**הרצאה 2**

דוגמה: חברת טלפונים רוצה למצוא לקוחות חדשים לאפליקציה

עלות=מחיר קבוע+מספר לקוחות\*קבוע

המטרה היא לשלוח ללקוחות אשר יענו

2-3% עונים

לכן ניתן לסדר את הלקוחות על פי ציון ונשלח ללקוחות עם הציון היותר גבוה

כך מספר הלקוחות ששולחים אליהם יורד והעלות תהיה קטנה יותר.

במקום לשלוח אקראית שולחים בצורה מכוונת לקבוצת עניין.

על מנת לעשות זאת לומדים ממודל קיים, אם יש.

אם אין אז קודם כל מורידים לקוחות שלא עונים אף פעם.

לאחר מכן לוקחים מוצר דומה קיים, לוקחים את כל הנתונים על הלקוח שיש לנו+ האם קנה את המוצר הקודם, כך נבנה מסווג האם הלקוח יקנה את המוצר החדש.

מקורות מידע: קובץ מרקטינג על הלקוח, בסיס נתוני שיחות, בסיס נתונים דמוגרפיים.

לשים הכל יחד ולתת לתכונת למידת מכונה להבין מי יקנה ומי לא.

פרמטרים טובים להגדרת לקוח:

כמה דקות אדם מדבר בטלפון, כמה שיחות נכנסות יש לו, אורך השיחות, לכמה אנשים שונים מתקשר, האם יש לו אופציה לקבל הודעות קוליות.

בנו שני רשימות: אחת של לקוחות אקראיים ואחת על פי הData mining.

נמצא שבקבוצה של הData mining 15% קנו את המוצר לעומת 3% בקבוצה השנייה.

זאת אומרת שזה חמישית מהעלות לקבל את אותן תוצאות בשימוש בData mining.

ניקח את התוצאות שקיבלנו ונשתמש בהם כדי לבנות מודל למוצר החדש על מנת לשפר ביצועים.

נשתמש במודל זה על כל השוק.

דוגמה נוספת: מכירת רכבי SUV

הצעה אחת היא לבנות מודל על פי ראיונות עם מומחים

הצעה שנייה היא לעצב מודל על פי נתונים.

יש 2 שלבים:

השתמשו ב רשת נורירונים כדי להבין אם אדם רוצה לקנות את הרכב.

לאחר מכן השתמשו בעץ החלטה על מנת להבין האם האדם יקנה את הרכב הספציפי הזה.

כך נקבל תוצאה יותר טובה מאשר שימוש במודל אחד.

בזכות זה שיש מיקוד טוב יותר לקבוצת עניין ניתן לתת לכל אדם שבא לבדוק את האוטו מתנה טובה יותר, על מנת להעלות את הסיכוי שיקנה.

**עצי החלטה**

בעלת גרסת סיווג- נותן לאיזה מחלקה שייך, ההסתברות להיות בכל אחת מהמחלקות.

וגם בעל גירסת רגרסיה-הערכת ערך מספרי מסוים +-

דוגמה: להחליט האם לשחק גולף או לא

Features: מצב מזג האוויר(שמשי, מעונן, גשום), טמפרטורה(חם, נעים, קר), רמת לחות(נמוכה, נורמלית, גבוהה),רמת רוח(true,false)

Labels:N(no)/P(play)

האלגוריתם מסתכל על הData ומחליט איך לסדר את העץ ומתי להגיע להחלטה, ומוציא את העץ הבא:

Diagram

Description automatically generated

ניתן לתרגם את העץ החלטה לחוקים וכך ניתן להסתכל על העץ החלטה ולהבין מה קורה.

העצים באלגוריתם עצי החלטה לא חייב להיות מאוזנים.

אלגוריתם טוב לשימוש לסיווג או חיזוי.

מקבלים את החוקים שעל פיו הוא עובד וניתן להשתמש בהם לדברים אחרים כמו שאילתת SQL.

מאפשר להבין מתוך הנתונים מסקנות לגבי כל תכונה. ככל שהתנאי לגבי התכונה מופיע יותר גבוה בעץ כך תכונה זו היא חשובה יותר לקבלת ההחלטה.

בכל שלב שמטיילים בעץ מקבלים פחות נתונים ויותר ספציפיים.

עץ בינארי- לכל החלטה יש רק 2 אופציות

עץ Nי- לכל החלטה יש N אופציות

הפיצול הכי טוב נחשב לפיצול שמפצל את הנתונים בצורה הכי טובה כדי להגיע לתשובה לגבי כל נתון. פיצול טוב הוא פיצול שמפצל את המידע בצורה מאוזנת (50/50).

טהורות- עד כמה הפיצול הוא טוב בלחלק את הנתונים למחלקות. ככל שהציון נמוך יותר(קרוב ל0) אז הפיצול יותר טוב.

נוסחות לחישוב את טהורות הפיצול:

נוסחה לחישוב **misclassification**- סיווג לא נכון: כאשר הוא ההסתברות לקבל את הclass הכי גדול.

**Gini**:הערך המקסימלי שניתן לקבל הוא ½ המצב הגרוע ביותר, כאשר ההסתברות שהדוגמה שייכת לכל class היא ½ . ככל שהחלוקה יותר טהורה אז הערך יותר נמוך-יותר קרובה ל0.

**Cross-entropy**: במצב שיש 2 class וכל הדוגמאות הם מclass אחד אז נקבל תוצאה 0- המצב הכי טוב. אם ההסתברות שכל דוגמה שייכת לקבוצה היא ½ אז נקבל 1- המצב הגרוע ביותר

ההסתברות שהדוגמה שייכת למחלקה מסוימת.

k-מספר הclasses.

**אם הציון של הפיצול הוא נמוך יותר מהציון של האב אז כדאי לבצע את הפיצול**

הנמכת שונות

c-ממוצע=

-מספר הדגמאות בקבוצה

-הערך של הפרט הנוכחי

אלגוריתם Decision Tree

Text

Description automatically generated

עבור כל אחד מהfeatures בוחרים ערך מסוים של ה feature על פיו נחלק את הנתונים ל2 קבוצות, הcut הוא החלוקה לפי ערך זה.

מחלקים את הקבוצות על פי ערך זה ( Cut(X,v,cut) )

בודקים את הציון של חלוקה זו

אם הציון של חלוקה זו יותר נמוך מהציון של החלוקה הקודמת

אז שומרים את חלוקה זו, הציון שלה כציון החלוקה הכי טובה, והערך על פיו מחלקים.

Text

Description automatically generated

משתמשים במיון על מנת לבצע את האלגוריתם בצורה יותר יעילה

Overfitting- העץ עובד ממש טוב רק על נתונים ספציפיים ולא על נתונים אחרים

**שיטות להימנע overfitting:**

**Pruning:**בונים את עצים על הtraining data בעומקים שונים ובודקים את הerror rate

בודקים את העצים גם על הvalidation data-קבוצת נתונים אחרת כדי לבדוק את העץ ובודקים את הerror rate שמתקבל

נחפש את עומק העץ בעל הerror rate הקטן ביותר עבור גם הtraining data וגם הvalidation data. הerror rate של הtest data -קבוצת נתונים שלא השתמנו בה בתהליך הבחירה בכלל צריך להיות בערך כמו ה error rate של הvalidation data.

**אופציה נוספת לpruning** היא לגזום כל פעם את הפיצול בעץ שעושה את השיפור הכי קטן בעץ ונבדוק את העץ על הvalidation data כדי למצוא את הנקודה בה אנחנו מוצאים את הerror rate הכי קטן.

**חוסר יציבות**

בודקים את האחוזי חלוקה לקבוצות בכל פיצול גם לtraining data וגם לvalidation data.

אם יש הבדל גדול בחלוקה לאחוזים בין הtraining לvalidation אז צריך לוותר על חלוקה זו.

**לייצר עוד פיצ'רים שהם פונקציות של הפיצ'רים הקיימים**

נבדוק האם הפיצ'רים המומצאים משפיעים על יצירת העץ. אם האלגוריתם מתעלם מהפיצ'רים

Tree ring diagram-דיאגרמה אשר מאפשרת לראות איך העץ מחלק את הנתונים, גודל החלוקה מראה את גודל כל class, ככל שהגוון של החלוקה יותר גבוה כך הסיכוי להיות בחלוקה זו גבוה יותר.

יתרונות של עץ החלטה

-קל להבין איזה תהליך חלוקה התבצע

-מתודה שמשמת גם לבעיות אחרות בעולם

-לא נדרש לבצע הנחות מוקדמות על הנתונים

-מאפשר לעבד גם נתונים נומריים וגם כטגורליים

חסרונות

-מוציא רק output אחד- מה הclass

-אלגוריתם עצי החלטה הם מאוד לא יציבים, עלולים להשתנות מאוד ביחס לנתונים שמשתמשים בהם. ניתן להריץ כמה פעמים את האלגוריתם על מדגמים שונים של אותה קבוצת נתונים ויכול להיות שנקבל כל פעם עץ אחר.

-עצים שיוצרים מנתונים מספריים עלול להיות מאוד עמוקים ומסובכים