**חיזוי תוצאות משחקים בליגה האנגלית לשנת 2019/20 על סמך שלוש העונות האחרונות**

**פרויקט בבינה מלאכותית 236502**

**הפקולטה למדעי המחשב , טכניון**

מגישים:

מוסא שאער

315651893

[Mousa.shaer@campus.technion.ac.il](mailto:Mousa.shaer@campus.technion.ac.il)

מוחמד איוב

318548666

[Ayoub@campus.technion.ac.il](mailto:Ayoub@campus.technion.ac.il)

**מבוא**:

ספורט היה חלק מחיי האדם במשך אלפי שנים, והעניין להיות יותר מסתם קהל פשוט מחזיר אותנו לפני יותר מ -2000 שנה, אז היוונים הלכו לקולוסיאום כדי להמר על גלדיאטור שבחרו. מאז אותה תקופה גבר העניין לנסות להכיר את העתיד במשחקי הספורט.

לדוגמא, פרשנים בתכניות טלוויזיה מנחשים איזו קבוצה תזכה בסופר בול או מנסים להבין את התוצאה של משחק בליגת העל, אנשים בתקשורת דנים מי יזכה באליפות העולם בטניס. זה כבר חלק מחיינו.

בנוסף, ענף ההימורים הספורטיביים גדל עד כדי כך שבשנת 2019, סך ההכנסות הגולמיות של ענף ההימורים הסתכם ב -85 מיליארד דולר [1]. העניין לדעת את התוצאה של משחקי ספורט לפני סיומם ברור כעת וחיוני לתעשיית הימורי הספורט.

מכיוון שלכדור רגל יש מספר גדול מאוד של מאפיינים שקשורים ישירות לתוצאה, קשה לבני אדם להתחשב בכל התכונות ולחזות בדיוק רב את תוצאות המשחק. במצבים אלה, יש צורך בטכניקה בעלת ביצועים גבוהים בכדי להתמודד עם כל הנתונים, וכאן נכנסת הבינה המלאכותית. ההתקדמות העצומה בטכנולוגיה זו מאפשרת לעבד כמות מופרזת של נתונים כדי להסיק מסקנות שימושיות ביותר.

המטרה העיקרית של פרויקט זה היא לחקור טכניקות למידת שונות על מנת לחזות את התוצאה של משחקי כדורגל בליגת האנגלית (English Premier League), באמצעות אירועי משחק ודירוגי השחקנים לפי אתר הFIFA .

**הגדרת הבעיה:**

בפרויקט הזה בחרנו להתמקד בחיזוי תוצאות המשחקים בליגה האנגלית לעונה 2019\20. כלומר בהינתן שתי קבוצות (קבוצת בית וקבוצת חוץ) המודל שלנו מנסה לחזות את התוצאה של המשחק ומחזיר אחת התוצאות הבאות:

H- הקבוצה הביתית ניצחה.

A- קבוצת החוץ ניצחה.

D- המשחק ניגמר בתיקו.

**תיאור הפתרון המוצע לבעיה:**

במסגרת פרויקט זה החלטנו לנסות להתמודד עם בעיה זו כבעיית למידה בה על סמך משחקים קודמים ונתונים קיימים ננסה לחזות את התכונות והסטטיסטיקה של המשחק ואז על סמך התכונות האלה ננסה לחזות את התוצאה של המשחק.

הרעיון שעמד מאחורי החלטה זו היא שרואים שבתוכניות טלוויזיה מנסים לחזות מה הולך לקראת במשחק על פי הנתונים שיש להם ומניסיון שלהם בניתוח משחקים קודמים ולכן יתכן שגם אלגוריתם למידה ידע לחזות איך ילך המשחק ואז יוכל לחזות את התוצאה.

* **איסוף מידע:**

החלק המאתגר ביותר בפרויקט הוא איסוף המידע.

חיפשנו בכל מני אתרים על מידע שיהיה עדכוני ואמין על מנת לקבל את התוצאות הכי טובות.

בסיס הנתונים שמצאנו ראשון היה מכיל סטטיסטיקה על משחקים משנת 2016 עד 2020 המידע היה מכיל תאריך המשחק, קבוצת בית וקבוצת חוץ,

בנוסף עבור כל קבוצה היה מכיל את מספר השערים, הקרנות, הבעיטות למסגרת ולחוץ המסגרת, כרטיסים צהובים ואדומים.

הנתונים האלו לא היו טובים מספיק בשבילנו כי הם מייצגים נתונים של משחק שכבר נערך ואנחנו היינו רוצים לחזות את התוצאות על סמך נתונים אחרים שקרו לפני המשחק, כמו למשל:

1. ההרכב של כל קבוצה, ידוע שכל קבוצה מכריזה על ההרכב שהולך לפתוח את המשחק שעה קודם מתחילת המשחק.
2. סטטיסטיקה על השחקנים של כל קבוצה, ופה בחרנו להסתמך על הנתונים של אתר הFIFA- כי הם מאוד משקפים את היכולת של השחקנים.

ופה היינו צריכים להשתמש בשיטת web scrapping על מנת להוציא מידע מאתרי אינטרנט, כי לא מצאנו אתר שמכיל את הנתונים שאנחנו צריכים ולכן כתבנו קוד שקורא את הנתונים שאנחנו צריכים מאתרי אינטרנט מסוימים ואז מסדר לנו את המידע בקבצי אקסל שנוח לעבור איתם.

החלק הזה היה קצת מורכב כי זה פעם ראשונה שלנו שהיינו משתמשים בשיטתweb scrapping והיינו צריכים ללמוד את זה בעצמנו, וכל זה היה אחרי שבזבזנו הרבה זמן לחפש אתרים שמכילים את המידע שמחפשים ולבדוק שאכן המידע נכון.

לכן הצלחנו בשיטה הזו למצוא עוד מידע עדכוני ואמין שמכיל:

ההרכב שכל קבוצה פתחה בו את המשחק, את השיטה שמשחקת בה כל קבוצה (כמה שחקנים בחלק הקדמי, באמצע המגרש ובהגנה) בנוסף הצלחנו למצוא את דירוגי ה FIFA של כל השחקנים כמו דירוג הכללי שלו, דירוג המהירות והיכולת הפיזית, דירוג של יכולת ההגנה וההתקפה ועוד.

אחרי שמצאנו את הנתונים האלו יצרנו תכונות חדשות שיעזרו לנו לחזות את הסטטיסטיקה בתוך המשחק עצמו. כמו למשל התכונות שמצאנו במאגר הנתונים הראשון (מספר בעיטת למסגרת, קרנות, כרטיסים אדומים וצהובים ועוד...), ויצרנו את התכונות הבאות:

1. רצינו קודם לקבל מידע על התקופה האחרונה של כל קבוצה ומה המצב שלה בליגה לכן ממאגר הנתונים הראשון שלנו עבור כל משחק בדקנו מה היה המצב של שתי הקבוצות בשלושת המשחקים האחרונים.

ולכן לקחנו ממוצעים של הסטטיסטיקות של כל קבוצה בשלושת המשחקים האחרונים כמו למשל ממוצע השערים שכבשה , ממוצע השערים שספגה, ממוצע הקרנות, ממוצע הכרטיסים הצהובים והאדומים ומספר הנקודות שכל קבוצה הצליחה להשיג.

1. בנוסף רצינו להוסיף תכונות על סמך העבר של המפגשים הישירים בין שתי הקבוצות ולכן יצרנו אותם תכונות אבל עכשיו עבור שתי המשחקים האחרונים הישירים בין שתי הקבוצות.
2. הוספנו עוד תכונות שתלויה בהרכב שכל קבוצה פתחה בו למשל:  
   * ממוצע המהירות של החלק הקדמי (שישקף את הכוח של החלק ההתקפי של הקבוצה).
   * את הממוצע של הכוח הפיזי של הקשרים והמגנים שיכול לשקף את איכות ההגנה של הקבוצה.
   * ההפרש בין ממוצע דירוג החלק ההתקפי לממוצע דירוג החלק האחורי.

בסיום השלב הזה קיבלנו את התכונות שיכולות לעזור לנו לחזות את הסטטיסטיקה של שתי הקבוצות בתוך המשחק, ולכן בנינו רשת נירונית שמקבלת ויקטור של התכונות שיצרנו ומחזירה ויקטור תכונות המייצג את החיזוי שמה הולך לקרות במשחק עצמו, נפרט על זה בהמשך.

לסיכום קיבלנו מידע אמין עבור שלושת העונות האחרונות כלומר עבור 1080 משחקים.

* **עיבוד מקדים של הדוגמאות:**

השלב הזה גם היה מורכב מכמה סיבות כאשר העיקריות בהן היו:

1. אספנו מידע ממקורות שונים לכן קיבלנו מידע שונה, למשל שמות הקבוצות והשחקנים הייתה שונה במסדי נתונים שונים.

את ההבדל בין שמות הקבוצות הצלחנו לסדר דרך פונקציה שלוקחת את שמות הקבוצות באחד הקבצים ועוברת על שאר הקבצים ומחליפה את השמות לפי מלון שנתנו לה, היה סך הכל 26 קבוצות שונות לכן היה ניתן למצוא mapping לשמות הקבוצות בקבצים השונים.

את ההבדל בין שמות השחקנים בקבצים השונים היה יותר מאתגר כי בכל קבוצה יש יותר מ 20 שחקנים ויש 20 קבוצות לכן היה מעל 400 שחקנים בכל עונה ולא היה אפשרי למצוא mapping לכן לא החלפנו את השמות של השחקנים בין הקבצים השונים אלא בנינו פונקציה שתעזור לנו במציאת השחקנים והיא עובדת באופן הבא:  
קודם כל לכל שחקן במאגר המידע שלנו יש גם הקבוצה שהוא משחק בה ולכן יכלנו לצמצם את החיפוש מ מעל 400 שחקנים לתוך 20-30 שחקן ואז על יד פונקציית קירוב שמחזירה את ההתאמה הקרובה ביותר הצלחנו להחזיר את השחקן המבוקש, למשל עבור השחקן מסעוד אוזיל שמשחק בארסינל באחד הקבצים הוא כתוב בצורה הזו M. Özil ובקובץ אחר הוא כתוב כך Mesut Özil אז הפונקציה מחפשת בשחקני ארסינל את שם השחקן הכי קרוב על ידי השוואת מחרוזות ומחזירה אותו. בנוסף עשינו ניסויים וודינו שהפונקציה עובדת כישר.

1. גם התאריכים היו שונים בקבצים שונים כלומר כל אחד היה בפורמט שונה לכן גם עשינו פונקציה שמאחדת את הפורמט בקבצים השונים.
2. עוד בעיה שפגשנו הייתה חלון ההעברות, בתקופה מסוימת במהלך העונה יהיה אפשרות לשחקנם לעבור מקבוצה אחת לאחרת ואז יהיה לנו בעיה למצוא את השחקנים. ולכן החלטנו לאסוף את המידע עבור השחקנים לפני ואחרי חלון ההעברות ואז לחפש את השחקן

* **למידה:**

שלב הלמידה השתמשנו באלגוריתמי למידה שונים על מנת לבנות מסווג שבהינתן משחק לא מסווג ידע לחזות את התוצאה של המשחק.

את האלגוריתמים האלה אימנו בשיטת cross validation כאשר חלקנו את מאגר הדוגמאות ל 5 קבוצות ובכל איטרציה האלגוריתם בוחר קבוצה מתוך חמש הקבוצות להיות קבוצת המבחן והשאר משמש לאימון, ואז מחזיר את ממוצע הדיוק עבור 5 האיטרציות.

חשוב לציין שבשלב הזה הרצנו את המסווגים עם פרמטרים שונים על מנת לקבל דיוק הטוב ביתר נפרט על זה בהמשך.

* **דילול המאפיינים:**

לאחר שלב עיבוד המידע הוספת וחיזוי תכונות ובחירת איך לאמן את המסווגים החלטנו ללמוד את התכונות הטובות ביותר כי חלק מהתכונות לא רלוונטיות ויכולות לגרום ל- overfitting ובנוסף תכונות רבות יגרמו להארכת זמן בניית המסווג.

על מנת לדלל את התכונות ביצענו features selection על ידי אלגוריתם select k best אשר בוחר אתK המאפיינים המשמעותיים ביותר עבור הסיווג בהסתמך על score function כלשהי אותה נרצה לבחור כך שתיתן ביצועים טובים על הנתונים שלנו. (נפרט על זה בהמשך)

* **יצירת רגרסור סופי :**

מבין כל אלגוריתמי הלמידה שהשתמשנו בהם נבחר את האלגוריתם שמחזיר את המסווג שנותן את הדיוק הטוב ביותר עם הפרמטרים שלו והמאפיינים הסופיים שבחרנו.

המסווג הסופי יהיה מוכן להפעלה על דוגמאות לא מסווגות, בהיתן שתי קבוצות המסווג מחזה את התוצאה של המשחק. (H, A, D)

* **תיאור כללי של דרך הפתרון :**

איסוף מידע מאתרי ספורט

בשיטת web scrapping

רשת נירונית

המידע אחרי עיבוד ויצירת תכונות חדשות על סמך משחקים קודמים

איחוד התכונות שהיו לנו עם התכונות שיצרה הרשת נירונית

איסוף מידע מאתרי מסדי נתונים

התכונות אחרי דילול

הערכת מסווג

מודל

אימון מודל בעזרת אלגוריתמי למידה

**אימון**

--------------------------------------------------------------------------------------------

**חיזוי**

Home Team vs

Away Team

תוצאה

רגרסור סופי

Feature Extraction

**תיאור המערכת**:

המערכת אותה בנינו לשם יישום דרך הפתרון מורכבת ממספר קבצי פיתון בנוסף לקבצי הנתונים, חלק אחד של קבצי מבצע עיבוד נתונים והחלק השני בוחן מסווגים שונים ובוחר מתוכן את המסווג האופטימלי שמשמש כפתרון לבעיה.

הקבצים כתובים ב python 3 ומשתמשים בהם בעיקר בספריית pandas לשם עיבוד הנתונים ו sklearn ללמידה.

**חלקי המערכת :**

* **שלב העיבוד המקדים של המידע:**

אחרי חילוץ הנתונים שצריכים מאתרי הספורט ויצירת מאגר הנתונים שעליו יתבססו האלגוריתמים השונים בבניית המסווגים, קיבלנו מספר קבצים שאחד מכיל את התוצאות של המשחקים וסטטיסטיקות של המשחק עצמו ומספר קבצים נקראים Lineups17-20.xlsx אשר מכילים בתוכם את ההרכב של שתי הקבוצות ועוד מספר קבצים נקראים Players17-20.xlsx שמכילים סטטיסטיקות על השחקנים.

כל הקבצים האלה עוברים עיבוד על מנת לקבל תאריך בפורמט אחיד בכל הקבצים ובנוסף על מנת לתאם את השמות של הקבוצות והשחקנים בקבצים השונים בשביל כך כתבנו קובץ פיתון הנקרא features\_preprocessing.py אשר עובר על הקבצים האלה ומייצר קבצים חדשים באותו שם אבל בתיקייה שונה השומרים על מבנה אחיד וניתן לעבוד איתם.

בנוסף הקובץ הזה עובר על כל המשחקים מייצר את התכונות החדשות על ידי סריקת שלושת המשחקים הקודמים של שתי הקבוצות והמשחקים הישרים ביניהם, לאחר מכן מחלץ את ההרכב של כל קבוצה על ידי חיפוש המשחק בקובץ המתאים (באחד קבצי ה Lineups) ואז עבור כל שחקן מחלץ את הסטטיסטיקה שלו מהקובץ המתאים (אחד קבצי ה Players) ומייצר את התכונות החדשות, וכך נקבל עבור כל משחק ויקטור של תכונות המוכן לכנס לרשת ה-נירונית על מנת לחזות עוד תכונות.

הווקטורים האלו נכנסים אל תוך הרשת ה-נירונית ומשם יוצאים את התכונות החדשות שמאחדים אותם עם התכונות לפני הכניסה לרשת וכך נקבל את התכונות שיכנסו לשלב הלימוד, תכונות אלו נמצאות בקובץ הנקרא selected\_features.xlsx.

* **שלב הלמידה והניסויים:**

בשלב הזה השתמשנו במספר אלגוריתמי למידה:

KNN, GradientBoosting, Decision Tree, Random Forest, SVM, Naïve Bayes  
כולם על ידי חבילת sklearn.  
כתבנו עוד קובץ בשם learning אשר מבצע את הדברים הבאים:

1. טוען את הקובץ selected\_features.xlsx
2. מחלק את הקובץ לשתי קבוצות training and validation לפי אלגוריתם cross validation הנמצא בספריית sklearn עם פרמטר 5 כלומר מחלק את הקובץ ל 5 חלקים אשר 4 משמשים ל training ואחת ל validation.
3. עבור כל אלגוריתם למידה מריץ את האלגוריתם עם פרמטרים שונים אשר הגדרנו כמלון ועבור כל אחד שומר את התוצאות בקובץ אקסל ממוינים בסדר יורד לפי הדיוק של האלגוריתם.
4. עבור כל אלגוריתם בוחרים את הפרמטרים שקיבלנו עבורם את התוצאות הטובות ביותר ומריצים עליהם select best K עבור ערכי K שונים הנעים מ 15-30 ובוחרים את האלגוריתם והערך K אשר החזירו את הדיוק הטוב ביותר.

* **המסווג הסופי:**

המודל הסופי נבחר לפי הניסויים שעשינו בשלב הלמידה, כאשר בוחרים את אלגוריתם הלמידה עם הפרמטרים וערך K אשר החזירו את התוצאה הטובות ביותר.

* **הקובץ אשר יריץ המשתמש:**

בקובץ הזה שמרנו את המסווג הסופי שבחרנו בשלב קודם.

הקובץ הזה טוען את הקובץ selected\_features.xlsx ומאמן את המודל שלנו עליו.ואז המשתמש יכול להכניס כקלט את המחשק שירצה לחזות בצורה הבאה:

צריך להוסיף את זה + דוגמאות הרצה

**מתודולוגיה ניסויית:**

המטרה שלנו בפרויקט זה היא למצוא אלגוריתם למדיה בעל דיוק גבוה לבעיה שהצגנו.

חלק הניסויים התחלק ל 3 שלבים:

השלב הראשון היה ללמוד לחזות את התכונות עצמם על ידי רשת נירונית (להסביר קצת על הרשת).

בשלב השניבצענו ניסויים שמטרתם למצוא את הפרמטרים המוצלחים עבור כל אלגוריתם למידה, ובחלק האחרון בצענו ניסויים שמטרתם מציאת המספר האידיאלי של מאפיינים בהם ישתמש האלגוריתם.

בחלק זה נציג את אלגוריתמי הלמידה שבחרנו, נסביר בקצרה את אופן הפעולה שלהם ונציין את הפרמטרים שבחנו. ובסוף נציג את תוצאות הניסויים ואת המסקנות מהם.

**הצגת אלגוריתמי הלמידה:**

* **אלגוריתמים חיזוי התכונות:**

(להסביר קצת על הרשת).

* **אלגוריתמים שלב הלמידה:**

החלטנו להשתמש ב 6 אלגוריתמי למידה שונים:

**Random Forest:**

**A close up of a map

Description automatically generated** הוא אלגוריתם למידה המורכב מודעת עצי החלטה.

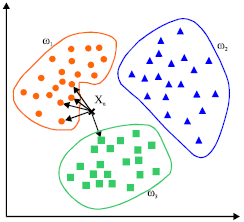
ועדה היא אוסף של עצים בה כל עץ מחזיר ערך משלו, הערך הסופי המוחזר הוא פונקציה המחושבת על כלל הערכים המוחזרים מעצי הועדה.  
  
וועדת היער האקראי בנויה כך שכל עץ מאומן על תת קבוצה שונה של דוגמאות ובכל פיצול של צומת פנימי בעץ נבחרת תת קבוצה של תכונות באופן אקראי, הפונקציה המשמשת לחישוב הערך המוחזר היא ממוצע הערכים שהחזירו שאר העצים. יתרון מרכזי של גישה זו הוא הפחתת תופעת ה over-fitting.

היער האקראי בו נשתמש הוא היער האקראי הממומש ע״י ספריית scikit-learn.

הפרמטרים שבחרנו לכוונן עבור אלגוריתם זה הם:

* + n\_estimators: מספר העצים בוועדה.
  + max\_depth: עומק העץ המקסימלי המותר.
  + min\_samples\_split: מספר הדוגמאות המינימלי הנדרש על מנת לפצל צומת פנימי בעץ

**KNN – K nearest neighbors:**

****

אלגוריתם זה מבוסס על ההנחה שלעצמים שקרובים זה לזה במרחב יהיה סיווג דומה.

שלב הלמידה כולל שמירה של הדוגמאות המתויגות והסיווג שלהן, ובהינתן דוגמא חדשה האלגוריתם מחשב את המרחק בין הדוגמא החדשה לבין כל אחת מהדוגמאות הידועות לו, ונותן לדוגמא החדשה את הסיווג של רוב הדוגמאות מבין הK הקרובות אליו ביותר.

במסווג זה בחנתי מספר פרמטרים:

* + n\_neighbors: מספר הדוגמאות שאיתן "יתייעץ" המסווג ועל ידי ההצבעה שלהן יינתן סיווג של דוגמא חדשה.

**Decision tree:**

עץ החלטה הוא מודל חיזוי המייצר עץ שממפה תצפיות לערכים המתאימים עבורן, העץ בנוי מצמתי החלטה שבכל אחד מהם נבדק תנאי מסוים על מאפיין מסוים של התצפיות ועלים המכילים את הערך החזוי עבור התצפית המתאימה למסלול שמוביל אליהם בעץ.

בהינתן עץ החלטה, ההערכה ניתנת ע״י בדיקת התנאי בכל צומת אליו מגיעים ומעבר לצומת המייצג את התשובה החל מהשורש, הערך המוחזר הוא הערך השייך לעלה בסוף המסלול.

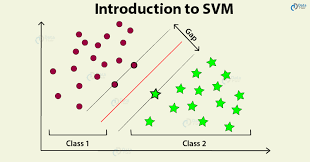
בפרויקט זה כל צומת פנימי מייצג מאפיין, התשובות האפשריות הן הערכים השונים עבור המאפיין, והעלים מייצגים סיווג טריניארי האם המשחק נגמר בניצחון ביתי, חוץ או תיקו.

עץ החלטה בו נשתמש הוא עץ ההחלטה הממומש ע״י ספריית scikit-learn.

הפרמטרים שבחרנו לכוונן הם:

* + Max depth: עומק העץ המקסימלי המותר.

:𝐒𝐕𝐌 (𝐒𝐮𝐩𝐩𝐨𝐫𝐭 𝐕𝐞𝐜𝐭𝐨𝐫 𝐌𝐚𝐜𝐡𝐢𝐧𝐞)

אלגוריתם זה מנסה לסווג את הדוגמאות על ידי מפריד ליניארי, תוך ניסיון למקסם את ה"שוליים"- המרחק בין המפריד הלינארי לדוגמאות.

המפריד שיבחר בסוף פעולת האלגוריתם הוא זה שמפריד בין הדוגמאות בצורה ה"חזקה ביותר", כלומר זה שהשוליים בו הם הרחבים ביותר.

הפרמטרים שבחרנו לכוונן הם:

* + C: לאחר ההפרדה יתכן מצב שבו ישנן דוגמאות אשר נמצאות באזור סיווג של דוגמאות המסווגות שונה מהן. פרמטר זה מגדיר מה יהיה ה"קנס" עבור דוגמאות אלו, והוא בעצם מבטא את ההעדפה שלנו באיזון בין שגיאות אלו לבין פונקציה פשוטה.
  + Kernel: פרמטר זה מגדיר את סוג ההפרדה בו האלגוריתם ישתמש.
  + Degree: דרגת הפולינום ('פולי'). מתעלמים מכל הגרעינים האחרים.
  + Probability: האם לאפשר הערכות הסתברות.
  + Gamma: מקדם גרעין עבור 'rbf', 'Poly' ו'sigmond '.

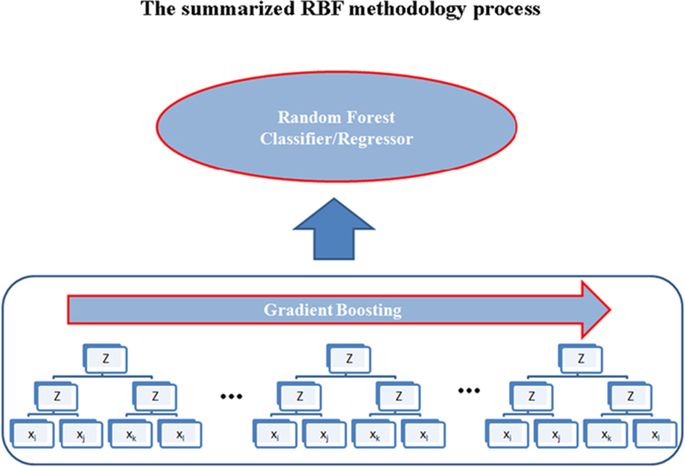
**AdaBoost:**

A picture containing sky

Description automatically generatedהוא אלגוריתם למידה המייצר מסווג לינארי כקומבינציה של מספר מסווגים אשר נבנים בצורה סדרתית.  
בכל איטירציה ממשקלים מחדש את קבוצת הדוגמאות, אחרי המשקול החדש מופעל אלגוריתם הלמידה כדי לבנות את המסווג , משקול הדוגמאות לשלב הנוכחי מתבצע לפי ביצועי המסווג בשלב שלפני, דוגמאות שסווגו לא נכון עולות במשקל כך שיצטרך המסווג החדש להתמודד איתן.

n\_estimators: פרמטר הקובע את מספר עצי ההחלטה בוועדה

**Gradient Boosting****:**

GB מייצר מודל חיזוי בצורה של אנסמבל של דגמי חיזוי חלשים, בדרך כלל עצי החלטה.

הוא בונה את המודל בצורה שלבית כמו שיטות boosting אחרות, והוא מכליל אותם בכך שהוא מאפשר לבצע אופטימיזציה של פונקציית אובדן מובחנת שרירותית.

הפרמטרים שנבחן הם :

* + min\_samples\_split:המספר המינימלי של דוגמאות הנדרש לפיצול צומת פנימי.
  + min\_samples\_leaf: המספר המינימלי של דגימות הנדרש להיות בצומת עלים
  + max\_depth: עומק מקסימאלי של אומדן רגרסיה בודד.
* **דילול תכונות:**

השתמשנו באלגוריתם SelectKBest שsklearn **.**

ובדקתי את ביצועי המסווגים עבור הערך הדיפולטיבי של func\_Score שהוא classif\_f (בגלל שזו בעיית קלסיפיקציה זו הפונקציה המתאימה).

והמשתנה K: שהוא מספר המאפיינים.

האלגוריתם מחזיר את K המאפיינים שקיבלו את ה"ציון" הגבוה ביותר**.**

**תוצאות הניסויים:**

**ביבליוגרפיה:**

[1] [sports gambling industry worth](https://bit.ly/36jY7rn) .