדול שווה מ-1 וההפך.

כעת נגדיר זאת באמצעות נוסחאות :

כעת נגדיר משתנה y בשביל לכתוב את הביטוי בצורה נוחה יותר :

- perpendicular vector to hyperplane

– data points

– constant

*– the class of data point normalized to -1 and 1*

כעת עבור ה support vectors נקבל את המשוואה הבאה :

מטרתנו היא למקסם את ה margin כפי שהוא נראה באיור (8) ולפי המשוואה לעיל אנחנו יודעים כעת מי הם נקודות שקובעות את ה margin ולכן נקח את וקטור ההפרש ביניהם ונכפיל אותו בוקטור יחידה על מנת לקבל את הרוחב

נניח ו הן נקודות ה supporting vectors אזי

כעת נציב לתוך המשוואה את משוואה 5 עבור 2 הנקודות ונקבל :

כעת על מנת למצוא את המרחק הגדול ביותר נגזור נגדיר:

המעבר האחרון בוצע על מנת להקל את המשך הפיתוח.

כעת מאחר ואנחנו רוצים למצוא את הקיצון של ביטוי שמאולץ לביטוי אחר נגדיר את כופלי לגרנז׳ עבור הביטוי, עם הביטוי נוכל לבצע מציאת קיצון

כעת נגזור לפי w ונשווה את הביטוי ל-0 כדי למצוא את הקיצון.

בשביל שנוכל להציב חזרה לביטוי (9) נרצה לגזור גם לפי b

וכעת נציב חזרה את ביטויים )11,12 (לתוך (9) :

ממשוואה (13) ניתן לראות כי המרחק האופטימלי תלוי אך ורק במכפלה בין 2 זוגי דגימות. כעת ניתן להציב חזרה לתוך משוואת ההחלטה (1) ולקבל:

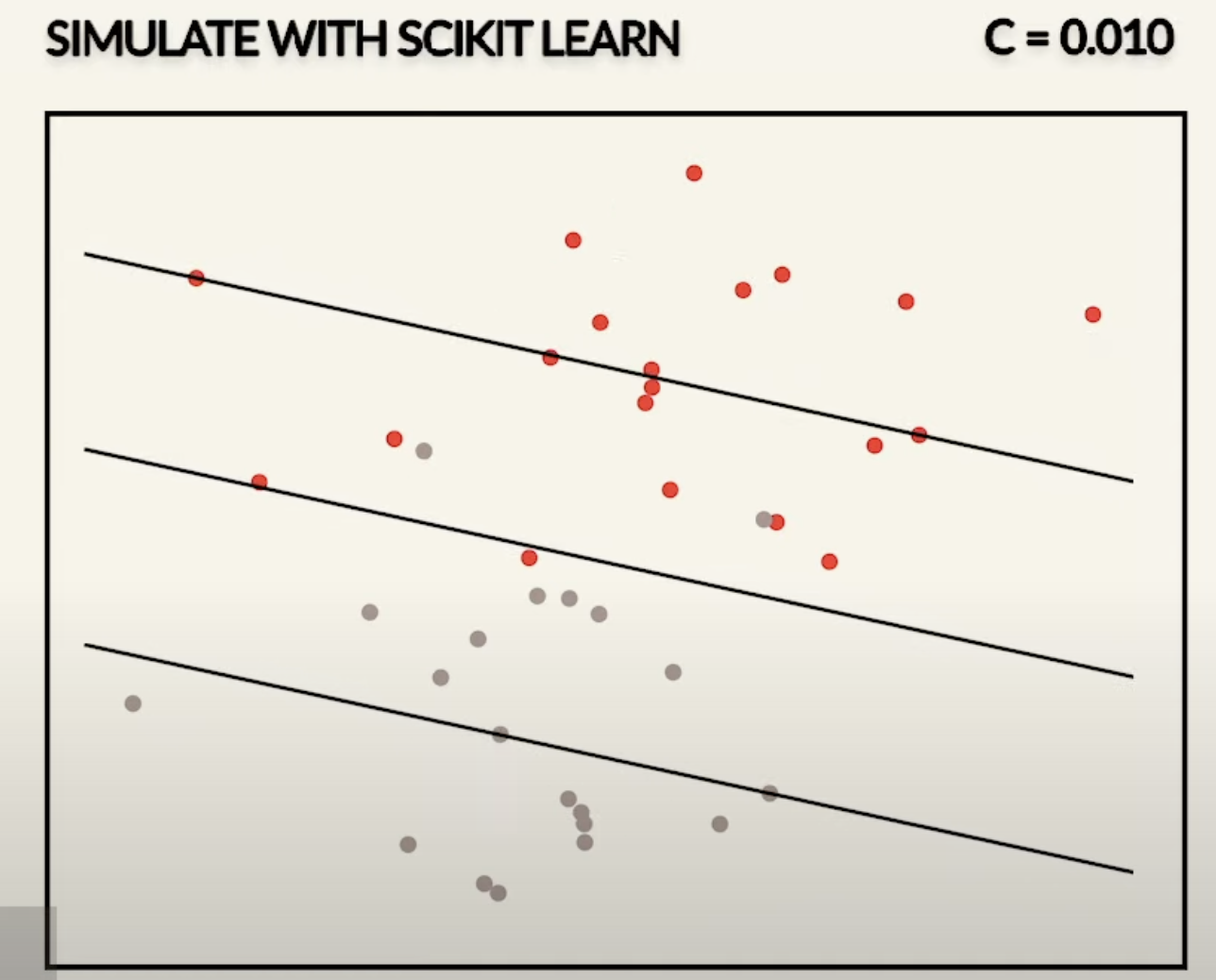
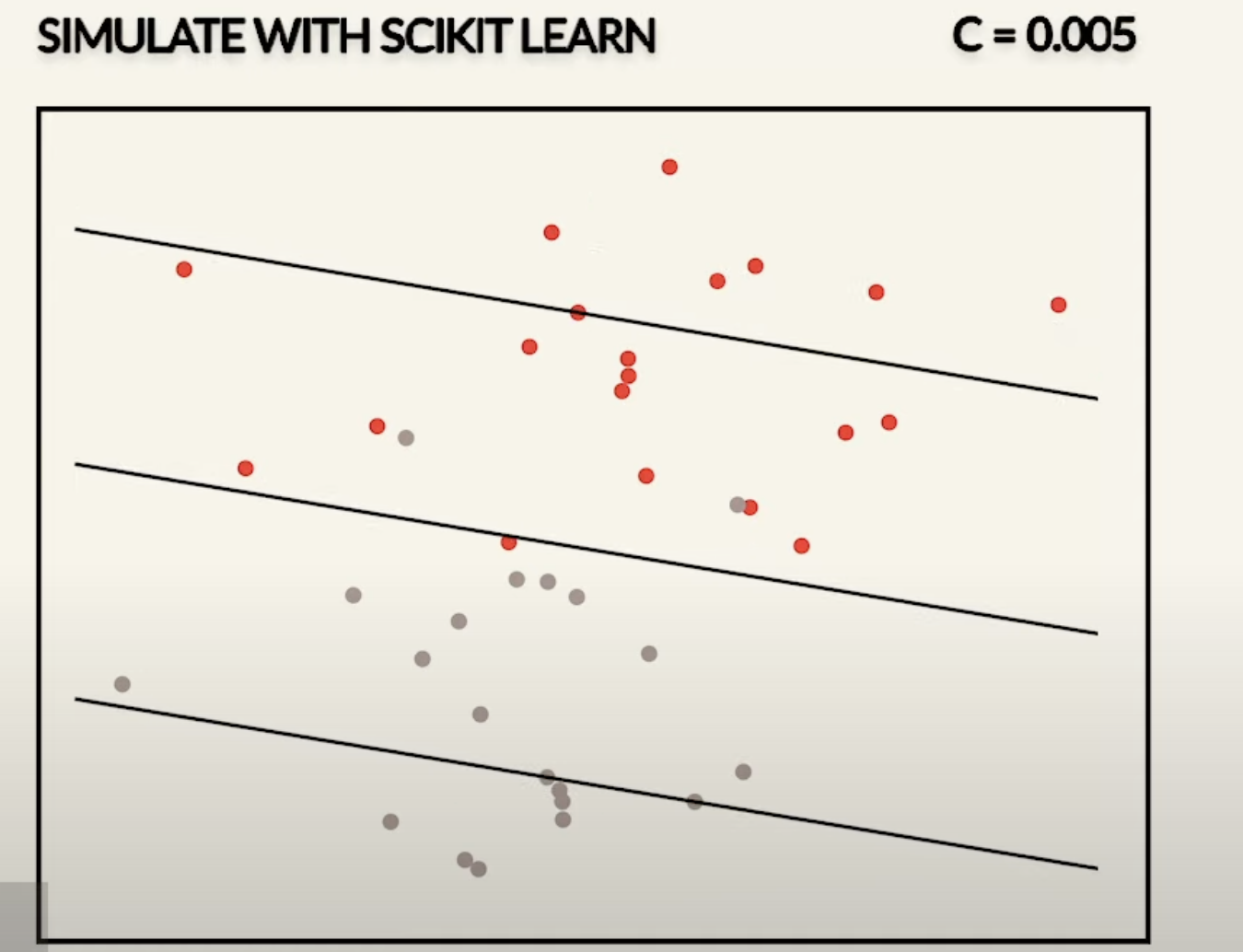
כל הפיתוח בוצע עבור המקרה בו כל מבנה המידע מופרד בצורה לינארית ללא outliers אך אם יש ולו אחד המודל שפתחנו כבר לא יתאים. לשם כך פותחה שיטת ה soft margin

בשיטת ה soft margin מתווסף גורם :

כעת ניתנה גם האופציה לקיים את המשוואה עבור outliers . ולכן הוגדרה תוספת נוספת שמגדירה כמה המודל ״מעניש״ את המשקולות עבור סיווג לא נכון.

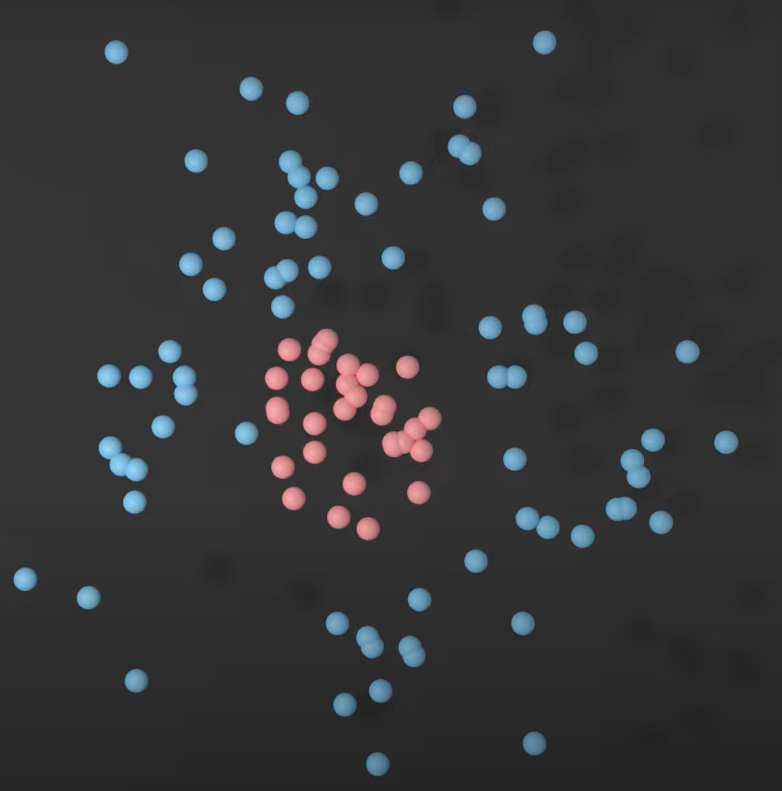
ולכן הפונקציה שאנחנו מחפשים היא :

כאשר C הוא פרמטר רגולציה שאומר כמה אנחנו רוצים להימנע מזיהוי לא נכון של נקודה מסוימת, ככל שהוא גדול יותר כך ל יש פחות השפעה ולהפך. כאשר עבור c ששואף לאינסוף נקבל מסווג שדומה יותר ל Hard margin .



איור 10 - דוגמא לsvm בתלות בפרמטר C

אחת הבעיות באלגוריתם זה, היא שה data ככל הנראה לא יהיה מסודר בצורה נוחה כמו באיורים 8,9 ולכן קו לינארי לא יצליח לבצע הפרדה בין 2 ה classes . למשל עבור data מהסוג שבאיור 11 קו לינארי לא יכול לבצע הפרדה בין 2 הclasses .



איור 11 - data שלא מופרד בצורה ליניארית

לשם כך, פותחה שיטה שנקראת "The Kernel" , מטרתה היא לבצע טרנספוציה ל data על מנת שלאחר הטרנספורמציה מבנה המידע שלנו יהיה ניתן להפרדה לינארית. לשם כך נתעמק ב2 סוגי kernel.

מאחר וראינו במשוואה (14) כי התלות היא אך ורק במכפלה בין זוג דגימות, ניתן להפעיל עליהם טרנספורמציה k כך:

כעת נגדיר את k לכל מיני סוגים כך למשל לדוגמא שבאיור 10 ניתן להגדיר:

ונקבל את הפרדה שתאפשר חציצה באמצעות plane לינארי.

**Polynomial kernel -**

r – constant

d- degree of freedom.

בכל אופן בחירת קרנל גורמת ל ״סידור״ שונה של מבנה המידע.

## **5.2.מודל** Random Forest

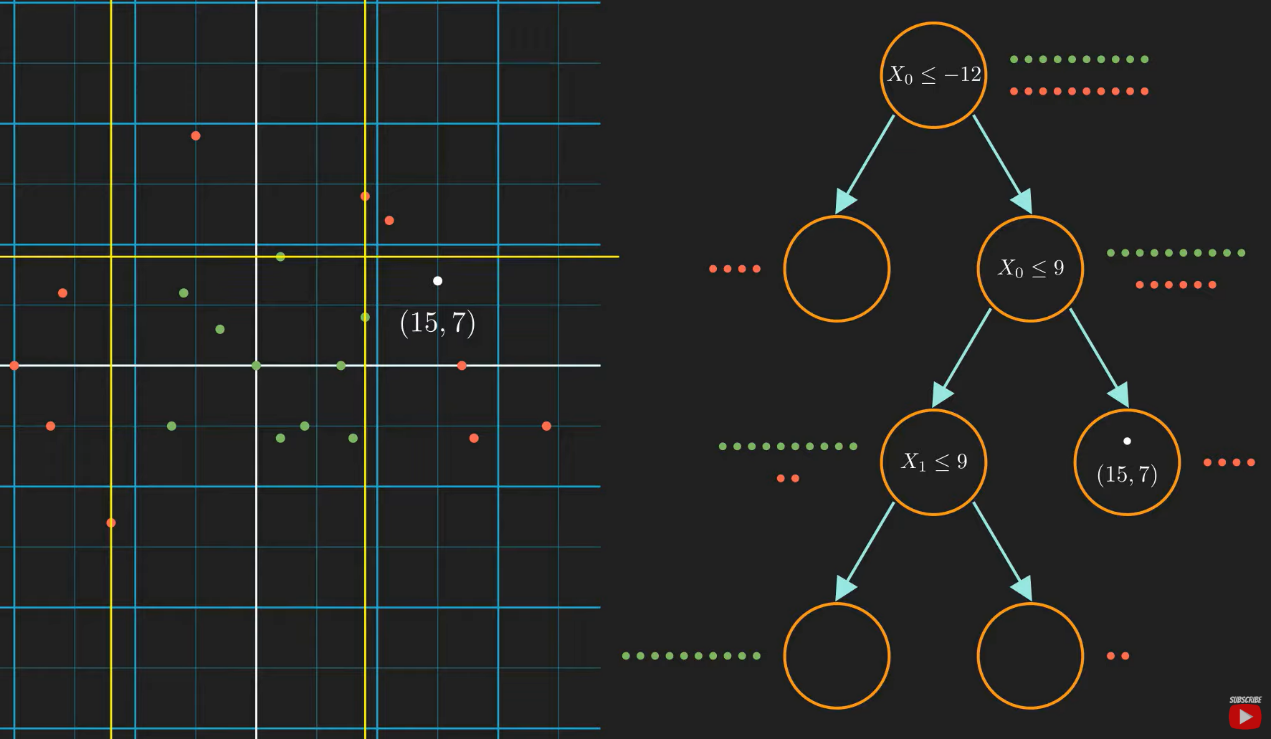
על מנת להבין את אלגוריתם Random Forest יש ראשית להבין מהו אלגוריתם עץ החלטה – Decision Tree שהוא אבן יסוד באלגוריתם Random Forest .

### Decision Tree 5.2.1

אלגוריתם עץ החלטה הינו אלגוריתם לסיווג וחיזוי מסוג supervised כאשר בבסיסו הוא יכול לבצע הן משימות בינאריות והן רגרסיות. בעבודה זו התעמקנו בעץ החלטה בינארי. עץ החלטה הינו תהליך שלב אחר שלב לביצוע חיזוי. בכל שלב נלקחת אחת התכונות- אחד ה features ודרכו מפצלים את הצומת ל2 ענפים, אשר צד אחד הוא תשובת כן על השאלה וצד שני תשובת לא . כך ממשיכים ובכל שלב שיש צורך להמשיך לפצל ממשיכים ושואלים על תכונה אחרת עד שמגיעים למצב שבכל "עלה" יש אך ורק נקודות מאותו ה class במצב כזה לא ממשיכים לפצל כי הגענו להחלטה שאכן כל הנקודות שייכות ל class מסוים. בבניית עץ, נשאלות שאלות כמו:

* לפי איזו תכונה לפצל את העץ?
* אם מפצלים לפי ערך נומרי אז מהו ערך הסף שקובע את הפיצול?
* מתי מפסיקים לפצל?

על שאלות אלה נענה בהמשך תת פרק זה.

נדגים בקצרה את אלגוריתם העץ, כאשר ציר הx הינו x0 וציר y הינו x1 ונרצה בהינתן נקודה חדשה להגיד האם היא מתאימה לאדום או לירוק:

איור 12- דוגמא לחלוקת הציר בהתאם לשאלות בעץ ההחלטה

באיור 12 ניתן לראות כי עבור כל שאלה אנחנו מפצלים את הדאטה שלנו עוד ועוד עד שלבסוף נותרים רק עלים עם אותו סוג של class .

כעת נענה השאלה כיצד נדע איך בדיוק לפצל את ה dataset שלנו, לשם כך נגדיר קריטריון טיב הנקרא Entropy – מדד לכמה המידע רנדומלי בתוך קבוצת מידע, אנטרופיה מקבלת ערכים בין 0 ל-1 כאשר אם הdata שלנו מחולק בדיוק לחצי מהכמות מתאים לclass 0 וחצי מהכמות מתאימה ל class 1 אזי האנטרופיה היא 1 (כלומר אנחנו במצב של "ניחוש" באופן מלא).

*נגדיר מדד הנקרא Information Gain שבאמצעותו ניתן להגיד האם החלוקה שבצענו תרמה לנו להמשך בניית העץ וכמה היא תרמה :*

*מדד זה מראה לנו את השיפור באנטרופיה בין ״עלה האבא״ לבין ״עלי הבן״, נזכור כי אנחנו רוצים מצב של אנטרופיה נמוכה ולכן ניתן לראות כי ככל שה IG גדול יותר משמע היה פיצול מוצלח. בבניית העץ בודקים את כל אפשרויות הפיצולים ולבסוף נבחרת האופציה עם ה IG הגדול ביותר.*

*מדד נוסף הקיים על מנת להגדיר האם חלוקה היא אפקטיבית או לא הוא מדד GINI המחשב את ההסתברות של feature ספציפי שמסווג לא נכון כאשר נבחר בצורה רנדומלית.*

*– ההסתברות של אלמנט להיות שייך לאחד הclasses*

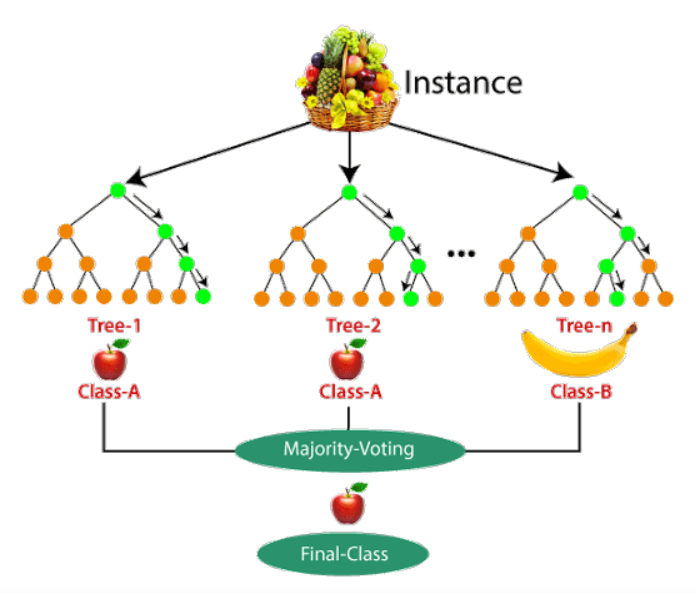
*כעת לאחר הפיצול שבוצע באמצעות ה IG אנחנו רוצים לבדוק האם יש עוד למה להמשיך לפצל את העץ, כאשר בעלה האנטרופיה שווה ל-0 אזי כל הנקודות מידע שהשתייכו לעלה הן מאותו class ושם נעצור את העץ. במידה והגענו לתנאי עצירה של max\_depth – פרמטר שנרצה להגדיר מראש, כמות הפיצולים תקבע לפיו כאשר אם אנחנו נמצאים בעלה שבו אין רק סוג אחד של נקודות מידע נבחר את השייכות לפי החלטת רוב.*

*אחת הבעיות הנפוצות בבניית עצי החלטה היא בעית ה overfitting ולכן לצורך פתרון בעיה זו אנחנו משתמשים בתנאי עצירה שונים ביניהם כפי שציינו max\_depth ובמודלים שכוללים שילוב של מספר פעמים את אותו מודל (ensemble) כאשר הנפוץ בהם עבור עצי החלטה הוא Random Forest עליו נפרט בתת פרק 5.2.2*

### Random Forest 5.2.2

*אלגוריתם זה באופן כמה עצי החלטה (כמות עצי ההחלטה הינו hyperparameter) אשר כל אחד מיוצר על ידי דגימה שונה של ה dataset ובוחר את החלטת הרוב של העצים במקרה הבינארי בו אנו עוסקים. בעצם, אלגוריתם זה הוא ממשפחת ה* ***ensemble learning*** *technics מסוג* ***bagging*** *כלומר אנחנו משתמשים בכמה מודלים בצורה מקבילית ולבסוף מבצעים החלטה בהתאם לרוב ההחלטות האינדיבידואליות של כל מודל. נגדיר את הצעדים הבסיסיים ביצירת random forest*

1. *מפצלים את הdata לכמה subset data ולכל אחד מייצרים decision tree*
2. *כל decision tree מייצר החלטה*
3. *ההחלטה הסופית נקבעת על פי רוב מההחלטות של ה decision tree*



איור 13 - דוגמא לתהליך של ה RF

***Hyperparameters:***

ל-Random Forest יש מספר היפר-פרמטרים שניתן לכוונן כדי לייעל את הביצועים שלו. ההיפר-פרמטרים החשובים ביותר של המסווג כוללים:

n\_estimators: מספר עצי ההחלטה ביער. הגדלת מספר העצים יכולה לשפר את ביצועי המודל אך גם מגדילה את העלות החישובית.

max\_features: המספר המרבי של תכונות שנחשבות לפיצול בכל צומת. צמצום מספר התכונות יכול להפחית overfitting אך עלול גם להוביל לחוסר התאמה.

max\_depth: העומק המרבי של כל עץ החלטות. הגדלת העומק המקסימלי יכולה לשפר את הביצועים של הדגם אך גם מגבירה את הסיכון ל overfitting.

min samples split: המספר המינימלי של דגימות הנדרש לפיצול צומת פנימי. הגדלת פרמטר זה יכולה למנוע מהמודל לפצל צמתים המכילים מעט מדי דוגמאות ויכולה להפחית overfitting.

min samples leaf: המספר המינימלי של דגימות הנדרש להיות בצומת עלה. הגדלת פרמטר זה יכולה למנוע מהדגם ליצור עלים עם מעט מדי דוגמאות ויכולה להפחית overfitting.

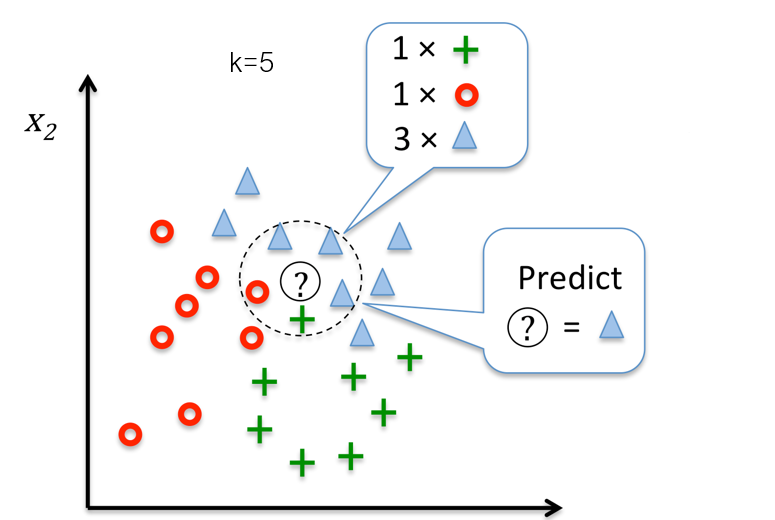
## 5.3.מודל KNN

אלגוריתם חסר פרמטרים - לא מתבצע תהליך של למידה לסיווג ולרגרסיה מקומית בשני המקרים הקלט תלוי ב-k התצפיות הקרובות במרחב התכונות.

K-NN לסיווג- בהינתן קלט של דוגמה חדשה, האלגוריתם משייכה לקבוצה הדוגמה משויכת למחלקה הנפוצה ביותר בקרב k השכנים הקרובים כאשר k מוגדר כמספר חיובי שלם, אם k=1 האובייקט משויך למחלקה של השכן הבודד הקרוב ביותר. כאשר חישוב המרחק מכל "השכנים" מתבצע ע"י חישוב מרחק אוקלידי. דוגמה לחישוב מרחק כאשר מדובר בבעיה דו ממדית :

נפרט את אלגוריתם המודל :

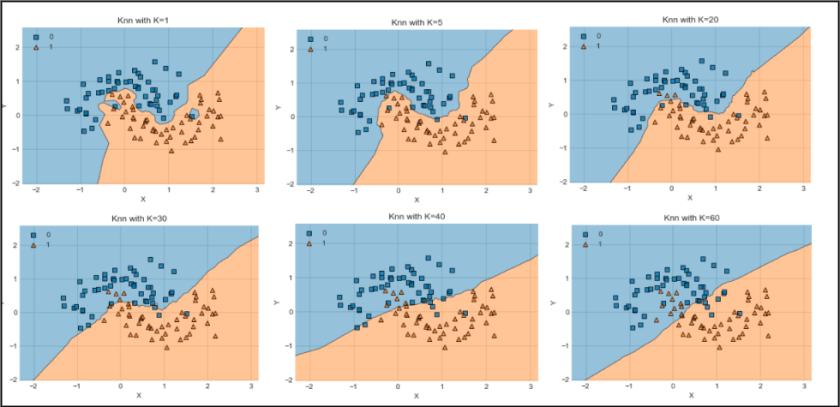
1. נייבא/ נייצר את ה train dataset
2. נבחר hyperparameter – K
3. נחשב את המרחק בין הנקודה שנרצה לסווג לכל הנקודות שנמצאות ב dataset
4. נסדר את תוצאות המרחק ברשימה ממוינת
5. נבחר את K את המרחקים הראשונים
6. נבצע בדיקה עבור K המרחקים לאיזה class שייכות הנקודות ולפי הרוב נסווג את הנקודה.



איור 14 - דוגמא לתהליך ה knn עבור k=5 ו classes=3

כפי שניתן לראות באיור (14) בהינתן נקודה חדשה (?) מחשבים את מרחקה למול כל שאר הנקודות שנמצאות במרחב ולבסוף לפי בחירת רוב מתוך k נסווג את הנקודה.

בחירת הפרמטר k נובעת מכמה שיקולים. עבור k קטן אנחנו מקבלים רגישות לרעשים – outliers ועבור k גדול מידי אנחנו יכולים לאבד הרבה נקודות. איור (15) ממחיש כיצד עבור k קטן יכול להיות התחשבות גדולה מידי ב outliers וסיווג נקודות בסביבת נקודה אחת ככחולות למרות שניתן לראות שהנקודה שגרמה להן להיות כחולה היא לא מייצגת את כל הנקודות, ועבור k גדול מידי ניתן לראות שהגבולות נוקשים מידי ואנחנו מסווגים הרבה מהכחולים ככתומים.



איור 15 - השפעת k על מרחב הסיווג

נקודה חשובה שיש לציין לגבי אלגוריתם זה הוא כמות החישוביות, וכמות הזיכרון שהוא תופס.

כאשר סיבוכיות האימון עבור n נקודות עומד על O(n)

ניתן לבחור את k בהתאם לכל מיני קריטריוני טיב אשר ככל הנראה לא כולם יתנו את אותה התוצאה. בפרויקט זה, נבחר k באחת הפעמים:

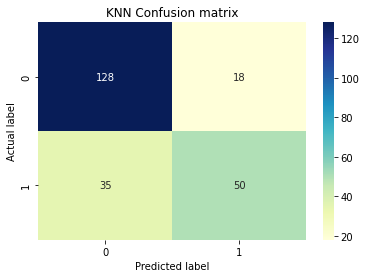
# 6.ניסויים

בשלב זה, בצענו ניסויים על כל אחד מהמסווגים כאשר בתחילה בצענו ניסוי עם ערכי default של הפרמטרים ואז בצענו אופטימיזציה. בפרק זה נבחן את עבודת המודלים ונשווה בין ביצועיהם עבור מבנה המידע המוצג בפרק 3. כאשר לכל אחד מהמסווגים נבנה מטריצת בלבול, גרף ROC ומדדי recall, accuracy .

## 6.1.מודל KNN

עבור תת פרק זה נציג את התוצאות הרלוונטיות עבור המודל knn לפי קריטריון הטיב שהוגדר בהצעת הפרויקט ובתת פרק 4.3 נרצה להשוות את ביצועי המודל כאשר הוא נלקח עם אופטימיזצית פרמטרים וללא, לשם כך נציג פעמיים את תוצאות המודל ונראה כיצד בחירת פרמטרים מתאימים משפיעה על התוצאה.

מטריצת בלבול עבור מודל ללא חיפוש k אופטימלי (ערך הdefault המוגדר בdatasheet של הפונקציה הוא 5 ):



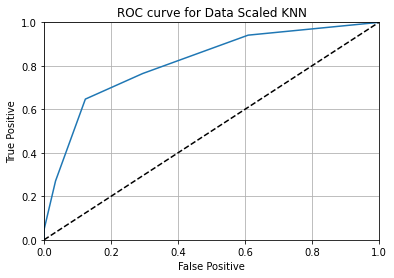
איור 16 – confusion matrix עבור knn ללא בחירת k אופטימלי

**F** **score** = 0.696

**Accuracy** = 0.79

**Recall** = 0.64

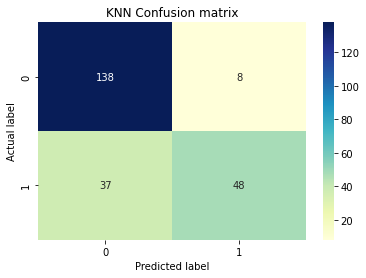
**AUC** =



איור 17 - גרף הroc עבור k ברירת מחדל

כעת עבור חיפוש אחר פרמטר accuracy סרקנו אחר k מ1 ועד 70 כאשר שמרנו בכל פעם את ציון ה accuracy ולבסוף שמרנו את המודל עם התוצאה הטובה ביותר.

מטריצת הבלבול ותוצאות עבור מודל מאומן בדגש על accuracy :



איור 18- confusion matrix עבור k אופטימלי ל accuracy

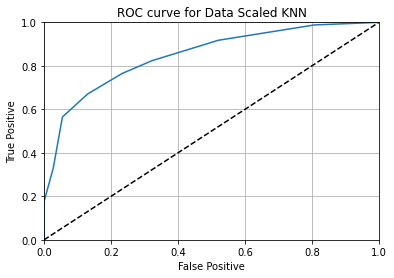
**F** **score** = 0.68

**Accuracy** = 0.805

**Recall** = 0.564

**AUC** =

כעת נציג את גרף ה ROC :



איור 19 - גרף הroc עבור k אופטימלי

## 6.2.מודל SVM

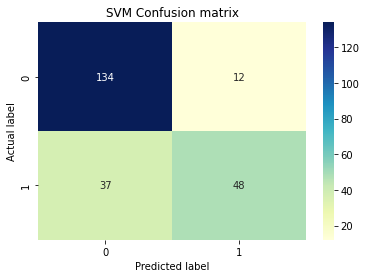
מאחר ומסווג זה בעל סוגים שונים כאשר נבצע GridSearchCV על מנת למצוא את הפרמטרים הטובים ביותר, שיטה זו דורשת להגדיר את כל הפרמטרים אותם נרצה לבדוק לאחר מכן מבצעת מתוכם את כל הקומבינציות האפשריות כך שלכל קומבינציה נוצר מודל, לאחר מכן מתבצעת בדיקה באמצעות cross validation במידה ואנחנו רוצים בדיקה ספציפית ניתן להגדירה מראש למשל אם נרצה לשמור את המודל עם הfscore הגבוה ביותר, וכך הפונקציה לבסוף תחזיר לנו את הפרמטרים שהניבו את המודל שקיבל את הscore הגבוה ביותר.

­

בפרויקט זה בדקנו את כל הקומבינציות בין הפרמטרים הבאים :

**parameters** ={ 'C':[**0.1**,100,1000],'kernel':['rbf',**'poly'**,'sigmoid','linear'],'degree':[**1**,2,3,4,5,6]}

הפרמטרים הסופיים שנבחרו מסומנים.



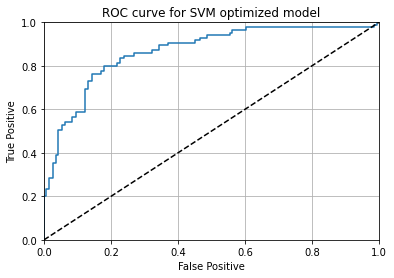
איור 20 - confusion matrix עבור מודל svm אופטימלי

**F** **score** = 0.662

**Accuracy** = 0.787

**Recall** = 0.80

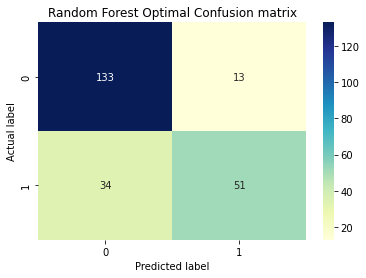
**AUC** =



איור 22- גרף הroc עבור svm אופטימלי

## 6.3.מודל Random Forest

גם עבור מודל זה בוצעה התאמה למציאת הפרמטרים הטובים ביותר. במסווג זה בוצע החיפוש באמצעות RandomizedSearchCV כאשר גם כאן הגדרנו כיצד את מרחב החיפוש כלומר את אופציות ה hyperparameters שאותם נרצה לבדוק ההבדל בין פונקציית חיפוש זו לבין GridSearchCV הוא שכאן לא כל הקומבינציות שמוצגות ב parameters מוגדרות אלא נלקחים באופן רנדומלי קומבינציות שונות בהתאם לכמות האיטרציות שנקבעה.



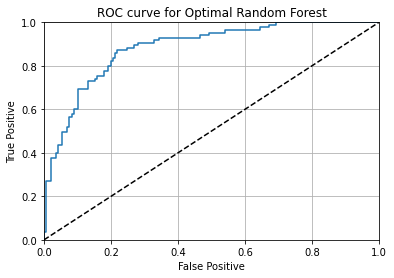
איור 21 - confusion matrix עבור מודל random forest אופטימלי

**F** **score** = 0.684

**Accuracy** = 0.796

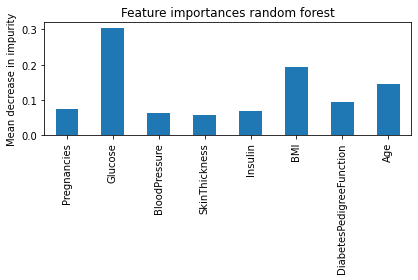
**Recall** = 0.6

**AUC** =



איור 23 - גרף הroc עבור random forest אופטימלי

עבור מודל זה בדקנו אילו תכונות משפיעות הכי הרבה על יצירת העצים, אנחנו מצפים לקבל קורלציה בין המאפיינים המשפיעים ביותר לבין אלו עם הקורלציה הגבוהה ביותר לעמודת התוצאה- בהתאם לאיור (4)



איור 24- חשיבות כל feature ביצירת העץ

מאיור 24 ניתן לראות כי אכן המאפיינים המשפיעים ביותר הם glucose,bmi כמצופה כמו שראינו באיור (4).

# 7.סיכום ומסקנות

בפרויקט זה, נבדקה יכולת של שלושה מסווגים שונים – SVM, KNN, Random Forest לסיווג סוכרת על מערך הנתונים של “Pima Indians Diabetes”. המטרה הייתה לבדוק את יכולת המסווגים לסיווג סוכרת.

בשלב הראשון של הפרויקט, נעשה עיבוד וטיפול בנתונים, כך שנוכל להשתמש בהם למטרות האנליזה. עיבוד הנתונים כולל קביעת seed אשר מגדיר כי בכל פעם שנריץ את התוכנית המידע יפוצל בדיוק באותו סדר, בדיקה וניקוי של מבנה המידע כך שכל הערכים הם מספריים ואין מקום שבו אין ערך, זיהוי outliers וביצוע נירמול ופיצול מבנה המידע לסט אימון ולסט בדיקה.

לאחר מכן, בשלב הניסוי, ביצענו ניסויים על כל אחד מהמסווגים כאשר בהתחלה ביצענו ניסוי עם ערכי ה default של הפרמטרים ואז בצענו אופטימיזציה. נבחנו עבודת המודלים וביצענו השוואה בין ביצועיהם לפי מטריצת בלבול, גרף ROC, ומדדי Recall, Accuracy.

מסקנות שעלו מפרויקט זה:

* רמת הגלוקוז, BMI וגיל זוהו כמאפיינים החשובים ביותר לסיווג סוכרת בהתבסס על ניתוח חשיבות התכונה באמצעות Random Forest.
* שלבי עיבוד מקדים, כגון טיפול בערכים חסרים ושינוי קנה מידה של התכונות, היו חיוניים לשיפור הביצועים של המסווגים.
* הביצועים הכוללים של המסווגים היו טובים, עם דיוק שנע בין 77% ל-81%. זה מצביע על כך שלמידת מכונה יכולה להיות כלי שימושי לסיווג סוכרת ועשויה לסייע באבחון ובטיפול במחלה.
* חשיבות גודל ואיכות מערך הנתונים - מערך הנתונים של Pima Indians המשמש בפרויקט זה הוא מערך נתונים קטן יחסית עם 768 דוגמאות בלבד, מה שיכול להגביל את היעילות של אלגוריתמי הסיווג.
* לבחירת ה hyperparameter יכולה להיות השפעה משמעותית על הביצועים של המסווגים.
* שימוש בסט נתונים מתאים ומגוון שיכול לתת מגוון רחב של תוצאות לאלגוריתמי הסיווג – חשוב לבדוק את הנתונים כדי לוודא שאין בעיות בנתונים עצמם כמו נתונים חסרים או שגיאות.
* בכדי להפוך את המודל לביצועי יותר ניתן להשתמש בטכניקות כמו אופטימזציה של פרמטרים והיפר-פרמטרים כדי לשפר את ביצועי המודל ולהגדיל את דיוקו.
* חקירת פיצ'רים נוספים לנתונים על מנת להבין טוב יותר את המאפיינים המשפיעים על הסיווג ולאפשר שימוש בפיצ'רים חדשים לסיווג מדויק יותר, לדוגמא: רמת הפעילות הגופנית- ישנם מחקרים שהראו שרמת הפעילות הגופנית משפיעה על הסיכוי לחלות בסוכרת. ניתן להוסיף את הפיצ'ר הזה ולבדוק האם הוא משפיע על הביצועים של המסווגים.