**הרצאה 3**

**Bagging-** לוקחים מדגמים שונים מאותה קבוצת נתונים ולפיהם בונים עצים. עבור כל הרצה נקבל עץ שונה. לאחר מכן על מנת למצוא סיווג של נקודה בנתונים(תכונות+סיווג) נריץ את הנקודה בכל העצים ונבצע ממוצע של התוצאות שיוצאות מכל עץ וכך נקבל את הסיווג הכי מדויק (תוצאות בין 0 ל1 כך שהתוצאה יותר קרובה ל0 אז הנקודה מסווג לקבוצה 0 וכאשר התוצאה יותר קרובה ל1 אז הנקודה מסווגת לקובצה 1.

ככל שהסטיית תקן יותר גדולה כך המסווג פחות בטוח בעצמו.

=התוצאה שיוצאת מהרצה של העץ מספר b

=הערך הממוצע שמתקבל מכל ההרצות של כל העצים

B= מספר העצים

=סטיית התקן, רמת הביטחון שהסיווג נכון. ככל שסטיית התקן גדולה יותר, הביטחון בסיווג נמוך יותר.

Out of bag error: עבור כל נקודה נחפש את העצים שבהם לא אימנו את העץ על נקודה זו, נעביר את הנקודה בעצים אלו ונבדוק את ממוצע התוצאות שמתקבלות מעצים אלו, על מנת לראות האם העצים מסווגים נכון- זה הtest.

Fit()- פונקציה אשר בונה מסווג

Test()- בודק את המסווג

**Random Forest-** סוג של bagging אשר ספציפית לעצים ועובדת יותר טוב מהbagging הרגיל עם עצים.

בשלב הראשון מחלקים, כמו בbagging, את הנתונים לקבוצות, באמצעות כל קבוצה בונים עץ. **בכל רמה בעץ בוחרים באופן רנדומלי תת קבוצה בגודל s מהp features** ובוחרים את הcut רק מהם.

הדיפולט של הגודל של s הוא .

מקבלים מה כל מסווג אמר עבור כל נקודה, הנקודות שבהם לפחות רוב העצים נתנו את הסיווג הנכון זה אומר שהסיווג עבד טוב.

אם הדיוק של עץ אחד הוא 80% אז כאשר העצים בקורלציה מלאה הסיכוי לקבל תוצאה נכונה הוא גם 80%.

במקרה בו הסיכוי שלפחות 2 עצים מסווגים נכון, ההסתברות לסיווג נכון היא 0.896

בעולם האמיתי מקבלים בדרך כלל עצים שונים, הקורלציה ביניהם לא מלאה, לכן נקבל תוצאה שיותר כקרובה למצב שבו יש לפחות 2 עצים שמסווגים נכון. זאת אומרת שהמטרה היא לבנות עצים מסווגים באיכות דומה, אבל לא מתאומים לגמרי.

בדיקה של איזה features חשובים

בודקים איפה הfeature נמצא בעץ, נותנים לכל feature ציון לפי הגובה שהוא נמצא בעץ-

ככל שהציון גבוה יותר כך הfeature יותר חשוב.

במקרה ונסתכל על כל feature בנפרד יהיה קשה להבין איזה feature יותר חשוב. לכן לאחר שנחלק את הנתונים לפי feature מסוים ואז נבדוק את ערך הfeature אז נוכל לראות האם הfeature מקבל ערך יותר גדול ואז נוכל לדעת אם הוא חשוב בשילוב של חלוקה עם הfeatures שלפניו.

כך ניתן לקחת את כמות הfeatures הכי טובים ולייצר מסווגים חדשים אשר משתמשים רק בfeatures אלו-שיטה של feature selection

גישה נוספת כללית יותר היא:

לבנות random forest, לחשב את הOOB-Out Of Bag error, לוקחים feature אחד ומערבבים את הערכים של הfeatures בין האובייקטים בנתונים- נותנים לאובייקטים ערכים שונים של אותו feature מאובייקט אחר. נחשב מחדש את ה OOB ואז נחשב את ההפרש בין הOOB הקודם לישן.

ככל שfeature זה יהיה חשוב יותר אז הOOB החדש יהיה יותר גדול, לכן ככל שהפרש בין הOOBs גדול יותר כך הfeature יותר חשוב (אם דופקים את feature זה- הטעות גודלת יותר).

מבצעים את תהליך זה לכל feature בנפרד וכך מחשבים את החשיבות של כל feature.

**אלגוריתם AdaBoost**

אלגוריתם זה מסתכל על מה שלומדים קודמים טעו בהם ומתמקד בהם.

במקום לקחת מסווג אחד מאוד טוב, עדיף לקחת הרבה מסווגים פחות טובים אשר ביחד מסווגים יותר טוב.

בונים מסווג, בוחנים אותו, מסתכלים על הדוגמאות שהמסווג טעה בהם, לאחר מכן בונים מסווג אשר יעבוד על סיווג הנתונים שבהם המסווג הקודם טעה, וכך הלאה. כל מסווג הוא עץ בגובה 1 (stump, אב עם 2 ילדים)

בזמן הtest נתן דוגמה כלשהי, ניתן לכל המסווגים לסווג אותה, כאשר כל מסווג נותן תוצאה 1- או 1. נכפיל את התוצאות של כל המסווגים, אם נקבל תוצאה אשר מתאימה לסיווג הנכון אז המסווגים יחד פועלים טוב.

N- מספר הדוגמאות

M- מספר המסווגים

כל דוגמה מקבלת תחילה משקל /N1, *כל דוגמה שהמסווג טעה לגביה, מעלים לה את המשקל*

-שורה במטריצה

-בודקת אם התשובה נכונה

על מנת לחשב את אחוז הטעויות, מחשבים את מספר הטעויות\* משקל הדוגמה חלקי סכום משקלי הדוגמאות

השגיאה המקסימלית שיכולה להיות היא 0.5, כיוון שאם נקבל שגיאה גדולה מ0.5 נוכל לקחת את התשובה שנותן המסווג ולסווג ההפך ואז זה יהפוך לסיווג נכון.

לכן אם יש מסווג שצודק בחצי מהמקרים הוא לא טוב

ככל שהerror נמוכה יותר אז המסווג יותר מדויק

ככל ש יותר גבוה, המסווג מקבל משקל יותר גבוה

*כל דוגמה שהמסווג טעה לגביה, מעלים לה את המשקל*

*exp= המספר e*

*מהנוסחה ניתן להבין שאם המסווג צדק לא נשנה את משקל הדוגמה, אם המסווג טעה אז המשקל שתקבל התצפית תגדל.*

*Stump=עץ בעומק אחד, מסווג רק לפי feature אחד.*

*הקו הירוק בגרף- התוצאה של כמה מסווגים ביחד, על פי* adaboost.

*מחשבים את הסכום של משקלי כל הדוגמאות בקבוצה אחת חלקי סכום משקלי כל הדוגמאות*

*כך גם לקבוצה השנייה.*

*כך נקבל את ההסתברות לגבי כל קבוצה על פי המסווג.*

*על פי תוצאה זו נוכל להבין מה רמת הדיוק של המסווג*

*אופציה נוספת היא לבנות וקטור של כל המשקולות של הנתונים*

*כאשר המקום הראשון היא הסכום של המשקל הw1, המקום השני הוא הסכום של המשקלים הראשון והשני, וכו'..*

*על הוקטור הזה מגרילים מספר רנדומלי מ0 עד סכום כל המשקלים. משתמשים במספר שיצא לחיפוש בינארי, אשר אומר איפה הערך שהשתמשנו בחיפוש אמור להיות בוקטור, התא שנבחר מכיל את הנקודה במספר שיצא בחיפוש בינארי. שומרים את נקודה זו. מבצעים את תהליך זהn פעמים וכך נקבל את הנקודות שבהן נשתמש כדי לבנות את המסווג. ככל שמשקל של נקודה גדול יותר כך הרווח לתא הבא הוא גדול יותר ולכן לנקודה זו יהיה יותר סיכוי להיבחר. כך נקבל את הנקודות עם המשקל הגדול יותר על מנת לבנות את המסווג.*

Gradient Boost

*יש הסבר בYouTube של* statQuest

*עוסק ברגרסיה ולא סיווג*

*לוקחים מסווג, בודקים באילו נתונים הוא טעה, ובונים מסווג חדש על הנתונים שהמסווג טעה בהם, וכך הלאה.*

*G- מסווג*

*- המסווג הראשון נותן ערך ממוצע לכל הנתונים על פי הממוצע של כל הערכים של הfeature שמסתכלים עליו.*

*Residual- עד כמה העץ טעה, עבור כל רשומה נקבל ערך residual מסוים ונבנה מסווג על ערכים אלו.*

*ואז ערך השגיאה של המסווג החדש יהיה המסווג הישן+ אלפא\* מה שמהסווג החדש נותן  
((*

*\**

*???? לא ברור, לראות את הסרטון*