# בדיקת הקשר בין אירועים במשחק כדורגל לניצחון קבוצת הבית

פרויקט בבינה מלאכותית

236502

מגישים:

**מבוא:**

חיזוי תוצאות משחק כדורגל הינו משאלה של כל אוהד כדורגל או מהמר, אך גם משאת ליבם של לא מעט מתכנתים ודי בחיפוש שטחי באינטרנט בכדי למצוא מאגרי נתונים הפרושים לציבור הרחב על מנת לנסות ולהפיק תחזית עבור תוצאת המשחק. יחד עם זאת ככל הנראה שאין אלגוריתם מושלם היודע לספק תחזית בהסתמך אך ורק על סטטיסטיקות קודמות. אילו היה כזה הרי שהמחזיק במידע היה יכול להתעשר בין לילה באמצעות חיזוי טופס טוטו או ווינר עם יחס גדול.

במאמר זה ננסה למצוא קשר בין האירועים במשחק עצמו ובין התוצאה של קבוצת הבית. ניבוי זה נראה לנו יומרני פחות מאחר והוא אינו מתבסס על נתונים וסטטיסטיקות קודמות אלא בעיקר על מידע שברובו נוצר תוך כדי מהלך המשחק. יחד עם זאת הוא מאוד מעניין. ננסה לגלות מהן התכונות הדומיננטיות והאם תכונות שנראות שוליות הינן בעלות משקל משמעותי על תוצאת המשחק. על מנת לגלות ולבחון את הקשר אספנו את הdata הגולמי הפזור ברחבי האינטרנט ועיבדנו אותו לכדי data שמכיל מספר רב של תכונות בכל משחק. את התכונות הנ"ל נחקור לעומק באמצעות אלגוריתמי למידה מסוגים שונים. ננסה להבין מהי הדרך הנכונה להציג את התכונות שחילצנו ואילו מהן אכן קשורות בקשר חזק לתוצאת המשחק.

לאחר חיפוש מצאנו לא מעט מידע באתר Kaggle אשר מכיל מאגר מידע רב. המידע כפי שציינו לא היה מלוטש והיה צורך בעיבודו. במאמר ננסה להבין בין היתר מדוע בחרנו בצורות הצגה מסוימות עבור התכונות שאספנו. חלק זה הינו בעל משמעות רבה שכן חילוץ מידע והצגתו בצורה נכונה יכול להשפיע על רמת הדיוק של האלגוריתם.

**זיהוי תכונות**

התכונות שחילצנו הינן התכונות הבאות:

התאריך שבו המשחק התרחש, סוג הליגה, מספר עונה, המדינה בה התרחש המשחק, קבוצת הבית, קבוצת החוץ, מספר שערים שהבקיעה קבוצת הבית, מספר השערים שהבקיעה קבוצת החוץ, תוצאות ההימורים עבור ניצחון לקבוצת הבית, תוצאות ההימורים עבור תיקו, תוצאות הימורים עבור ניצחון קבוצת החוץ, הזמן בו האירוע התרחש.

לאלו נוספו:

11- Attempt(shot)  
2- קרן  
3- עבירה  
4- כרטיס צהוב  
5- כרטיס צהוב שני  
6-כרטיס אדום ישיר   
7- חילוף  
8- Free kick won  
9- נבדל  
10- נגיעת יד  
11- פנדל   
12 – כדור עובר לקבוצה השנייה  
13 - כדור כושל  
14-כדור חוץ   
15-גול עצמי

כל אירוע שכזה הינו כפול למעשה מאחר והוא יכול לקרות לטובת קבוצת הבית ולטובת קבוצת החוץ. ניתן היה לכווץ את התכונות ולא להכפילם בעמודות נוספות על ידי חיסור בין מספר הפעמים בהם האירוע קרה עבור קבוצת הבית ומספר הפעמים בהם הוא קרה עבור קבוצת החוץ אך העדפנו להכפיל את התכונות ולהכניס כתכונה את מספר הפעמים בהם אירע אירוע i עבור קבוצה j. כלומר חילצנו את מספר הפעמים בהם אירע האירוע לטובת קבוצת הבית והצגנו זאת כתכונה ואת מספר הפעמים בהם האירוע היטיב עבור קבוצת החוץ. בצורה זו נוכל לבדוק האם תכונות דומיננטיות עבור צד אחד הינן דומיננטיות גם עבור הצד השני הרחבה על כך תינתן בהמשך.

מהתכונות הקודמות החלטנו להסיר את המדינה בה המשחק התרחש מאחר והיא כלולה בסוג הליגה.

מרבית התכונות שהשארנו הינן מספריות בדידות הואיל ומדובר **במספר הפעמים** שהאירוע התרחש. מאידך, קיימות גם מעט תכונות קטגוריאליות דוגמת "סוג הליגה" ואף תכונות מספריות אשר הינן למעשה קטגוריאליות דוגמת "מספר העונה" שהרי אין באמת ערך למספר כמספר והוא אינו חשוב מפני הסדר שהוא מייצג. בנוסף לאלו קיימות תכונות מספריות רציפות דוגמת "תוצאת ההימורים עבור קבוצת בית\חוץ\תיקו".

**פיצול ה-DATA**

החלטנו לחלק את הדאטא כך:

Train 60%

Validation 20%

Test 20%

בחרנו גודל יחסית גדול לקבוצת הוולידציה כי רצינו שהבדיקות עליה ידמו בצורה קרובה ככל הניתן את הבדיקות על קבוצת המבחן.

**בדיקת ביצועים**

כדי להעריך את טיב קבלת ההחלטות שביצענו, כמו בחירת המתודות בהן אנו עושים outlier detection, scaling ו-feature selection וקביעת פרמטרים שונים לאלגוריתמים בהם השתמשנו, החלטנו להשתמש במסווג ולבדוק את ביצועיו (אחוזי ה-accuracy) על ה-data שעבר עיבוד על ידינו, אשר מאומן על ה-train data ונבדק על ה-validation data.

החלטנו לבחור במסווג שביצועיו על ה-data הראשוני שלא עבר עיבוד לא גבוהים מידי, כדי שנוכל לראות שינויים יחסית משמעותיים כאשר נריץ אותו על ה-data המעובד. כמובן שלצורך הבחירה הסרנו את התכונות של תוצאת המשחק עבור קבוצת הבית וקבוצת החוץ להם קיים קשר ישיר עם החיזוי (לשם המחשה כאשר בדקנו את ה- accuracyעם התכונות הנ"ל התקבלה התוצאה 1 עם random forest ו- 0.999 עם svm).

אולם ראשית עלינו להתמודד עם השאלה מהי הצלחה. מאחר ולמשחק קיימות 3 אפשרויות שונות ואנו חוקרים רק אחת מהן לא יהיה נכון לומר שמעל 50 אחוז נחשב להצלחה בדיוק כשם שאלגוריתם החוזה את מספר הפעמים בהם לא נקבל 6 בקובייה ומסמן שלא קיבלנו כל הזמן לא יחשב להצלחה מאחר וההסתברות לטעות היא 1/6. על מנת להעריך את טיב הביצועים החלטנו להריץ את הdata שלנו עם עמודות רנדומליות ולבחון את טיב הביצועים שלו.

לצורך בחירת מסווג אשר יעזור לנו רק להעריך את ה-data preparation החלטנו לבדוק את random forest ו-svm בהרצה עם עמודת ערכים רנדומלית התקבלו התוצאות הבאות:

|  |  |
| --- | --- |
| **מסווג** | **accuracy** |
| Random forest | 0.557 |
| svm | 0.557 |

כלומר עבור נתונים רנדומליים אחוז הדיוק הוא קצת מעל 50 אחוז. אם נצליח להשיג אחוזים משמעותיים יותר ככל הנראה שהמסווג עובד. כמות הנתונים הכוללת שאספנו הינה כ-10,000 משחקים ולכן חוק המספרים הגדולים תופס למקרה שלנו. לצורך המחשה עבור הרצה רנדומלית לחלוטין עם מסווג טרינארי ועמודות רנדומליות התקבל דיוק של 0.442.

כעת נעבור לבחירת המסווג הראשוני אשר איתו נבצע בדיקה ראשונית עבור עיבוד הדאטא. בהשוואה בין המסווגים הנ"ל התקבלו התוצאות הבאות:

|  |  |
| --- | --- |
| **מסווג** | **accuracy** |
| Random forest | 0.695 |
| svm | 0.557 |

החלטנו לבחור במסווג svm שביצועיו על ה-data הראשוני היו דומים למסווג רנדומלי ולבדוק אם אכן ביצועיו משתפרים לאחר עיבוד ה-data.

**עיצוב תכונות**

כפי שציינו לעיל ניתן לייצג את הdata שלנו בצורה כפולה כאשר פעם האירועים יספרו עבור קבוצת הבית ופעם עבור קבוצת החוץ (תצוגה א'). אך ניתן לייצגם בצורה של חיסור התכונות בהם קרה הדבר לטובת קבוצת החוץ מהפעמים בהם האירוע התרחש לטובת קבוצת הבית (תצוגה ב'). נשווה בין התוצאות על מנת לזהות האם מתקבלת השפעה משמעותית על הדיוק.

לאחר השוואה התקבלו תוצאות זהות ברמת הדיוק (0.557) בין שני המקרים ולכן בחרנו להמשיך עם תצוגה א' בכדי לראות האם תכונות דומיננטיות עבור קבוצה אחת הינן דומיננטיות גם עבור קבוצה אחרת אך נמשיך להשוות ביניהן בהמשך.

את התכונות הבאות זיהינו כבינאריות ולכן המרנו את ערכיהם למינוס אחד ואחד:

Winner – הקבוצה המנצחת

League – הליגה בה שוחק המשחק

:One hot התכונות הבאות הכילו מעל 2 ערכים אשר אין סדר ביניהן ולכן החלטנו להמירן ל-

Season – שנת העונה

**נרמול**

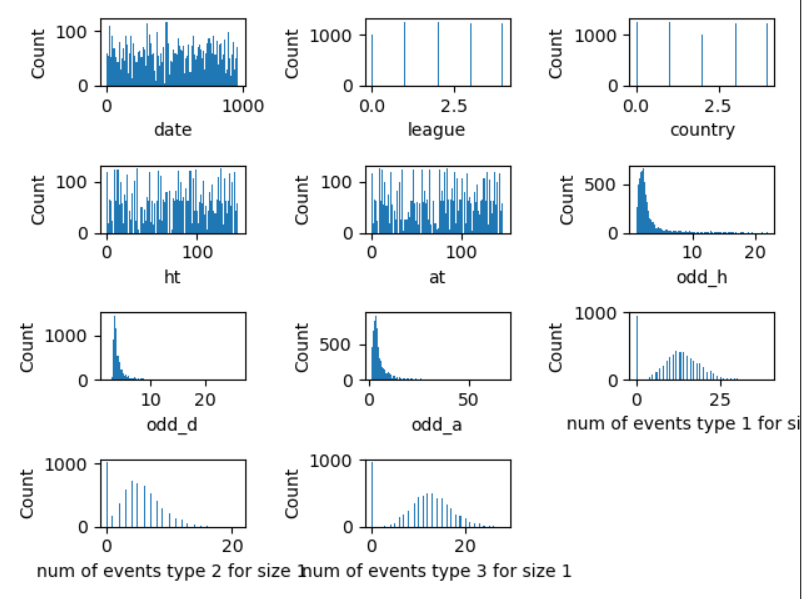
התלבטנו בין 2 אפשרויות נרמול. אחת מסוג min max שנוסחתה היא:

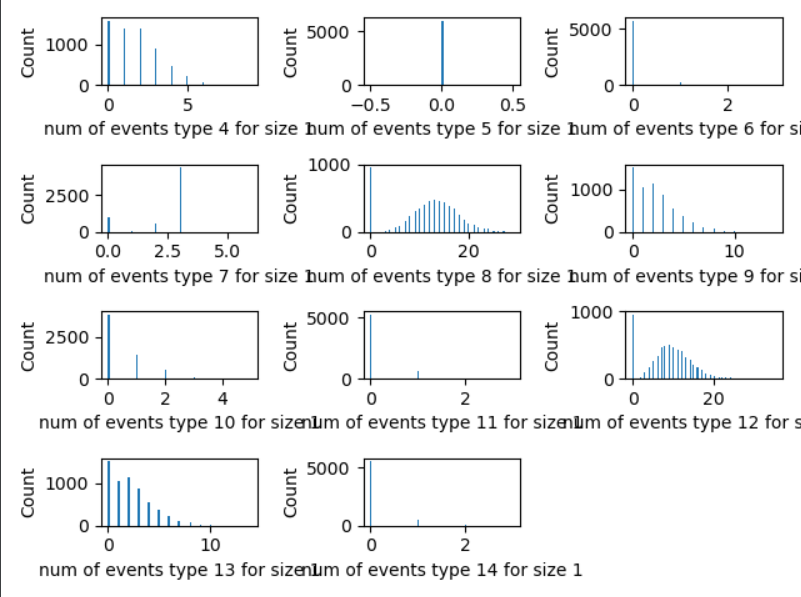
כאשר המטרה היא להביא את כל הערכים לאינטרוול [-1,1]

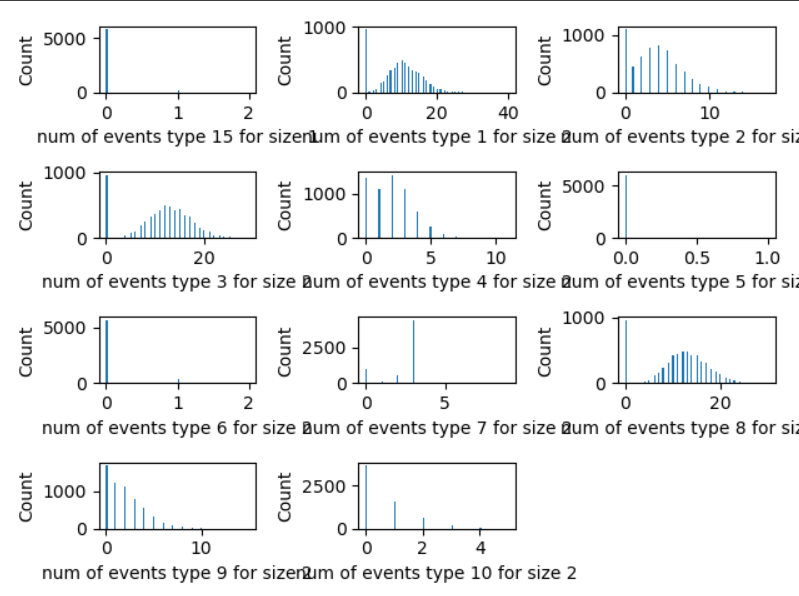
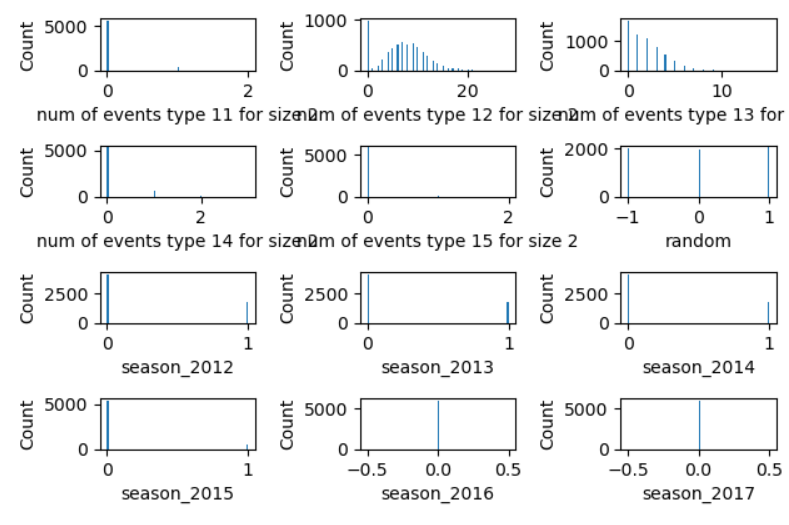
והשנייה מסוג z score שנוסחתה היא כאשר mean זהו הממוצע ו-std זוהי השונות. כלל האצבע הוא שכאשר מדובר בהתפלגות נורמלית לבחור ב-z score וכאשר מדובר בהתפלגות אחידה לבחור ב-min-max.

כך נראו הגרפים של התכונות לפני נרמול:

מקרא: ציר ה-y הוא מספר התכונות עם הערך שמכיל ציר ה-x. הקשר בין המספר והתכונות לפי המקרא המופיע במבוא



****

****

ניתן לראות כי מרבית התכונות אינן בעלי התפלגות נורמלית ולכן בחרנו עבורן באפשרות של z score ואילו עבור קבוצת הבית, קבוצת החוץ והתאריכים ניתן אכן לראות כי מדובר בהתפלגות נורמלית ועבורם נבחר ב-min-max דבר זה לא מפתיע אותנו שכן היינו מצפים ששמות הקבוצות יהיו אחידות לעומת מהלכי המשחק שנוטים להשתנות בין משחק למשחק.

לאחר הנרמול האופן הנ"ל קיבלנו דיוק של 0.711 שזהו שיפור משמעותי. (מאחר ובחרנו עבור הציון מסווג svm סביר להניח שהוא יהיה רגיש לנרמול מאחר והוא תלוי מאוד בסדר הגודל של הערכים).

**הסרת דוגמאות רועשות**

על מנת לזהות דוגמאות רועשות/ חריגות **בקבוצת המבחן** החלטנו להשתמש באלגוריתם Isolation Forest של הספריה Sklearn. האלגוריתם מחשב ערך "נורמליות" ועם הערך נמוך מסף מסוים הוא מגדיר אותה כרועשת.

ערך הנורמליות מחושב כך:

האלגוריתם בונה "יער" של עצים. כל עץ נבנה בצורה איטראטיבית – בכל איטרציה נבחר פיצ'ר וערך פיצול אקראי להפרדת הדוגמאות על פי הפיצ'ר הזה, בדומה לעץ החלטה. ערך הנורמליות של דוגמה הוא ממוצע אורכי המסלולים בעצים עד להפרדה מוחלטת של הדוגמה משאר הדוגמאות.

הרעיון הוא שדוגמה רועשת ניתן להפריד בקלות יחסית משאר הדוגמאות כי היא "מרוחקת" מהם במרחב ולכן מסלול הפרדה עבורה יהיה קצר ברוב המקרים, לעומת דוגמה לא רועשת שלרוב תהיה קרובה להרבה דוגמאות נוספות ולכן מסלול ההפרדה עבורה בעץ יהיה יותר ארוך.

בחרנו באלגוריתם הזה לאחר שקראנו עליו במהלך חיפוש דרכים שונות לזיהוי רעשים, ומצאנו שהוא אלגוריתם יעיל עם ביצועים טובים המתאים לדאטה עם הרבה דוגמאות.

לאלגוריתם יש פרמטר של כמה אחוזים מה-data לזרוק (וע"פ פרמטר זה הוא מחשב את ערך הסף של ה"נורמליות"). על מנת לבחור את הערך לפרמטר זה, ניסינו חמש אפשרויות ובדקנו את הביצועים לאחריהם כפי שניתן לראות בטבלה:

|  |  |
| --- | --- |
| **% of the data to drop** | **Accuracy score** |
| 5 | 0.705 |
| 4 | 0.708 |
| 3 | 0.709 |
| 2 | 0.705 |
| 1 | 0.715 |

הערך הגבוה ביותר התקבל עבור 1% ולכן בחרנו בערך זה. בנוסף הפערים אינם משמעותיים ולכן אין טעם להסיר יותר מכך.

**בחירת מודל**

כעת נשווה בין מספר מודלים על מנת למצוא את המסווג הטוב ביותר עבור ה-data שאספנו. המודלים שנשווה הינם המודלים הבאים:

Svm, Random Forest, Knn, Decision Tree, Voting, Stochastic GD, Multy-layer Preceptron, Pca, Lda .

עבור כל אחד מהם נבחן את הפרמטרים שהוא צריך לקבל ואת התכונות הטובות ביותר באמצעות cross validation. נסביר מעט על המודלים והפרמטרים אותם אנו בודקים עבורם.

**SVM**

מסווג זה מנסה למקסם את השוליים בין דוגמאות עם תווית אחת לדוגמאות עם תווית אחרת. (חסר - הוספת תמונה של מסווג SVM) עבור מסווג זה ניתן לבדוק מספר פרמטרים. כגון פונקציית ה-kernel בה הוא משתמש על מנת לבצע את ההפרדה, ערך הקנס אשר ניתן לדוגמאות החוצות את השוליים. את בדיקת הפרמטרים ביצענו באמצעות cross validation עם k fold השווה ל-10. כלומר חילקנו את קבוצת האימון ל-10 חלקים שווים ובכל פעם חזינו לפי 9 מהם את תוצאות החלק העשירי ולבסוף חישבנו את הממוצע.

התוצאות עבור ערכי הקנס מתוארים בטבלה הבאה:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **linear** | **poly** | **rbf** | **sigmoid** |
| **0.001** | 0.673 | 0.541 | 0.539 | 0.539 |
| **0.01** | 0.696 | 0.557 | 0.541 | 0.615 |
| **0.1** | 0.694 | 0.607 | 0.681 | 0.688 |
| **0.5** | 0.693 | 0.645 | 0.694 | 0.635 |
| **1** | 0.693 | 0.652 | 0.688 | 0.62 |
| **2** | 0.693 | 0.653 | 0.687 | 0.616 |
| **5** | 0.694 | 0.643 | 0.673 | 0.613 |
| **10** | 0.693 | 0.629 | 0.659 | 0.616 |

ניתן לראות שהציון הטוב ביותר מתקבל עבור פונקציית kernel לינארית עם ערך שגיאה של 0.01.

חסר – גרף של הערכים והתוצאות והסבר קצר על הפונקציות עצמן. אפשר להוסיף חישוב של שיפור אחוז הדיוק לעומת 0.55

**בחירת תכונות**

כעת נבצע הסרת תכונות וננסה לזהות את מספר התכונות הדומיננטי שהאלגוריתם מותיר לנו. לצורך כך בחנו את התוצאות בשיטה הפשוטה של חיזוי על קבוצת האימון ובדיקה על קבוצת הוולידציה על מנת לא לסבך את האלגוריתם. את התכונות ניתן להפריד בשתי שיטות אחת בדיקה לאחור(backward) והשנייה בדיקה לפנים (forward) כלומר אחת מתחילה עם כל התכונות והשניה מתחילה ללא התכונות ומוסיפה תכונה כאשר היא משפרת את התוצאות.

תוצאות הדיוק על קבוצת הוולידציה מתוארות בטבלה:

|  |  |  |
| --- | --- | --- |
|  | **backward** | **forward** |
| **ללא הסרת אף תכונה** |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Random Forest**

אלגוריתם זה מקים וועדה של עצים ומקבל החלטה לפי רוב העצים בוועדה. הפרמטרים אותם החלטנו לבדוק הם הפרמטרים הבאים: מספר העצים, עומק מקסימלי של העץ. גם כאן בדקנו כמובן בשיטת cross validation.

להלן התוצאות:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **d/n** | **1** | **5** | **10** | **25** | **50** |
| **1** | 0.562 | 0.624 | 0.614 | 0.573 | 0.592 |
| **10** | 0.631 | 0.67 | 0.668 | 0.647 | 0.65 |
| **25** | 0.628 | 0.67 | 0.675 | 0.668 | 0.665 |
| **50** | 0.646 | 0.671 | 0.683 | 0.673 | 0.67 |
| **100** | 0.637 | 0.67 | 0.682 | 0.681 | 0.681 |
| **200** | 0.643 | 0.67 | 0.683 | 0.683 | 0.681 |

התוצאה הטובה ביותר ניתנת עבור 50 עצים עם עומק 10 אך עבור עוד פרמטרים אשר דורשים הוספת עצים ועומק ולכן משיקולי יעילות בחרנו בנמוך מביניהם.

חסר – גרף של הערכים והתוצאות. אפשר להוסיף חישוב של שיפור אחוז הדיוק לעומת 0.55

**בחירת תכונות**

כעת נבצע הסרת תכונות וננסה לזהות את מספר התכונות הדומיננטי שהאלגוריתם מותיר לנו. לצורך כך בחנו את התוצאות בשיטה הפשוטה של חיזוי על קבוצת האימון ובדיקה על קבוצת הוולידציה על מנת לא לסבך את האלגוריתם. את התכונות ניתן להפריד בשתי שיטות אחת בדיקה לאחור(backward) והשנייה בדיקה לפנים (forward) כלומר אחת מתחילה עם כל התכונות והשניה מתחילה ללא התכונות ומוסיפה תכונה כאשר היא משפרת את התוצאות.

תוצאות הדיוק על קבוצת הוולידציה מתוארות בטבלה:

|  |  |  |
| --- | --- | --- |
|  | **backward** | **forward** |
| **ללא הסרת אף תכונה** | 0.701 | 0.701 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Knn**

אלגוריתם זה מקבל החלטה לפי קבוצת הדוגמאות הקרובה ביותר אשר קיימת לו במאגר (קבוצת האימון). הפרמטרים אותם בחנו הם הפרמטרים הבאים: מספר השכנים הקרובים לפיהם הוא מחליט והמשקולות הניתנים לשכנים כאשר האופציות הן משקולת אחידה או כפונקציה של המרחק.

להלן התוצאות:

|  |  |  |
| --- | --- | --- |
|  | **uniform** | **distance** |
| **1** | 0.575 | 0.575 |
| **2** | 0.598 | 0.575 |
| **3** | 0.596 | 0.596 |
| **5** | 0.604 | 0.604 |
| **10** | 0.626 | 0.620 |
| **25** | 0.64 | 0.64 |
| **50** | 0.652 | 0.652 |
| **100** | 0.653 | 0.653 |
| **150** | 0.655 | 0.655 |
| **200** | 0.652 | 0.652 |

ניתן לראות כי לא קיים הבדל משמעותי בין הפונקציות והדיוק הטוב ביותר ניתן עבור 100 תכונות.

חסר – גרף של הערכים והתוצאות. אפשר להוסיף חישוב של שיפור אחוז הדיוק לעומת 0.55

**בחירת תכונות**

כעת נבצע הסרת תכונות וננסה לזהות את מספר התכונות הדומיננטי שהאלגוריתם מותיר לנו. לצורך כך בחנו את התוצאות בשיטה הפשוטה של חיזוי על קבוצת האימון ובדיקה על קבוצת הוולידציה על מנת לא לסבך את האלגוריתם.

את התכונות ניתן להפריד בשתי שיטות אחת בדיקה לאחור(backward) והשנייה בדיקה לפנים (forward) כלומר אחת מתחילה עם כל התכונות והשניה מתחילה ללא התכונות ומוסיפה תכונה כאשר היא משפרת את התוצאות.

תוצאות הדיוק על קבוצת הוולידציה מתוארות בטבלה:

|  |  |  |
| --- | --- | --- |
|  | **backward** | **forward** |
| **ללא הסרת אף תכונה** | 0.682 | 0.682 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |