למידת מכונה:

תוכן עניינים

מרצה: ד״ר גדי פנקס

נכתב על ידי: יעלי בר גימלשטיין

מבוא ללמידת מכונה:
למידת מכונה, Machine Learning , היא תת-תחום במדעי המחשב ובבינה מלאכותית המשיק לתחומי הסטטיסטיקה והאופטימיזציה. התחום עוסק בפיתוח אלגוריתמים המיועדים לאפשר למחשב ללמוד מתוך דוגמאות, ופועל במגוון משימות חישוביות בהן התכנות הקלאסי אינו אפשרי.

דוגמאות ללמידת מכונה:
טכנולוגית למידת המכונה מופיעה בכמעט כל תחום בחיינו

- ✓ חיפוש בגוגל: השלמת חיפושים, Ranking
- ✓ תיוג אוטומטי של תמונות חדשות שהועלו ל FB
- ✓ זיהוי Spam email
- ✓ התראות החלפת נתיב, מכונית אוטונומית
- ✓ זיהוי פרצופים, זיהוי דיבור,
- ✓ כריית מידע- מציאת תבניות בבסיסי נתונים

ההבדל בין בינה מלאכותית ולמידת מכונה:
בינה מלאכותית מייצגת את הרעיון בו מכונות, תוכנות או כל מנגנון טכנולוגי מחקה מנגנון חשיבתי אנושי. הדבר בא לידי ביטוי ברמת בניית אפליקציות ומערכות שונות. למשל, אפליקציה שמשחקת נגדנו שחמט או השואב הרובוטי שדואג לא ליפול במדרגות, לחשב את גודל החדר, מסלול הניקוי האופטימלי, ואפילו יודע לחזור לבד לתחנת הטעינה שלו כי הסוללה שלו נגמרת.

לעומת זאת, למידת מכונה, מייצגת את רעיון טיפול ממוחשב בנתונים מן העולם האמיתי עבור בעיה מסוימת, כאשר לא ניתן לכתוב תוכנת מחשב עבורה. הדבר בא לידי ביטוי במידול, חיזוי או גילוי (דטקציה) של עובדות לגבי העולם האמיתי.

לסיכום הדברים, בינה מלאכותית (AI) פותרת משימות הדורשות אינטליגנציה אנושית בעזרת אפליקציות מובנות ומוגמרות בעוד למידת מכונה (ML) היא קבוצת משנה של בינה מלאכותית הפותרת משימות ספציפיות עבור בעיות אמיתיות מעולמנו על ידי למידה מנתונים ותחזיות.

הצורך בלמידת מכונה:
למידת מכונה, כאמור, היא טכנולוגיה מיוחדת השונה מתכנות פרוצדורלי.
נשאלת השאלה מדוע תכנות פרוצדורלי אינו מספיק?

התשובה לשאלה זו היא פשוטה- קשה. קשה לכתוב תכנית פרוצדורלית לבעיות שהמוח האנושי יכול לפתור אבל המחשב לא. כלומר, קשה לכתוב תוכניות שפותרות בעיה שקשה להגדיר באופן מדויק. תוכניות אלה לא כלכליות, התוצאה שלהן פחות טובה ולפעמים זה פשוט בלתי אפשרי להגדיר בדיוק את הבעיה. בנוסף, הקושי נובע מכך שהעולם האמיתי עטיר פרטים (חלקם חשובים וחלקם לא) ועטיר חוסר ודאות.

פתרון למידת מכונה הוא פתרון מוצלח. במקום לכתוב תוכנית, נראה למכונה הרבה דוגמאות שמתארות מה הפלט הנכון לקלט נתון. אלגוריתם ML מיצר תוכנית שיודעת לייצר את הפלט הנכון בהנתן קלט.

יתרונותיה העיקריות של למידת מכונה:

- ✓ הררי נתונים + חישוב זול (יחסית), לכן, כלכלי יותר להפיק תוכניות בצורה אוטומטית מאשר לשלם לתוכניתנים עבור כתיבת תוכנות מסובכות ועתירות תחזוקה
- ✓ בעיות קשות במיוחד לא ניתן כלל להגדיר עבור מתכנת ותוצאת ה ML תהיה בד"כ טובה יותר

חסרונותיה העיקיריות של למידת מכונה:
לא קל לעשות ML ולא תמיד מצליחים.

מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

ההבדל בין תכנית המיוצרת על ידי מתכנת לבין תכנית למידת מכונה:

	למידת מכונה	תכנית רגילה
ארכיטקטורה	בד"כ ארכיטקטורה כללית שנקבעה מראש, אך מכילה גמישות המאפשרת התאמה של הארכיטקטורה (או פארמטרים בה) לנתונים	בד"כ ארכיטקטורה מדויקת, ספציפית שנקבעה מראש, ללא גמישות
למידה	אם ה"למידה" מצליחה, התוכנית תדע ל"הכליל" לקלטים שעדיין לא נראו	התכנית לא לומדת, יודעת לבצע פעולה כלשהי עבור קלט מוגדר מראש
שינויים	כל זמן שיש לנו דוגמאות חדשות, נוכל להמשיך באימון והתוכנית שלנו תמשיך להשתנות	התכנית קבועה ואינה משתנת

מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

שימושי למידת מכונה:
ללמידת מכונה יש שימושים רבים בעולם העסקי, ביניהן

זיהוי וחיזוי Recognition, Prediction ניבוי מה יהיה בעתיד.
מה הסיכוי של לקוח לנטוש את החברה/חנות/מוצר, איזו פרסומת/קופון כדאי להציג לאיזה לקוח, מה יהיה מחיר הזהב בעוד שבוע.

מודלים שבעזרתם ניתן לזהות ולחזות:

1. מודל רגרסיה:
בסטטיסטיקה, ניתוח רגרסיה הוא שם כולל למשפחה של מודלים סטטיסטיים להערכת הקשרים בין משתנים. המשותף לכל המודלים הוא קיומם של משתנה התלוי ומשתנה בלתי תלוי אחד או יותר.
בעזרת מודל רגרסיה ניתן ללמוד כיצד ערכו של המשתנה התלוי משתנה כאשר חל שינוי בערכו של אחד המשתנים הבלתי תלויים.

מבחינה הסתברותית, מודל הרגרסיה אומד בדרך כלל את התוחלת המותנית של המשתנה התלוי בהינתן המשתנים הבלתי תלויים. הפלט הוא משתנה רציף. כעזר בהבחנה, ניתן לחשוב על הפלט, אם נרצה פלט רציף עלינו להשתמש ברגרסיה.

דוגמא: בהינתן מאפינים (גיל, שנה, השכלה) רוצים לחזות ערך נומרי (שכר).

2. מודל קלסיפיקציה:
בסטטיסטיקה, קיטלוג / סיווג/ קלסיפיקציה היא פעולה שמחלקת קבוצת עצמים לתת-קבוצות. לעיתים, אין סיווג "נכון". במקרים אחרים, פעולת הסיווג אמורה לחקות חלוקה שאינה תלויה בפעולת הסיווג עצמה, ואז כלל הסיווג נקרא מסווג.

קיים משתנה תלוי ומשתנה בלתי לוי אחד או יותר. בעזרת מודל קלסיפיקציה ניתן ללמוד כיצד ערכו של המשתנה התלוי משתנה כאשר חל שינוי בערכו של אחד המשתנים הבלתי תלויים. על פי מאפיינים רוצים לסווג את המשתנים הבלתי תלויים לקטגוריות בדידות. הפלט הוא משתנה קטגורי, בדיד.

כעזר בהבחנה, ניתן לחשוב על הפלט, אם נרצה פלט בדיד, עלינו להשתמש בקלסיפיקציה.

דוגמא: על פי מאפייני גיל, משכורת ומצב משפחתי, רוצים לצפות את הסיכוי שאדם יחזיר משכנתא.

ניתן לחלק את מודל הקלסיפיקציה לסוגים:

- ✓ קלסיפיקציה בינארית, נקרא גם זיהוי קונספט: שאלת כן או לא.
- ✓ קלסיפיקציה רבת קטגוריות Multi-class, Multi-Category: קיימים שני סוגים
  - קטגוריות הדדיות, Mutual Exclusive Categories
    זיהוי פרצופים, ספרות, הפעלת כיסא גלגלים ב EEG, זיהוי מילה שנאמרה בדיבור
  - כמה קבוצות של קטגוריות
    זיהוי הפרצוף, מינו ומצב רוחו

הסקה וגילוי ידע Knowledge Discovery פרשנות לאירועים שקרו.
אילו מאפיינים יש ללקוחות שנוטשים בסיכוי גבוה, מציאת סוגים שונים של לקוחות שמתנהגים בצורה אופינית, מה מאפיין מוצרים שנמכרים היטב למגזר לקוחות מסוים.

לא תמיד מסתפקים בחיזוי נכון של הפלט (קופסא שחורה). לפעמים רוצים להבין או להסביר את המודל. למשל, איזה קשר יש למשתנה פלט עם כל מאפייני הקלט? האם ניתן לקבל תמצית הקשר באמצעות משוואה פשוטה? או חוקים (לוגיים) שניתן להבינם?

מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

דוגמאות עבור שימוש בהסקה וגילוי ידע: ככל שתקציבי הפירסום גבוהים המחירות גבוהות יותר, אך אחוזי הגידול זניחים בתקציבים הגבוהים. פירסום ברדיו עובד כאשר אין פרסום בטלוויזיה, אך אינו מוסיף למכירות כאשר ישנו תקציב גדול לפרסום טלוויזיה.

עסק עשוי להתענין יותר או פחות בחיזוי לעומת גילוי ופרשנות. למשל: עסק נדלן יכול להתענין בפרשנות: כמה משפיעה הקומה על המחיר? כיצד משפיע אחוז הפשיעה על המחיר? לחילופין, העסק יכול להיות מעונין במודל חיזוי המעריך מחיר דירה על פי מאפייניה (מבלי להתעניין בפרשנות).

ישנם אלגוריתמים היכולים לשמש גם לחיזוי וגם בפרשנות- למשל מודלי חיזוי לינאריים הינם פשוטים להסבר ולכן שימושיים גם לפרשנות. לעיתים, מודלים מסובכים מאוד (למשל רשתות נוירונים) מאפשרים חיזוי מדויק אך אין אפשרות לפרש אותם– הסקת מסקנות עסקיות לכן הינה יותר מאתגרת...

זיהוי מצביי אנומליה Anomaly Detection התנהגות לקוח לא אופינית, חריגות.
בבכרטיס האשראי - Fraud, בתנועות העכבר- פריצה למחשב.

מרצה: ד״ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

טכניקות של למידת מכונה:

למידה מונחת:

למידה מונחית, Supervised learning , היא טכניקה בענף למידת המכונה, המאפשרת לפתח מכונה או מערכת (בדרך כלל תוכנית מחשב) שלומדת לפתור בעיות על בסיס מאגר גדול של דוגמאות "פתורות". הלמידה עצמה נעשית באמצעות חיפוש היפותזה - פונקציה ממרחב הדוגמאות למרחב התיוגים שמתארת את המידע בצורה הנכונה ביותר. אלגוריתם למידה אופטימלי הוא אלגוריתם שהפונקציה הנלמדת על ידיו תוכל לחזות נכונה את התשובה גם בעבור דוגמאות שטרם נראו על ידי המערכת.

או במילים פשוטות, לדאטה שנאספה יש תגיות: מניחים שקיים מורה שיודע לתת תוויות לדוגמאות הדאטה בעזרת תגיות אלו, אלגוריתם למידה מתאמן ומייצר מודל חיזוי.

בעזרת למידה מונחת ניתן לבצע מספר שימושים של למידת מכונה: חיזוי וניבוי בעזרת המודלים שפורטו וכן הסקה וגילוי ידע.

מתודולוגיה בעלת 3 שלבים ללמידה מונחת:

1. אימון. בניית מודל חיזוי מנתונים מתוייגים.
2. אימות. בחינת המודל אל מול נתונים שלא נראו. שפר את האלגוריתם במידת הצורך.
3. בחינה אחרונה
4. ייצור. שימוש במודל המוגמר באלפקיציות העולם האמיתי.

למידה בלתי מונחת:

למידה בלתי מונחית, Unsupervised Learning , היא טכניקה בענף למידת המכונה, שבה מנסים ללמוד את התכונות והמבנה של אוסף דוגמאות נתונים כאשר הנתונים זמינים כפי שהם ללא תוספת תיוגים.
למשל, נתונים הכוללים מידע רפואי על נבדק, כמו: חום, דופק, לחץ דם; ללא תיוג המציין אם הנתונים שייכים לאדם חולה או בריא.

ואכן, רוב הדאטה הנאספת ע"י האנושות היא ללא תגיות, האם ניתן להשתמש בדאטה זה עדיין. ניתן.

התובנות לגבי התכונות של הנתונים הבלתי מתויגים יכולות לשמש למשל כדי לזהות אנומליות או כדי לחלק את הנתונים לקטגוריות בדרכים שונות:

1. ניתוח אשכולות, קלסטרינג:
   בכריית מידע וסטטיסטיקה, ניתוח אשכולות, Cluster Analysis , מתייחס למשימה של קיבוץ אובייקטים לקבוצות (אשכולות) כך שהאובייקטים הנמצאים באותה קבוצה דומים זה לזה יותר מאשר לאובייקטים השייכים לקבוצות אחרות.

   לניתוח אשכולות יש שימושים רבים במגוון תחומים. לדוגמה במחקר שיווקי, ניתוח אשכולות משמש לביצוע פילוח של הלקוחות לפי התנהגות צרכנים ותכונות דמוגרפיות. ביולוגים מקבצים מידע גנטי לאשכולות כדי לאתר תתי אוכלוסיות או זנים. בסוציולוגיה נעזרים בניתוח אשכולות כדי לחלק את החברה לתת-קבוצות על בסיס קשרים בין-אישיים.

2. דחיסה של דוגמאות ממימד גבוה (הרבה מאפיינים) לוקטור מספרים ממימד נמוך יותר
3. זיהוי חריגה מנורמליות: למידה מהי התנהגות נורמלית וזיהוי דוגמאות שחורגות מהנורמה
4. מציאת פונקצית התפלגות "נורמלית", חיזוי מתי דגימה שייכת או איננה שיכת לפונקצית ההתפלגות

זאת להבדיל מלמידה מונחית שבה הנתונים הזמינים כוללים גם תיוג ומטרת הלימוד היא בדרך כלל לחזות את התיוג של נתונים עתידיים (למשל לחזות אם נתונים רפואיים שלא תויגו עדיין שייכים לאדם חולה).

למידת חיזוק:

למידת חיזוק, Reinforcement Learning, היא טכניקה בענף למידת המכונה, שבה סוכנים נוקטים פעולות בתוך סביבה כדי למקסם את הרווח המצטבר כתוצאה מהפעולות הללו. סוכנים אלו נותנים חיווי "הצליח" או "לא הצליח".

למידה לא מונחת חלקית:
למידת לא מונחת חלקית, Semi-Unsupervised Learning, היא טכניקה בענף למידת המכונה, שבה קודם לומדים בצורה לא מונחת מכמויות נתונים גדולות ואח"כ מבצעים כיוונו בעזרת מעט נתונים מסווגים.

למידה מונחת:
מושגים בסיסיים בלמידת מכונה מונחת:
אלו הם המושגים הבסיסיים על מנת לנסח בעיה של למידת מכונה מונחת

- ✓ נתונים D
  אוסף של דוגמאות (נקודות, מיקרים, דגימות)
- ✓ נתוני וקטור הקלט $x = (x_1, x_2, \dots, x_n)$
  נתונים אלה כוללים מאפיינים (Features, Properties, Independent Variables)
- ✓ משתנה המטרה t (supervised):
  יכול להיות Target, Dependent variable, label

הגדרת הבעיה עבור למידה מונחת:
מתוך אוסף נתונים D המתויגים בעזרת פונקציה לא ידועה $t = f(x)$ הממפה מאפייני קלט x למשתנה פלט t, מעונינים למצוא היפותזה h המקרבת את f.

מעונינים כי ההיפותזה h תהיה מסוגלת **להכליל ולחזות** פלטים נכונים מתוך קלטים חדשים (שלא נכללו ב D).
את השגיאה שעושה החיזוי נמדוד בעזרת פונקציה loss או בעזרת מדדים אחרים מקובלים . נרצה למזער את השגיאה: למידה היא אופטימיזציה.

במילים אחרות, נתונים זוגות של קלט-פלט (קבוצת אימון), רוצים להסיק (ללמוד) פונקציה f דטרמיניסטית שממפה את הקלט לפלט, ולמידת המכונה מתבטאת באלגוריתמים לשערך את f בעזרת h
כך ש $f(x) \approx h(x)$ והפער ביניהם מינימלי. בנוסף קיימת שגיאה אינהרנטית $t = f(x) + \varepsilon$, שמקורה בטעויות מדידה , הקלט שאיננו מספיק עבור חישוב הפלט , אקראיות והרעש במערכות פיסיקליות וכו'.
בפרקטיקה, מניחים שהתוחלת של השגיאה ε מתאפסת וגם שε אינה תלויה סטטיסטית בקלט.

למידה כבעיית אופטימיזציה:
נתונה קבוצת אימון $D = \{\langle x, t\rangle\}$ וקבוצת מבחן $V = \{\langle x, t\rangle\}$.
רוצים ללמוד את הפונקציה h (מודל): $t = f(x) + \varepsilon \approx h(x)$ כך ששגיאת החיזוי $loss_v(h)$ על דוגמאות המבחן תהיה מזערית.

הערכת הפונקציה $f(x)$:
ניתן להעריך את הפונקציה בשיטות פרמטריות ולא פרמטריות.
שיטות פארמטריות: מניחים ש$h(x)$ היא פונקציה מסוימת הכוללת פארמטרים, ואלגוריתם הלמידה צריך רק למצוא את הפארמטרים של הפונקציה, פרוצדורת האימון מוצאת את הפארמטרים שימזערו את השגיאות שבין חיזוי הפונקציה ל t שבנתונים. למשל משקלים.

שיטות לא פארמטריות: משתמשות במדגם כדי לבנות את הפונקציה.

הגדרת פתרון עבור למידה מונחת:
אלגוריתם ML מיצר מודל שיודע לחשב הפלט y בהנתן קלט x. המודל מממש היפותזה $y = h(x)$.
ההיפוטזה נקראת גם מודל חיזוי.

במקום לתכנת את הפונקציה f שאיננה ידועה , נראה למכונה הרבה דוגמאות "אימון" שמתארות מה הפלט הנכון לקלט נתון. בתקווה שהמודל שיבנה, יקרב את הפונקציה הרצויה וישגה מעט גם על דוגמאות מבחן.

אם הלמידה "מספקת", המודל ידע למפות נכון את הדוגמאות, אך גם ל"הכליל" על קבוצת המבחן ולייצר פלטים נכונים גם לקלטים חדשים.

מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

אלגוריתם חיזוי= רגרסיה לינארית:

רגרסיה ליניארית היא מודל פרמטרי פשוט לשערוך $f(x)$ בעזרת משערך לינארי $h(x) = w_1x + w_0$. כלומר נשערך ישר / מישור לינארי שבקירוב טוב יהיה $f(x) \approx h(x)$.זוהי שיטה פרמטרית עבור למידה מונחת.

כדי לדעת האם הישר / המישור שנוצר הוא קרוב מספיק לנתונים האמיתיים, נגדיר:

- ✓ סכום ריבועי השגיאות= Residual Sum of Squares Errors :

$$\sum_{i=1}^{n}\left(t - h(x)\right)^2 = \sum_{i=1}^{n}\left(f(x) + \varepsilon - h(x)\right)^2$$

- ✓ שגיאה ריבועית ממוצעת= Mean Square Error : $\frac{1}{n}RSS$

נשים לב כי למזער את הRSS זהה למזעור SME שכן היחס ביניהם ישר.

<u>משתנה מסביר אחד:</u>

המקרה הפשוט ביותר הוא זה שבו קיימים שני משתנים: משתנה בלתי תלוי x ומשתנה תלוי y. לדוגמה, אפשר לנסות להסביר ולנבא באמצעות המודל את גובהו של עץ תפוחים (y במטרים), על פי משקלו של הזרע שממנו הוא צומח (x בגרמים).

כאמור, בבסיס השיטה עומדת ההנחה כי המודל המסביר את הקשר בין המשתנים הוא מודל ליניארי, כלומר, שמשוואה מסוג $y = ax + b + error$ תתאר נכונה את הקשר.

נתון אוסף נקודות $D = \{\langle x, t\rangle\}$. נרצה למצוא היפותזה לינארית (קו ישר) מדויק ככל האפשר. הדיוק מתבטא בכך שממזער את סכום ריבוע השגיאות. כלומר: נרצה למצוא שני פרמטרים: w_0, w_1 כך שהמשוואה $y = w_1x + w_0$ תהיה משוואה של קו ישר המתאים ביותר לאוסף הנקודות. במילים אחרות, מחפשים וקטור של משקולות (נקודה במרחב המשקולות) שתמזער את הפונקציה המוגדרת ע״י קבוצת האימון.

<u>רגרסיה מרובה:</u>

במקרים רבים מבקשים להסביר משתנה תלוי y באמצעות מספר משתנים בלתי תלויים x_i . לדוגמה, ייתכן שכדי להסביר את גובהו של עץ תפוח, יש להתחשב לא רק במשקל הזרע, אלא גם בכמות המשקעים השנתית במקום שבו הוא גדל, בגובהו של העץ שממנו הגיע הזרע, ובמליחות הקרקע.

במקרה זה, נרצה למצוא היפותזה לינארית שהיא מישור (או היפר מישור כתלות במספר המאפיינים).

נשים לב כי חריגים (outliers) מוגברים ברגרסיה ליניארית שכן השיטה מתבססת על חישוב סכומי ריבועי השגיאות, ומספר זה נובע ומושפע מהחריגים.

<u>פונקצית הMSE: מזעור השגיאה</u>

MSE שייך למשפחה של פונקציות loss הממוזערות ע״י אלגוריתמי למידה.

כאשר קבוצת הדוגמאות D נתונה:

- ✓ פונקצית הרגרסיה f נותנת ערך עבור כל וקטור משקולות ממימד n
- ✓ ה MSE היא פונקציה שמעבירה ממרחב המשקולות ממימד n+1 : $MSE: R^{n+1} \to R^+$

עבור רגרסיה לינארית במשתנה יחיד, MSE היא פרבולה דו מימדית (קערה):

$$MSE_{(w_0,w_1)} = \frac{1}{n}\sum_{i\in D}(t_i - y_i)^2 = \frac{1}{n}\sum_{i\in D}\left(t_i - (w_1x_i + w_0)\right)^2$$

מקרה פרטי של הפרבולה, הוא אם ישנה רק משקולת אחת, ואז הפרבולה היא חד מימדית.

מרצה: ד״ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

מזעור MSE:

כידוע, לפרבולה מינימום יחיד, ונקודת מינימום זו היא שתמזער את השגיאה של היפותזת המישור.
אפשר להציג את ההיפותזה של הרגרסיה הלינארית בהצגה מטריציונית:

כאשר ישנם n מאפיינים (features) $h(x) = w \cdot x^T = (w_0, w_1, \dots, w_n) \cdot \begin{pmatrix} 1 \\ x_1 \\ \vdots \\ x_n \end{pmatrix}$.

כאשר יש לנו קבוצת אימון של m דוגמאות, שבה השורות הם הדוגמאות, נוסיף עמודה ראשונה של אחדים.
אם נעשה שחלוף (transpose) נקבל מטריצה x^T שבה יש $n+1$ שורות ו m עמודות, כעת דוגמאות האימון מופיעות כעמודות.

המכפלה $w \cdot x^T$ נותנת וקטור שורה של תחזיות $y = h(x) = w \cdot x^T$.
לפעמים נוח לכתוב את הביאס בנפרד מוקטור המשקולות $y = h(x) = w \cdot x^T + b$.

מציאת המשקולות:

נשאלת השאלה האם קיימת נוסחה מפורשת אנליטית למשקולות ברגרסיה הלינארית.

שיטה 1 – המשוואה הנורמלית:

ניתן לחשב את המינימום בעזרת נוסחה סגורה ע"י איפוס הנגזרת החלקיות ופתירת מערכת משוואות לינאריות (הפיכת מטריצה גדולה). כלומר, בנקודת המינימום, המשקולות יהיו המינימליות.
השוואת הנגזרות החלקיות ל 0 מייצרת מערכת $n+1$ משוואות לינאריות, לכן קימת נוסחה סגורה באלגברה לינארית שמוצאת את נקודת המינימום (באמצעות הפיכת מטריצה גדולה).

עבור קבוצת אימון D המכילה m דוגמאות $(x_1, t_1), \dots, (x_m, t_m)$ בעלי n מאפיינים (features)
נחשב ניגזרת חלקית של פונקצית ה MSE עבור כל אחד מ n המשקולות, מקבלים n+1 משוואות לינאריות שניתן לפתור באמצעות מציאת מטריצה הופכית:

$$w = (x^T x)^{-1} x^T t$$

x הינה מטריצה שבה השורות הם הדוגמאות והעמודות הם המאפיינים (עמודה ראשונה היא 1).
בנקודת המינימום, n+1 הנגזרות החלקיות (לפי w) של פונקצית השגיאה מתאפסות.

פונקצית השגיאה: $\frac{1}{2}(wx^T-t)^2$, הגראדיינט מתאפס: $x^T(wx^T-t)=0$ ואז:
$x^T(xw^T-t)=0$ מכיוון ש: $xw^T=wx^T <= x^Txw^T-x^Tt=0 <= x^Txw^T=x^Tt <= w^T=(x^Tx)^{-1}x^Tt$

המשוואה הנורמלית דורשת כי כל הנתונים יהיו בזיכרון וגם הפיכה של מטריצה (n+1)x(n+1) בסיבוכיות $O(n^3)$. שיטה זו מדויקת אך מתאימה רק עבור מימדים קטנים.

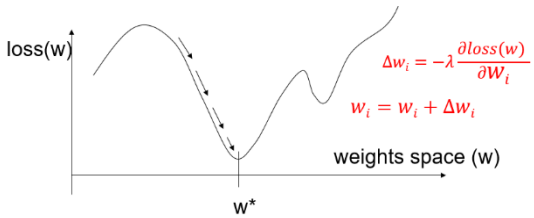
שיטה 2- אלגוריתם GD-Gradient Descent:

גרדיאנט הוא הכללה של מושג הנגזרת בעבור חשבון אינפיניטסימלי של מספר משתנים. הגרדיאנט הוא אופרטור וקטורי המופעל על שדה סקלרי, וקטור הנגזרות החלקיות:

$$\nabla \equiv \hat{x}\frac{\partial}{\partial x} + \hat{y}\frac{\partial}{\partial y} + \hat{z}\frac{\partial}{\partial z} \equiv (\partial_x \ \partial_y \ \partial_z)$$

ניתן לעשות שימוש בשיטת GD שעושה שימוש בחישוב הגרדיאנטים כדי למצוא את נקודת המינימום.
מתחילים מנקודה אקראית (של משקולות), בדוק על סמך השיפוע, לאיזה כיוון כדאי לרדת כך שתהיה ירידה הכי חזקה בעלות ובעזרת חישוב את הגראדיינט של פונקצית השגיאה בנקודה, ובצע צעד של שינוי משקולות שיביא את הנקודה החדשה קרוב יותר לנקודת המינימום. גודל הצעד נקבע על פי פארמטר קצב הלמידה λ.

מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין



Epoch = מונח המשמש בלמידת מכונה ומציין את המעברים שאלגוריתם הלמידה השלים על מערך הלמידה כולו (training set).

בהינתן

- ✓ פונקצית עלות $loss_{D,h}(w)$ שהגראדיינט שלה ניתן לחישוב
- ✓ קבוצות אימון D המכילה דוגמאות $\{\langle x_i, t_i \rangle\}$

רוצים למצוא ווקטור w שימזער: $Min_w\{loss_{D,h}(w)\}$ באופן הבא:

1. התחל מ $(w_0, w_1, \dots, w_n)$ משקולות אקראיים ובצע Epochs שוב ושוב עד להתקימותו של תנאי העצירה- למשל: עד כאשר loss מפסיק לרדת, או שהגענו למקסימום Epocs .
2. בכל Epoc: עבור D , חשב את הגראדיינט של פונקצית העלות בנקודה w ושנה את $(w_0, w_1, \dots, w_n)$ כך ש $loss((w_0, w_1, \dots, w_n))$ יקטן, חישוב השינוי יהיה צעד כנגד כיוון השיפוע (אם השיפוע שלילי נרצה להגדיל את המשקולות, אם השיפוע חיובי נרצה להקטין אותן). ככל שקבוע למידה קטן יותר למידה מדויקת יותר אבל אלגוריתם איטי יותר.

$$\Delta w_i = -\lambda \frac{\partial loss_{D,h}(\mathrm{w})}{\partial \mathrm{w_i}}$$

$$w_i = w_i + \Delta w_i$$

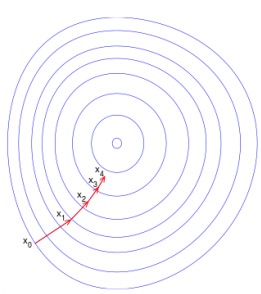
עבור המקרה הפרטי, הלינארי:

$$\Delta w_i = -\frac{\lambda}{m}\sum_{p\in D}(t_p - y_p)\frac{-\partial y_p}{\partial w_i} = \frac{\lambda}{m}\sum_{p\in D}(t_p - y_p)\frac{\partial yp}{\partial w_i} = \frac{\lambda}{m}\sum_{p\in D}(t_p - y_p)\, x_i$$

אלגוריתם זהו off line כיוון שמשתמשים בכל סט האימונים בכל epoch. ורסיה אחרת של האלגוריתם הזו הופכת אותו להיות on line ובו יש עדכון משקולות אחרי כל דוגמא. במקרה זה $\Delta w_i = \lambda(t_p - y_p)x_{pi}$.

בשיטת DG למיזעור $MSE_D(w)$ עבור היפותזה פארמטרית כללית $h_w(x)$ (שאינה בהכרח לינארית), פונקצית loss תיראה בצורה שונה:

$$lossD,h(w) = \frac{1}{2}MSE_D(w) = \frac{1}{2m}\sum_{p=1}^{m}\left(t_p - h_w(x_p)\right)^2$$



אלגוריתם זה הוא גם נקרא Steepest Descent היות ופונקצית העלות MSE עבור רגרסיה לינארית במימד 1 היא פרבולה דו-מימדית (קערה) בעלת נקודת מינימום אחת. ההתקדמות של וקטור המשקולות היא בניצב לקווי הגובה של הפרבולה. ניתן להוכיח כי הניגזרת הכיוונית: מקסימלית בכיוון המאונך לקו הגובה, אורטוגונלית עם קו הגובה. מסלול steepest descent ב GD, מוביל אותנו כמעט במאונך למיקום המינימום. ההתקדמות היא איטית.

מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

נרמול שדות:
פונקצית העלות MSE עבור רגרסיה לינארית במימד 1 היא פרבולה דו-מימדית (קערה).
נקודת מינימום אחת. ההתקדמות היא בניצב לקווי הגובה של הפרבולה, אבל, מה קורה כאשר הקערה איננה סימטרית?

תופעה זו קורה: כאשר למאפיינים features יש התפלגות שונה וטווח ערכים שונה למשל מספר חדרים בדירה 2-6 וגודל הדירה 50-200 מ"ר.

על מנת לטפל בתופעה זו, נרצה לנרמל שדות. נרמול = הסבה של ערכי השדה כך שיהיה לכולם אותו טווח (ולפעמים גם אותו ממוצע וסטית תקן). נרצה לנרמל את ה features (שדות) כך שערכם יהיה בסדר גודל דומה.
למשל בין 0 ל 1 או בין [-1,1] לחילופין: עם ממוצע 0 ו/או עם סטית תקן 1.

דרכי נירמול שונות:
1. Min,max Scaling: יעביר ל [0,1].
   נמיר את ערכי שדות לערכים מאותו סדר גודל: (x-Min)/range ה Range הוא טווח הערכים (במידה וכולם חיוביים), למשל (room#-2)/(8-2)
2. Mean-Normalization: נמיר את ערכי השדות כך שיתנו ממוצע 0, (X-mean(X))
3. SD-Scaling: נקבל סטיית תקן 1. נחלק בסטיית התקן: (X/sd(X))
   הערך החדש של המאפיין Feature משקף את המרחק שלו מהממוצע בסטיות תקן

איזו שיטה עדיפה? שיטת GD או normal equation? עבור ביג דאטה נבחר בGD ולעומת זאת, עבור סט אימונים קטן המשוואה הנורמלית תהיה מדויקת וקלה לחישוב ומהירה.

הרחבות למודל:
ניתן להגדיל את הגמישות של המודל הלינארי ולשלוט עליו. למשל בעזרת:

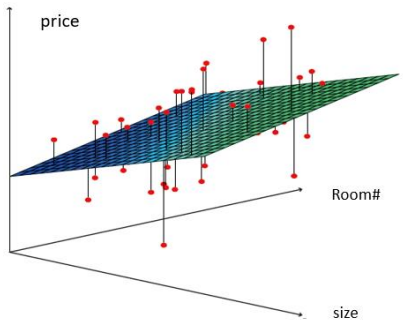
רגרסיה פולינומיאלית:
הוספת מאפיינים שהם טרנספורמציות של מאפיינים קיימים.
כאשר ההיפותזה חלשה מידי, נראה הרבה שגיאות בזמן האימון. נוכל לנסות:
א. טרנספורמציות על מאפיין בודד = למשל חישוב גיל מתאריך לידה.
ב. טרנטפורמציות מורכבות = מכפלות של מאפיינים (למשל שטח מתוך אורך ורוחב), מכפלות משולשות של מאפיינים, מספר טרנספורמציות של מאפיינים בודדים.
ישנו חופש ליצור מאפיינים features חדשים המבטאים תלויות בין מאפיינים מקוריים ולהכניס אותם לנוסחת הרגרסיה. על ידי הוספה זו, הופכים את מרחב הקלט למרחב גדול יותר. במרחב החדש, יש סיכוי שגם להיפותזיות לינאריות תהיה שגיאה קטנה יותר.

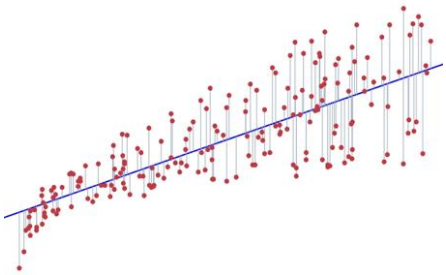
התאמת ההיפותזה = נוסיף מאפיינים לפי דרגת הפולינום שבו מתעניינים להיפותזה (ניתן להתאים פולינומים מכל דרגה) על ידי הכפלה של מאפיין קיים בעצמו. הרגרסיה נשארת לינארית גם אם הפולינום מדרגה גבוהה יותר, היות והקלט הגולמי נשאר כפי שהיה, אבל את הרגרסיה מבצעים במרחב מאפיינים רחב יותר.

נשים לב כי יכולה להיות התפוצצות של מספר המאפיינים כאשר מימד הקלט הוא גדול.
היפותזה לינארית $y = w_0 + w_1 x$ ואילו היפותזה פולינומאלית $y = w_0 + w_1 x + w_2 x^2$ בעזרת הוספה של מאפיין חדש שהוא ריבוע של מאפיין קיים.

מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

תוצאה ויזואלית של המודל:





מרצה: ד״ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

אלגוריתם קלסיפיקציה = רגרסיה לוגיסטית:

סיווג classification = הפלט הרצוי t הוא תווית או קטגוריה. במקרה הפשוט: פלט אחד ובינארי= כן או לא.
במקרים אחרים (רב קטגוריות): זיהוי פרצופים, זהוי המילה הבאה בטקסט.

רגרסיה לוגיסטית היא שיטה פרמטרית עבור למידה מונחת והיא מודל סטטיסטי המתאר קשר אפשרי בין משתנה קטגורי תלוי לבין המשתנים הבלתי תלויים שהם יכולים להיות איכותיים או כמותיים.

המודל מאפשר לאמוד את מידת ההשפעה של שינוי בערכו של כל אחד מהמשתנים הבלתי תלוים על ערכו של המשתנה התלוי. במילים אחרות, המודל מאפשר לאמוד מתאמים בין המשתנים הבלתי תלויים למשתנה המוסבר.

המודל לבדו אינו מספיק כדי לקבוע קשר סיבתי בין המשתנים המסבירים והמשתנה המוסבר.

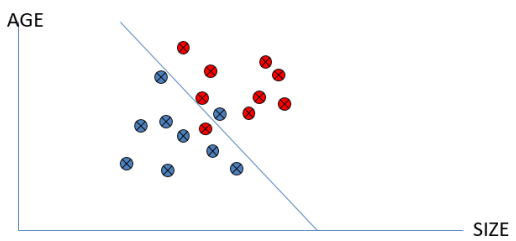
ההיפותזה:

בהינתן קבוצת: $D = \{\langle x, t\rangle\}$. רוצים ללמוד את הפונקציה: $t = f(x) + \varepsilon$ ע"י מציאת היפותזה (קירוב) $y = h(x)$ שממזער את פונקצית $loss$. ההיפותזה המתקבלת תיצור גבול הפרדה או מישור הפרדה בין נתונים השייכים לקטגוריות שונות. t נבחר מקבוצת ערכים דיסקרטיות (קטגוריות).

בהניתן פונקצית $loss_D(w)$ והיפותזה החוזה הסתברות $y = p(t = 1|x)$, מחפשים משקולות w שימזערו את $loss_D(w)$.

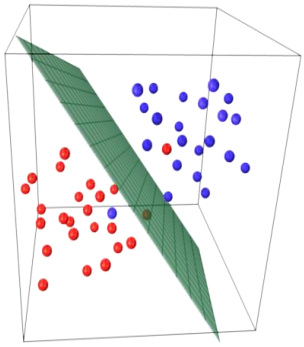
דוגמא – קלסיפיקציה בינארית בשני מימדים:

מעוניינים לחזות אם גידול הוא שפיר או ממאיר כתלות בגודל הגידול ובגיל הנבדק.
גבול ההחלטה בדו מימד הוא קו שכל נקודותיו על הישר $w_0 + w_1x_1 + w_2x_2 = 0$.



$$p(t = 1|x) \approx$$
$$y = g(z) = \frac{1}{1 + e^{-z}}$$
$$z = w_0 + w_1x_1 + w_2\,x_2$$

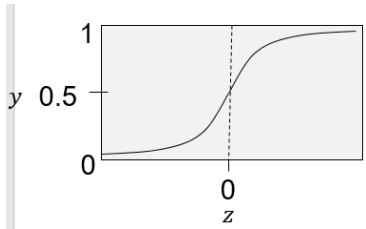
בשלושה מימדים, גבול ההחלטה הינו מישור.
בפראקטיקה קיימים הרבה מאפיינים (features) שהם מימדים שיכולים לעזור בסיווג.



$$p(t = 1|x) \approx$$
$$y = g(z) = \frac{1}{1 + e^{-z}}$$
$$z = w_0 + w_1x_1 + w_2\,x_2 + w_3x_3$$

מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

היפותזה עבור רגרסיה לוגיסטית היא מהצורה $y = g(z) = \frac{1}{1+e^{-z}}$, $z = w_0 + \sum_i w_i x_i$.
היפותזה זו משמשת לקלסיפיקציה בינארית ומחזירה הסתברויות. נשים לב שהפונקציה $g(z)$ היא סיגמואיד.



המודל פועל באופן הבא:

- ✓ אימון: בהמלך אימון המודל מחפשים משקולות של מישור מפריד. ככל שהאימון יהיה טוב יותר החיזוי יהיה מדויק יותר.
- ✓ חיזוי: חישוב ההסתברות של נקודה להיות בקטגוריה כלשהיא על פי מרחקה מהמישור המפריד. ההסתברות על הקו היא חצי, ומתחיו ומעליו פחות ויותר בהתאמה.

המרחק בין נקודה x' למישור המפריד $wx = 0$ הוא:

$$\frac{wx'}{||w||} = \frac{z}{||w||} \sim z$$

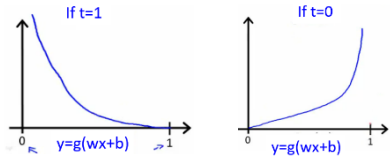
כשהנורמה שואפת ל-1 המרחק שואף ל-z .

<u>פונקציית השגיאה ברגרסיה לוגיסטית:</u>
לא רצוי להשתמש בפונקצית MSE ללמידת האלגוריתם שכן הפונקציה איננה קמורה ובשונה מרגרסיה לינארית כאן, נרצה קמירות. לכן, הפונקציה שבה משתמשים היא Cross Entropy . זוהי פונקציה קמורה, ללא נקודות מינימום מקומיות.
עבור תבנית בודדת $\{\langle x, t\rangle\}$: כאשר המצב החזוי הוא $y = g(z)$:

$$C(y,t) = -t \cdot \log(y) - (1-t)\log(1-y)$$

נשים לב כי

$$C(y,t) = \begin{cases} -\log(y) & if\ t = 1 \\ -\log(1-y) & if\ t = 0 \end{cases}$$



שגיאת Cross Entropy עבור קבוצת אימון הכוללת m דוגמאות:

$$lossD(w) = \frac{1}{m} \sum_{p=1\ldots m} C(y_p, t_p)$$

כדי למקסם את יעילותו של האלגוריתם נרצה למזער את השגיאה ולקבל את המשקולות בהתאם. לשם כך נרצה למצוא את המשקולות עבורן המישור (או הישר) יהיה מדויק כמה שיותר.
לשם כך, נבצע נגזרות:

$$z = wx, y = h(x) = g(z) = \frac{1}{1+e^{-wx}}, \frac{dy}{dz} = y(1-y), \frac{\partial y}{\partial w_i} = y(1-y)x_i$$

נרצה למעזר את פונקציה זו כדי לקבל את המישור הכי מתאים לנתונים שאפשר. זוהי פונקציה קמורה ללא נקודות מינימום מקומיות:

$$\frac{\partial CE_D}{\partial wi} = 1/m \sum_{i \in D} (y_p - t_p) x_i$$

מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

כלל עדכון המשקולות = מעזור השגיאה:
ניתן להשתמש בשיטת GD על מנת למצוא את המשקולות, ועידכון המשקולות יהיה באופן הבא:

$$\Delta w_i = \lambda/m \sum_{i \in D} (t_p - y_p) xi$$

$$\Delta w_0 = \lambda/m \sum_{i \in D} (t_p - y_p) 1$$

<u>קלסיפיקציה של יותר משתי קטגוריות:</u>
ניתן להעזר בקלסיפיקציה בינארית עבור יותר משתי קטגוריות באופן הבא:

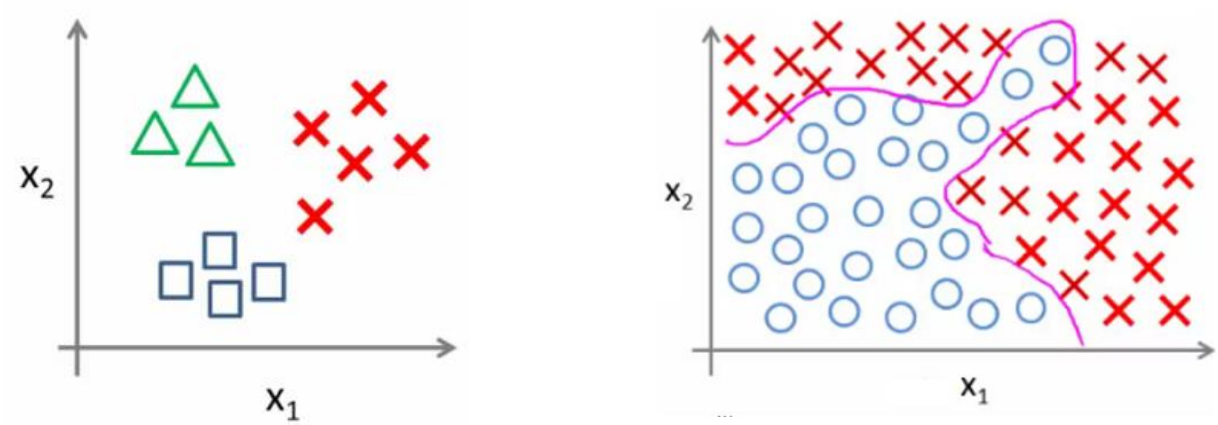
1. גישת one vs rest = נבנה מסווג לכל קטגוריה, כלומר נעביר קו חוצה בינה לבין הקטגוריות האחרות סה"כ k מסווגים. אימון: נלמד להפריד קטגוריה אחת מכל השאר. חיזוי: בהינתן דוגמה לסיווג נבדוק תחזית של כל מסווג ונחזיר את הקטגוריה שלה חוזה הסתברות הגבוהה ביותר.
2. גישת one vs one = נבנה מסווג לכל זוג קטגוריות, כלומר נעביר קו חוצה בין כל זוג קטגוריות. סה"כ $C(k,2)$ מסווגים. אימון: נלמד להפריד קטגוריה אחת מכל השאר. חיזוי: מספר אפשרויות. אחת תהיה הצבעת הרוב, אחרת תהיה חישוב הסתברות ממוצעת לכל קטגוריה ובחירת קטגוריה שמקבלת את ההסתברות המקסימלית.

פונקצית השגיאה:

$$MCCE_D(y,t) = -\frac{1}{m} \sum_p \sum_i^k t_{pi} \log(y_{pi}) + (1 - t_{pi}) \log(1 - y_{pi})$$

ניתן לראות שכאשר יש הרבה קטגוריות, חישוב הפונקציה לא יעיל במיוחד.

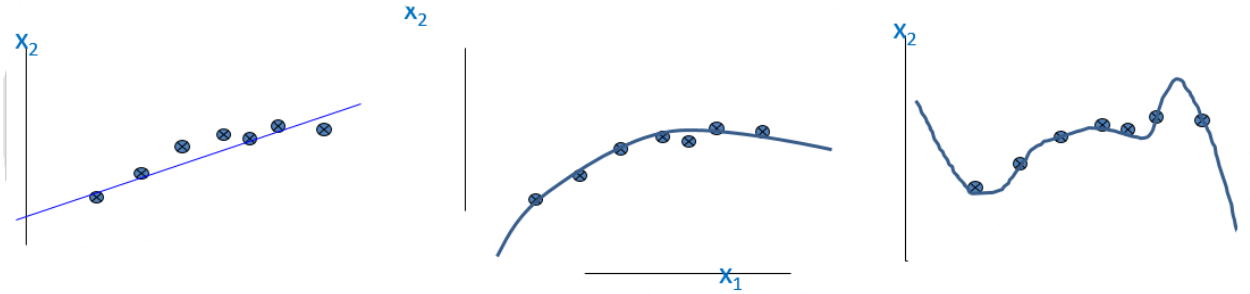
תוצאה ויזואלית של המודל:

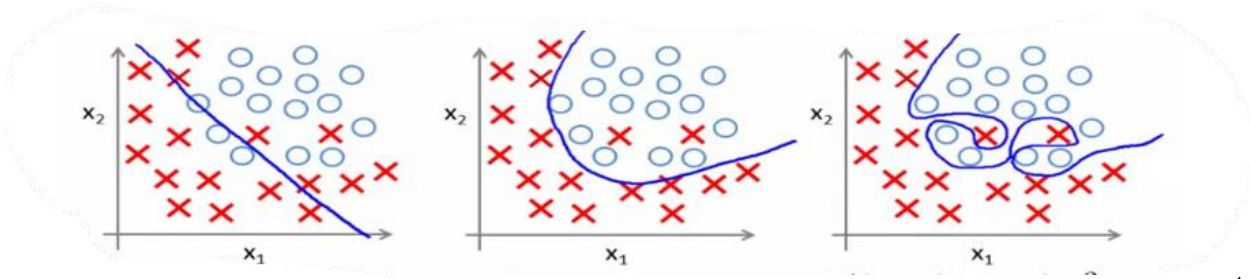


מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

<u>גמישות המודלים:</u>
מודל יכול להיות גמיש יותר או פחות.

במודל רגרסיה (חיזוי), הגמישות מתבטאת בדרגת הפולינום $h(x) = w_0 + w_1x_1 + \cdots$.
ככל שדרגת הפולינום גבוהה יותר, הוא גמיש (מפותל) יותר.





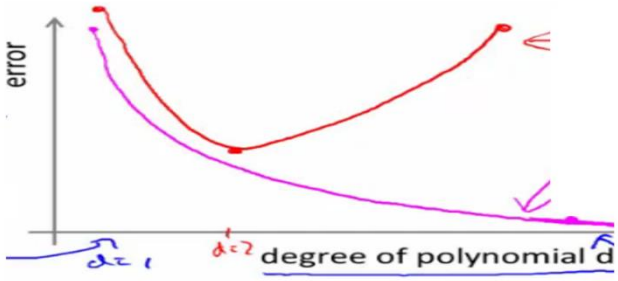
לשם כך נגדיר מושגי גמישות:

1. התאמה לא מספקת under fitting . אין התאמה מספיק טובה בין המודל הנוצר לבין התוויות הנתונות. דבר שכיח ונפוץ מאוד ברגרסיה לינארית. ניתן לקרוא לשגיאה high bias .
2. תאמה יתר על המידה over fitting . התאמה טובה, אף טובה מידי בין המודל הנוצר לבין התוויות הנתונות. דבר שכיח ונפוץ כאשר דרגת הפולינום במודל רגרסיה עולה. ניתם לקרוא לשגיאה variance.

<u>לא נשמש במודל הגמיש ביותר:</u>
יש מספר סיבות שבגללן לא משתמשים במודל הגמיש ביותר שניתן למצוא:

- ✓ גמישות יתר עשויה לגרום להתאמת יתר (שגיאת הכללה)
- ✓ גמישות דורשת יותר דוגמאות אימון (לא תמיד קיימות)
- ✓ מודל גמיש יהיה "כבד" יותר לחישוב
- ✓ מודלים גמישים בד"כ קשים לפרשנות (Interpretability)

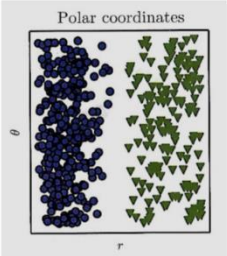
אם נסתכל שוב על המודלים (רגרסיה או קלסיפיקציה): ככל שדרגת הפולינום עולה, שגיאת האימון יורד (עקב over fitting) , ולעומת זאת, שגיאת המבחן יורדת עד שמגיעה להיות אופטימלית (מינימלית) ואז היא מתחילה לעלות עקב אותה התאמת יתר.

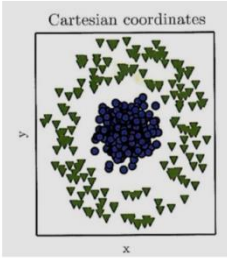


מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

כמו ברגרסיה, גם בקלסיפיקציה – גבול הפרדה לינארי עלול לגרום לשגיאות התאמה. פתרון פשוט לבעיה זו הוא הוספה טכנית של מאפיינים פולינומיאלים (התפוצצות קומבינטורית) או הסבת מאפיינים בצורה חכמה שיכולה לעזור, למשל:

- ✓ הפכית קורדינטות קרטזיות לפולריות. כך סט נתונים שלא היה ניתן לבצע לו הפרדה לינארית, כעת כבר אפשר





שימוש בטרנספורמציות לקלטים והגדלת מימד הקלט על ידי הוספת מאפיינים חדשים מאפשר גבולות החלטה שאינם לינארים ומסובכים.

הוספת יותר מידי מאפיינים יכולה לייצר גמישות יתר והתפוצצות מימדים.

<u>אומנות הוספת המאפיינים:</u>

כמה מאפיינים להוסיף? אילו מאפיינים להוסיף? אין תשובה חד משמעית, אבל באופן כללי נוכל לומר,

- ✓ הטרנספורמציות הטובות תלויות בנתונים
- ✓ המהנדסים צריכים להכיר היטב את הנתונים ולהיות יצירתיים (למרות שישנן טרנספורמציות פופולריות Kernels)
- ✓ Feature Engineering זה תהליך יקר אך הוא לעיתים קרובות נחוץ (בעיקר כשאין הרבה דאטה)

<u>היפותזות מורכבות וגמישות יותר:</u>

למדנו רגרסיה לינארית ולוגיסטית שמשתמשות במודלים לינארים פשוטים שאינם גמישים במיוחד (אלא אם כן מוסיפים מאפיינים). ניתן להשתמש במודלים לא לינארים ומורכבים מאוד.

רשתות נוירונים הם מודלים כאלו שמורכבתם גדלה ככל שעומק הרשת גדל. ברשתות, היפותזות פארמטריות מסובכות באמצעות הרכבה של הרבה פונקציות לא לינאריות ממושקלות. למשל, שימוש בפונקציה הלוגיסטית מספר רב של פעמים (רשת של סיגמואידים).

<u>שימוש במאפיינים טובים:</u>

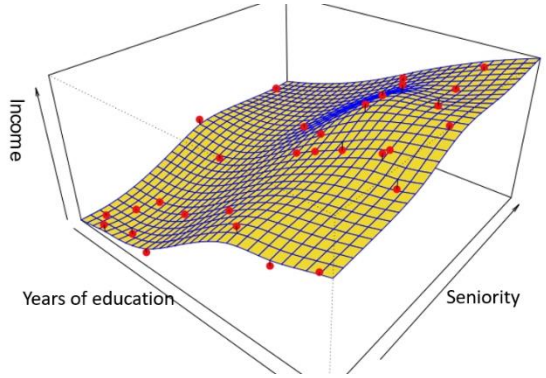
שימוש במאפיינים שהם טובים הוא המפתח להצלחה:

- ✓ טרנספורמציות מסוימות לקלט, משפרות את ביצועי המערכת הלומדת אך מציאתם באמצעים ידניים (מהנדסים מוכשרים) היא קשה ויקרה
- ✓ רשתות נוירונים עמוקות: הם היפותזות פרמטריות מורכבות, ניתן להסתכל עליהם כעל פונקציות שמבצעות טרנספורמציות על קלט גולמי ומיצרות מאפיינים חדשים בצורה היררכית- אם יש הרבה דאטה, יש סיכוי שרשת נוירונים תמצא התמרות "טובות" יותר ממהנדסים כישרוניים
- ✓ Kernels: טרנספורמציות אוטומטיות המעבירות את הקלט למרחב מאפיינם ממימד גדול יותר, שבו ניתן לבצע הפרדה לינארית ביתר קלות. קל ונוח להשתמש בהם כשאין יכולת לעשות "הינדוס מאפינים" יקר וידני

<u>רגרסיה בעזרת Thin Plate Spline:</u>

זו היא שיטת רגרסיה שאינה פרמטרית. הדגימות הן אלו שקובעות את הבליטות, ולא היפותזה קבועה.

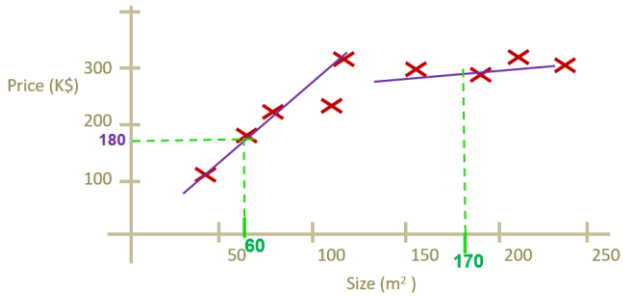
כאמור, המודל הכי פשוט (ולא גמיש) הוא הלינארי. לעומת זאת, לספלינים של לוח דק יש דרגות גמישות שונות על פי ההתנגדות של הלוח הדק (פזיקה).



מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

שליטה על גמישות רגרסיה לינארית:

על מנת לשלוט בגמישו המודל הלינארי, ניתן להשתמש בשיטת Locally Weighted Linear Regression .
לפי שיטה זו, לא נבנה מודל בזמן האימון, אלא רק בעת השאילתה (query). זוהי שיטה שהיא שילוב של שיטה לא פארמטרית ביחד עם רגרסיה לינארית.



בהינתן שאילתה (וקטור מאפיינים x), נבנה מודל רגרסיה, אך ההשפעה של הנתונים השונים תהיה שונה: הנקודות בסט הנתונים D שקרובות יותר לוקטור x ישפיעו יותר על מודל הרגרסיה מאשר נקודות רחוקות יותר.

אימון:

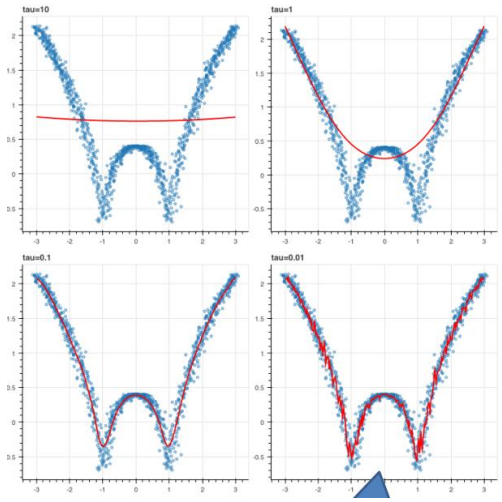
✓ שומרים בזיכרון דוגמאות ומשתמשים בהם בכל פעם שזקוקים לחיזוי.

חיזוי:

✓ בהינתן וקטור x מחשבים משקולות [0-1] לדוגמאות האימון על פי מרחקיהם מהוקטור x כאשר דוגמה רחוקה- משקלה יתקרב לאפס, ואילו דוגמה קרובה- משקלה יתקרב לאחד.

חישוב המרחק לפי פונקציה גאוסיאנית מסביב לנקודה $\beta_i = e^{\frac{-\|x_i - x\|^2}{2\tau^2}}$. נשים לב כי ככל שהקבוע τ קטן יותר והמרחקים הם גדולים, אז המשקלים המתקבלים הם משקלי אפס. לעומת זאת, ככל שהקבוע τ גדול יותר נלקחות בחשבון גם נקודות רחוקות יותר. פונקציית המרחק מודדת גם אי דמיון.

✓ מבצעים רגרסיה עם פונקציית MSE ממושקלת: $WMSE_w = \frac{1}{2m}\sum_i \beta_i (wx_i - t_i)^2$



בעזרת פונקצית המרחק ניתן לשלוט על הגמישות של המודל. ככל שהקבוע τ קטן יותר המשקל של נקודות שהן מרוחקות שואף לאפס ואילו נקודה קרובה משפיעה יותר. ככל שהקבוע τ גדול יותר, כל נקודות האימון משפיעות באופן דומה ומקבלים רגרסיה לינארית שהיא רגילה.

מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

הקשר של פונקצית השגיאה MSE לבין bias , variance :
למידה היא בעיית אופטימיזיה, מתוך מרחב ההיפותזות, נרצה למצוא היפותזה ה"מוצלחת ביותר" שנותנת שגיאה מינימלית על קבוצת הבדיקה (מבלי שנותנים ללמוד מקבוצת הבדיקה).

השגיאה (ב MSE) מורכבת מריבוע שגיאת ביאס, שגיאת ווריאנס ושגיאה אינהרנטית.

- ✓ הביאס הוא תוחלת ההפרשים בין החיזוי (התלוי ב D) לתוצאה הרצויה. ככל שההיפותזה מתגמשת ומתאימה עצמה ל D, הביאס יתקרב ל 0. היפותזות שאינן מאפשרות התאמה טובה ל D יתנו שגיאת ביאס גבוהה.
- ✓ השונות הוא השונות בחיזויים של ההיפותזה על פני כל המדגמים. התאמת יתר של ההיפותזה למדגם, תיגרום לוואריאנס גבוה בין ההיפוזות ולכן שגיאת ווריאנס גבוהה

פירוק מתמטי של פונקצית השגיאה לגורמים bias , variance:
השגיאה שיש למודל על נתוני המבחן מורכבת משלושה סוגי שגיאה :

$$loss = (Bias)^2 + Variance + irreducible$$

על שגיאת irreducible לא ניתן לשלוט. ואילו שאר השגיאות תלויות באלגוריתם הלמידה, ויש קשר ביניהן :

- ✓ ככל שהמודל קשיח יותר (לא גמיש), הוא עלול לא לקרב באופן הדוק את f (אותה אנחנו מחפשים או רוצים לדמות בעזרת ההיפותזה). אם כך, תוחלת ההפרש בין הפונקציה האמיתית f למודל החיזוי לא תהיה שווה לאפס. זוהי שגיאת bias .
- ✓ ככל שהמודל גמיש יותר, הוא עלול להתאים את עצמו לתבניות לא משמעותיות בנתוני האימון – השונות של החיזויים של המודלים נובעת מהתאמת יתר של המודל לדגימות האימון. זוהי שגיאת variance.

כאמור, תוחלת ריבועי שגיאת החיזוי ניתנת לפירוק. נוכיח :

בהינתן :

- ✓ D קבוצת אימון (מדגם)= משתנה מיקרי מטריציוני - אוסף דוגמאות מתוך התפלגות לא ידועה
- ✓ x הקלט= משתנה מיקרי וקטורי שאינו תלוי במדגם D מתוך התפלגות לא ידועה
- ✓ f פונקציה דטרמיניסטית= אותה רוצים לשערך בעזרת h (איננה משתנה מיקרי)
- ✓ ε רעש= משתנה מיקרי שאין לנו שליטה בו, בעל תוחלת אפס ואין תלות בינו לבין הקלט
- ✓ t ערך המטרה= משתנה מיקרי התלוי בקלט x וכולל שגיאה בלתי נמנעת ε
  ניתן להגדיר : $t_x = f(x) + \varepsilon$
- ✓ y תוצאת החיזוי= משתנה מיקרי התלוי במדגם ובדוגמא $y = h_D(x) \approx f(x)$

פונקצית השגיאה MSE התיאורתית של אלגוריתם שלומד מתוך D היא התוחלת של ריבועי השגיאה על כל המדגמים D וכל הקלטים x $MSE = E_x E_D[(h_D(x) - t_x)^2]$. ניתן לקצר
. $MSE = E[(h_D(x) - t_x)^2]$

שלב 1 = נראה כי $MSE = E[(h_D(x) - f(x))^2] + Varriance(\varepsilon)$:
להלן ההוכחה :

$$MSE = E_x E_D\left[\left(t_x - h_D(x)\right)^2\right] = E\left[\left(f(x) + \varepsilon - h_D(x)\right)^2\right] =$$
$$= E\left[\left(f(x) - h_D(x)\right)^2 + 2\left(f(x) - h_D(x)\right)\varepsilon + (\varepsilon - 0)^2\right] =$$
$$= E\left[\left(f(x) - h_D(x)\right)^2\right] + 2E[f(x) - h_D(x)]E[\varepsilon] + E[(\varepsilon - 0)^2] =$$
$$= E\left[\left(f(x) - h_D(x)\right)^2\right] + E[(\varepsilon - 0)^2] = E[(h_D(x) - f(x))^2] + Varriance(\varepsilon)$$

מרצה : ד"ר גדי פנקס
נכתב על ידי : יעלי בר גימלשטיין

כלומר, פונקציית השגיאה MSE מתפרקת לשני גורמים:
שגיאה שלא ניתן להפחית אותה, irreducible error , $Varriance(\varepsilon)$, שהיא הרעש של איסוף הנתונים. וכן, שגיאה שניתן להפחית אותה, reducible error, $E[(h_D(x)-f(x))^2]$, שהיא תוחלת ריבועי הפרשים בין הפונקציה להיפותזה.

שלב 2= נראה כי השגיאה הניתנת להפחתה מתפרקת לסכום של שני סוגי שגיאה, הרצויים:
נגדיר: פונקציית תוחלת החיזויים $\overline{h_D(x)} = E[h_D(x)]$ ואז:

$$E\left[\left(h_D(x)-f(x)\right)^2\right] = E\left[\left(h_D(x)-\overline{h_D(x)}+\overline{h_D(x)}-f(x)\right)^2\right] =$$

$$= E\left[\left(h_D(x)-\overline{h_D(x)}\right)^2 + 2\left(h_D(x)-\overline{h_D(x)}\right)\cdot\left(\overline{h_D(x)}-f(x)\right) + \left(\overline{h_D(x)}-f(x)\right)^2\right] =$$

נשים לב כי $\overline{h_D(x)}-f(x)$ היא פונקציה דטרמיניסטית, ותוחלת של משתנה מקרי בפונקציה דטרמיניטית היא מכפלת התוחלת בפונקציה.

$$= E\left[\left(h_D(x)-\overline{h_D(x)}\right)^2\right] + 2E\left[\left(h_D(x)-\overline{h_D(x)}\right)\cdot\left(\overline{h_D(x)}-f(x)\right)\right] + E\left[\left(\overline{h_D(x)}-f(x)\right)^2\right] =$$

תוחלת ההפרשים של משתנה מקרי מהתוחלת שלו היא אפס.

$$= E\left[\left(h_D(x)-\overline{h_D(x)}\right)^2\right] + E\left[\left(\overline{h_D(x)}-f(x)\right)^2\right] = variance[h_D(x)] + (Bias[h,f])^2$$

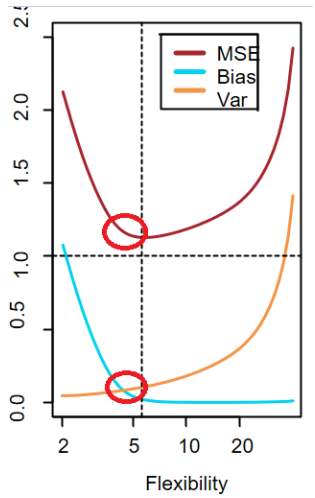
כלומר, בסה"כ, פונקציית השגיאה MSE מתפרקת:

1. שגיאה שלא ניתן להפחית אותה, irreducible error , $Varriance(\varepsilon)$, שהיא הרעש של איסוף הנתונים.
2. שגיאה שניתן להפחית אותה, reducible error, $E[(h_D(x)-f(x))^2]$, שהיא תוחלת ריבועי הפרשים בין הפונקציה להיפותזה. את השגיאה הזו ניתן לפרק אפילו יותר:
   a. $variance[h_D(x)]$
   b. $(Bias[h,f])^2$

ניסוח מתמטי של הגורמים bias , variance:
כפי שנראה בהוכחה, נגדיר באופן מילולי ומתמטי את שני המושגים:

1. הביאס= תוחלת ההפרשים של ההיפותזה והפונקציה $Bias[h,f] = E\left[\left(\overline{h_D(x)}-f(x)\right)^2\right]$,
   כלומר, מדד הביאס מציין עד כמה ההיפותזה שונה מהפונקציה אותה אנחנו מחפשים. אם ההיפותזה לא דומה כלל לפונקציה אותה מחפשים נקבל under fitting.
2. השונות = תוחלת ההפרשים של ההיפותזה לבין התוחלת שלה $= variance[h_D(x)]$
   $E\left[\left(h_D(x)-\overline{h_D(x)}\right)^2\right]$ כלומר, מדד השונות מציין עד כמה רחבה ומגוונת ההיפותזה יכולה להיות.
   כמובן שהשונות תלויה במרחב המדגם, היות ויכולים להיות מדגמים שונים. אם ההיפותזה מותאמת יותר מידי למדגם מסוים נקבל over fitting.

מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין



פשרת The bias – variance tradeoff:

ככל שמעלים גמישות של מודל הביאס יורד והשונות עולה.בפרט, פונקצית MSE עולה או יורדת בתלות לקצב הירידה בביאס וקצב העליה בשונות.

בד"כ בהתחלה, קצב הירידה בביאס מהיר מקצב העליה בשונות ולכן יש ירידה התחלתית בשגיאה ואחכ עליה, ונרצה לבחור את האיזור בו השגיאה היא מינימלית.

כאשר המודל אינו מספיק גמיש ושגיאת Bias גבוהה: משלמים למהנדסים "יצירתיים" כדי שיפיקו טרנספרמציות למאפיינים "טובים" , או שמשתמשים במודלים גמישים ובטכניקות לביצוע טרנספורמציות אוטומטיות (LWLR, עצים, רשתות נוירונים, Kernels).

כאשר המודל גמיש מידי ושגיאת השונות גבוהה: קורה כאשר מימד הקלט גבוה או שהוספנו יותר מידי מאפיינים. ניתן לתיקון על ידי השקעה כספית: קונים חומרה חזקה , משלמים כדי לקבל יותר נתונים מסווגים או משתמשים בטכניקות לצמצום מספר המימדים (למשל feature selection, PCA) או משתמשים בטכניקות לצמצום שגיאת ה variance: מורידים גמישות המודל, משתמשים בטכניקות לרגולריזציה, או מוסיפים (או מחוללים) נתונים.

מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

הצלחת המודלים:
נרצה להעריך את מידת ההצלחה של המודל הלומד.
KPI's = Key Performance Indicators = מדד ניתן לכימות המשמש להערכת הצלחת ארגון, עובד בעמידה ביעדים לביצועים.

מידת ההצלחת של קלסיפיקציה בינארית:
על מנת לדבר על הצלחה של סיווג שהוא בינארי, נרצה להבין אילו טעויות רווחות במודל מסוג זה:

1. שגיאה מסוג 1 = type 1 error = false positive : במילים פשוטות, שגיאה זו מתארת אזעקת שווא. למשל, מעולם הרפואה יהיו אלה בריאים שסווגו כחולים או גבר שנאמר לו שהוא בהריון.
2. שגיאה מסוג 2= type 2 error = false negative : במילים פשוטות, שגיאה זו מתארת פיספוסים. למשל, מעולם הרפואה יהיו אלה חולים שסווגו כבריאים או אישה הריונית שנאמר לה שהיא לא בהריון.

למדנו פונקציות עלות (שגיאה) MSE, CEntropy, פונקציות שימושיות, קמורות (עם פונקצית אקטיבציה נכונה) אבל לא אינטואיטיביות לבני אדם.....משתמשים/רופאים/כלכלנים – עולם האפליקציה.

עבור משתמשים אלו (עולם האפליקציה) עדיף למדוד ביצועים בדרכים אחרות, KPI's:

sensitivity, recall, hit rate, or true positive rate (TPR)	$\frac{TP}{TP+FN} = 1 - FNR$
specificity, selectivity or true negative rate (TNR)	$\frac{TN}{TN+FP} = 1 - FPR$
precision or positive predictive value (PPV)	$\frac{TP}{TP+FP} = 1 - FDR$
negative predictive value (NPV)	$\frac{TN}{TN+FN} = 1 - FOR$
miss rate or false negative rate (FNR)	$\frac{FN}{FN+TP} = 1 - TPR$
fall-out or false positive rate (FPR)	$\frac{FP}{FP+TN} = 1 - TNR$
false discovery rate (FDR)	$\frac{FP}{FP+TP} = 1 - PPV$
false omission rate (FOR)	$\frac{FN}{FN+TN} = 1 - NPV$
Positive likelihood ratio (LR+)	$\frac{TPR}{FPR}$
Negative likelihood ratio (LR-)	$\frac{FNR}{TNR}$

הדיוק, Accuracy משמש כמדד סטטיסטי לעד כמה מבחן סיווג בינארי מזהה או מוציא תנאי בצורה נכונה.
כלומר, הדיוק הוא חלקן של התחזיות הנכונות (הן חיוביות אמיתיות והן שליליות אמיתיות) מתוך סך המקרים שנבדקו $\frac{TP+TN}{TP+TN+FP+FN}$.

לעומתו, הסיווג השגוי, Misclassifications משמש כמדד סטטיסטי לעד כמה מבחן סיווג בינארי מזהה שגוי או מוציא תנאי בצורה לא נכונה $1 - Accuracy$.

מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

מטריצת בלבול = confusion matrix :

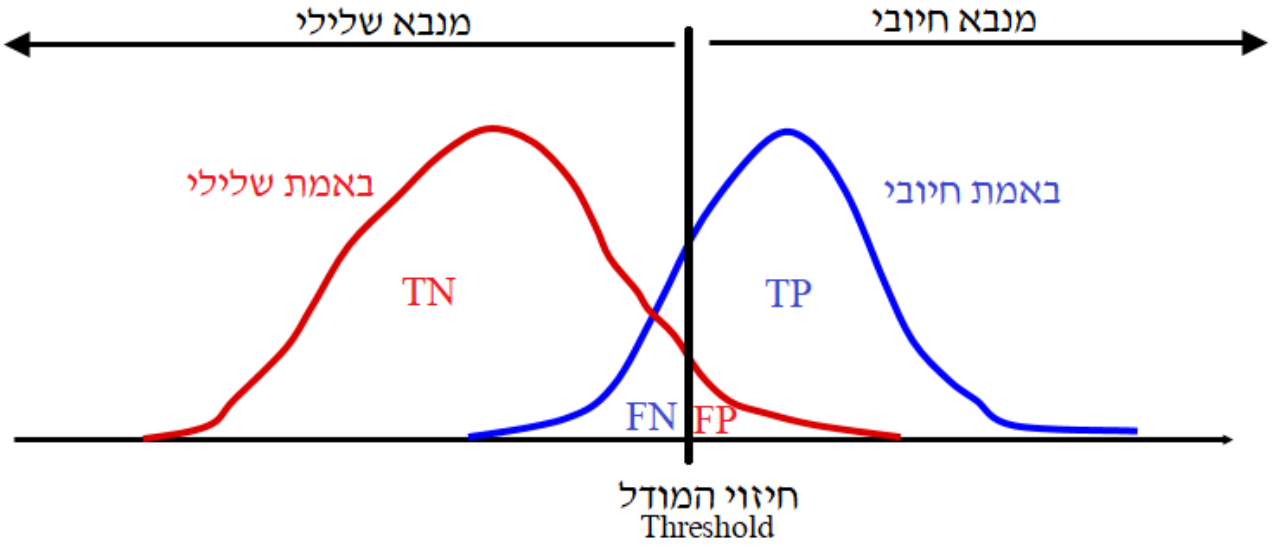
מטריצה שבא נעזרים כדי להציג את כל הנתונים הנדרשים לחישוב המדדים שתוארו מעלה

Confusion matrix		החיזויים (y)	
		חיזוי חיובי	חיזוי שלילי
הסיווג האמיתי (t)	באמת חיובי	*True Positive*	*False Negative* שגיאה מסוג 2
	באמת שלילי	*False Positive* שגיאה מסוג 1	*True Negative*

כעת, ניתן להוסיף את המדדים השונים ליצוגם בטבלה :

Confusion matrix		החיזויים (y)		
		חיזוי חיובי	חיזוי שלילי	
הסיווג האמיתי (t)	באמת חיובי	*True Positive*	*False Negative* שגיאה מסוג 2	$TPR\ or\ recall = \frac{TP}{P}$
	באמת שלילי	*False Positive* שגיאה מסוג 1	*True Negative*	$TNR = \frac{TN}{N}$
		$PPV\ or\ Precision = \frac{TP}{Predicted\ P}$	$NPV = \frac{TN}{Predicted\ N}$	$Accuracy = \frac{TP + TN}{ALL\ Examples}$

ניתן להציג גם בעזרת גרף :



מרצה : ד״ר גדי פנקס
נכתב על ידי : יעלי בר גימלשטיין

פשרת precision VS recall:
בקלסיפקציה מחשבים הסתברות אך ההחלטה כיצד לסווג תלויה בסף threshold. למשל: במיקרים רבים מחליטים לסווג לקטגוריה, אם הסיכוי גדול מ 0.5. נוכל לשנות את הסף וכך לשנות את הפרופורציות של Precision, Recall בהתאם להשפעתם הכלכלית.

נוכל לצייר גרף שמראה כיצד משתנים הדיוק (precision) והאיחזור (recall) כפונקציה של הסף. כאשר הסף נמוך, מדד recall משתפר ואילו הדיוק יותר ולהפך.

מטריצת השגיאה כאשר חלק מהקטגוריות נדירות – מדד F1 score:
במקרים בהם אחת מהקטגוריות נדירה (למשל הסיכוי לחלות במחלה קטן מאוד), מדד $Accuracy$ לבדו לא יהיה מדויק מספיק, זאת כיוון שכנראה אחד המדדים הדיוק (precision) או האיחזור (recall) יהיה שווה לאפס ומדד $Accuracy$ לא מראה זאת. ואילו אנו מעוניינים למקסם את מדד הדיוק והאיחזור.

לכן, נעזר בממד חדש, $F1\ score = 2 \cdot \frac{P \cdot R}{P+R}$ שהוא ממוצע הרמוני של שני המדדים. ככל שהמדד $F1\ score$ יותר גבוה משמע שגם הדיוק וגם האיחזור גבוהים.
נהוג שמדד $F1\ score$ משמש גם לבדיקה של אלגוריתמי למידה שונים.

הכללה- F measure:
ניתן להכליל ולהסתכל על המדד F measure . מדד זה הוא ממוצע הרמוני של מדדי הדיוק והאיחזור

$$F - Measure = \frac{1}{\alpha \cdot \frac{1}{Precision} + (1-\alpha) \cdot \frac{1}{Recall}}$$

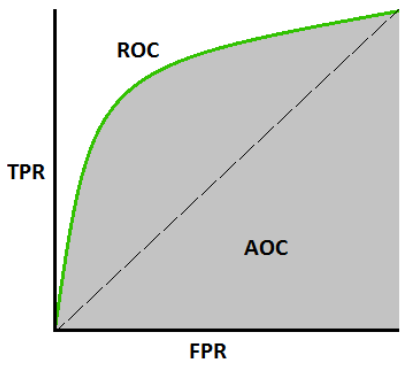
כאשר α מייצג את המשקל המועדף לכל מדד. אם α גדול מחצי, יועדף מדד הדיוק ואילו אם α קטן מחצי, יועדף מדד האיחזור.
מקרה פרטי של F measure הוא $F1\ score$, כאשר F measure מאוזן, כלומר $\alpha = 0.5$.

כדי לחשב איזה משקל α נדרש לכל חישוב (למשל בחישובי כלכלה) = $\alpha = \frac{W_{fp}}{W_{fp}+W_{fn}} = \frac{cost\ for\ fp}{cost\ to\ make\ mistake}$
כלומר הדיוק תלוי בכמה עולה לחברה לטעות.

מדד balanced accuracy למודל שאינו מאוזן:
מדד זה מהווה שיטה נוספת לתת מספר אחד המשערך ביצועי קלסיפייר כאשר המבחן איננו מאוזן.
Balanced accuracy הוא בעצם ממוצע (רגיל) של המדד האיחזור recall של כל קטגוריה.
מדד זה רלוונטי קלספיקציה בינארית וגם מרובה. כאשר הטסט מאוזן, נקבל את ה Accuracy .

בקלסיפיקציה בינארית, זהו הממוצע בין האיחזור (recall) לסגוליות (specificity).
דוגמא:
- ✓ בקטגוריה A: מתוך 100 חולים סווגו נכון 90 חולים כלומר, $recall = 0.9$
- ✓ בקטגוריה B: מתוך 200 חולים סווגו נכון 160 חולים כלומר, $recall = 0.8$

ואז ניתן לראות את ההבדל בחישוב בין שני המדדים:
- ✓ $Accuracy = \frac{90+160}{300} = 0.833$
- ✓ $Balanced\ Accuracy = \frac{0.8+0.9}{2} = 0.85$



גרף Receiver Operating Characteristics =ROC25 :
זהו גרף שמתאר את הקשר בין TPR לבין FPR כתלות בסף.

במסווגים שהם טובים העקומה תהיה מוטת שמאלה ומעלה, כלומר כמה שיותר true positive rate וכמה שפחות false positive rate .

ניתן ללמוד את טיב המסווג באמצעות AOC = Area Under the Curve , כלומר השטח שמתחת לגרף, כאשר השטח המקסימלי יהיה 1. עבור מסווג שמסווג רנדומלי נקבל $AOC = 0.5$, קו מרוסק. נשתמש בשיטה זו כדי להשוות בין מסווגים שונים.

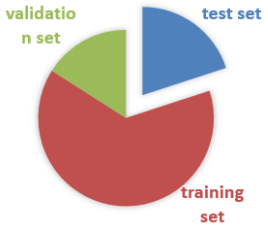
מידת ההצלחת של קלסיפיקציה רבת קטגוריות:
נוכל ליצור מטריצת בלבול גם עבור קלסיפיקציה שאיננה בינארית ונוכל להשוות בין המסווגים השונים.
האופן בו נבצע זאת יהיה התאמה של המטריצה. גודל המטריצה יהיה כמספר הקטגוריות. אם ישנן n קטגוריות אז גודל המטריצה יהיה $n \times n$. ואז יהיה ניתן לחשב את כל המדדים שנלמדו על כל קטגוריה בנפרד.

אין שיטה מקובלת לשקלל מדדים עבור כל המודל. אפשר למשל:
- ✓ למצע את ה AUC של K הקטגוריות
- ✓ לחשב ממוצע ללא שיקלול של הרגישות של כל קטגוריה (או לשקלל בעלות הכלכלית)

שיערוך מודל בעזרת ואלידציה:
בעת כתיבת ולמידת המודל נרצה לדעת: מתי לעצור את האימון? איזה קצב למידה להשתמש? באיזה גודל mini batch להשתמש? איזו דרגת גמישות (למשל דרגת פולינום) להשתמש כהיפותזה?לאלגוריתמי למידה יתכנו הרבה היפר-פארמטרים שונים ומשונים....

היינו רוצים לשערך את הביצועים הצפויים מהמודל (loss, F1 score, Precision, Recall , B. Accuracy) שיצרנו לכשירוץ על טסט אמיתי ואז לבחור במודל ובהיפר הפארמטרים שיתנו ביצוים משוערכים הכי טובים.

השיטה הפשוטה והמקובלת לשיערוך מודלים היא ולאידציה, כלומר להקצות חלק מקבוצת האימון לואלידציה. כלומר נבצע חלוקה למידע ברשותנו באופן הבא:

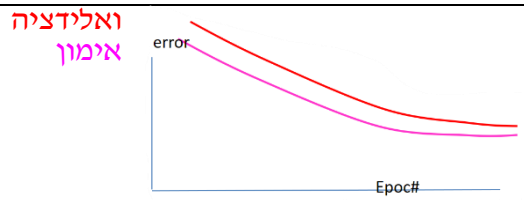
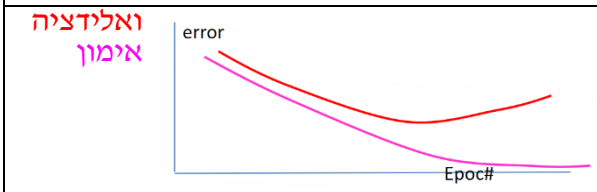


- ✓ מידע אימון, training data = קבוצת אימון ללמידת המשקולות
- ✓ מידע תיקוף, validation data = "מבחן בית", נשתמש בקבוצת תיקוף כדי לבחור ולשערך את ההיפותזה, הפרמטרים ועוד
- ✓ מידע מבחן, test data = "מבחן גמר", נשתמש בקבוצת מבחן כדי להעריך את המודל במבחן אמיתי, נצפה שהערכה זו תהיה קירוב מוצלח ליכולו החיזוי במציאות

שימושים של ולידציה:
לשיטת הולידציה שימושים רבים, ביניהם:
1. קריטריון עצירה = השגיאה באלגוריתם GD אמורה לרדת מונוטונית כל איטרציה, אבל מהו תנאי העצירה? היינו רוצים לעשות כאשר השגיאה על האימון נמוכה אבל אז נקבל over fitted model ומצד שני אסור להשתמש במידע המבחן. לכן, קריטריון עצירה יהיה כאשר שגיאת הואלידציה נמוכה ואינה משתפרת אחרי עוד אימונים.
2. בחירת קצב למידה = אם רואים שהשגיאה עולה בהתמדה או אינה יורדת, יתכן וקצב הלמידה גבוה מידי וכל עדכון באלגוריתם GD זורק לכיוון השני לנקודה גבוהה יותר. נרצה להוריד את קצב הלמידה. מצד שני, אם קצב הלמידה נמוך מידי, ההתקדמות תהיה איטית מידי. קצב למידה אופטימלי יהיה הגדול ביותר שאינו מרע משמעותית את השגיאה של הואלידציה.

מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

נוכל להתבונן בגרפים של שגיאת האימון לעומת שגיאת הואלידציה:

Under fitted	Over fitted
 שגיאת האימון גבוהה יחסית ואינה ממשיכה לרדת.	 שגיאת האימון יורדת מונוטונית ואילו שגיאת ההכללה יורדת בהתחלה ואז גדלה

באופן נורמלי: שגיאת ההכללה (וולידציה או טסט) תהיה קצת גבוהה משגיאת האימון.
בהרבה מיקרים נוכל לאבחן מצבי over/under fitting באמצעות השוואת שגיאת האימון לשגיאת הוולידציה.

<u>חסרונות בשיטת הואלידציה:</u>

- ✓ בחירה רנדומלית של קבוצת וולידציה קטנה מידי עלולה לא לשקף את המבחן האמיתי, הערכת חסר.
  קבוצת ולידציה קטנה מידי עלולה לא לשקף את הטסט. סכנת התאמת יתר: בחירת ההיפר-פארמטרים עלולה לגרום למודל להיות מותאם מידי לוולידציה . שיערוך השגיאה הצפויה עלול להיות מוטה לקבוצת הוולידציה
- ✓ בחירה רנדומלית של קבוצת וולידציה גדולה מידי עלולה לא לשקף את המבחן האמיתי, הערכת יתר.
  קבוצת האימון קטנה יותר, יש פחות דוגמאות אימון והמודל יהיה פחות טוב ממודל שיואמן על כל הדוגמאות.

<u>שיטות ואלידציה מתוחכמות:</u>
וולידציה קטנה ואקראית, סביר שלא תאפשר לנו לשערך נכון את השגיאה וכך גם כיוונון ההיפר פארמטרים לא יהיה אפקטיבי, מצד שני וולידציה גדולה, לא תשאיר לנו דוגמאות אימון

נלמד שיטות אימות צולב (Cross Validation) מתוחכמות יותר.

מאפייני CV:

- ✓ נרצה הערכת שגיאה אמינה יותר
- ✓ נרצה להתאמן על כל הנתונים הקיימים (לא בהכרח נרצה לוותר על הנתונים שבקבוצת הוולידציה לצורך אימון)

General Cross Validation =
את קבוצת האימון נחלק להרבה (k) קבוצות שונות של זוגות אימון (D-v) + וולידציה (v). נשים לב שמספר הזוגות האפשריים הוא גודל קבוצת החזקה של מספר הדוגמאות הקיימות עבור גודל ולידציה שאינו קבוע או שמספר הזוגות האפשריים הוא $C(m,v)$ כאשר ישנן m וגודל הולידציה הוא v וזה יהיה מספר המודלים שיחושבו.

תהליך אימון:

1. אמן כל אחד מהזוגות בנפרד.
2. צור מודל
3. חשב את שגיאת הוולידציה לכל זוג
4. הערכת השגיאה: מצע את שגיאות הוולידציה (MSE, CE, F1,P,R,Accuracy…)

$$CVloss = \frac{1}{k}\sum_{v} loss_v$$

בחירת המודל עבור המבחן:

1. ניתן לבנות את המודל הסופי מקבוצת האימון כולה (נהמר שהערכת השגיאה שעשינו רק על חלק מהנתונים תקיפה גם למודל שאומן על יותר נתונים)
2. נשתמש באנסמבל: נבחר ב k המודלים עם שגיאת וולידציה הנמוכה ביותר ונמצע את התוצאות (לרגרסיה) או נסווג על פי החלטת הרוב (קלסיפיקציה) או נבנה מטה-מודל שחוזה על פי k החיזויים של המודלים

לא מעשי לבצע מספר רב של וולידציות ולכן,ישנן שיטות וולידציה חסכוניות יותר.

מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

K-Fold Cross Validation =

תהליך האימון:

1. חלק את הנתונים המתויגים ל K קבוצות
2. שמור 1/K מהנתונים כקבוצת ולידציה: אמן על (K-1)/K מהנתונים ובדוק על הוולידציה
3. בצע K אימונים שונים בכל פעם על קבוצת וולידציה אחרת וקבל K שיערוכים שונים לשגיאה (validation loss, F1….)

בחירת המודל עבור המבחן:

4. מצע את k השיערוכים כדי לקבל הערכה לשגיאת הטסט

למשל, תהליך האימון עבור 10-fold CV

- ✓ נחלק את ה training ל 10 קבוצות: נאמן על 9 ונשתמש בעשירית כ validation.
- ✓ נמשיך כך 10 פעמים, כשבכל פעם בוחרים קבוצת validation אחרת
- ✓ כמובן, כתוצר לוואי ניתן לבנות קלסיפייר שהוא אנסמבל של k קלסיפיירים שונים

סה"כ ישנם k מודלים שיחושבו.

Leave K Out Validation =

תהליך האימון:

1. בהינתן קבוצת דוגמאות מתויגות D בחר קבוצת וולידציה של k דוגמאות כאשר יש v כאלה, אמן על D-v הדוגמאות שנותרו. מספר האימונים יהיה C(m, k) כאשר m=D

בחירת המודל עבור המבחן:

2. שערך את השגיאה ע"י מיצוע של השגיאות על כל הדוגמאות ב D.

סה"כ ישנם C(m, k) מודלים שיחושבו.

Leave One Out Cross validation =

תהליך האימון:

1. בחר דוגמא אחת בלבד כוולידציה, אמן על m-1 הדוגמאות שנותרו
2. עשה זאת עבור כל אחת מדוגמאות האימון ובדוק את המודל על דוגמת הוולידציה שנבחרה.

בחירת המודל עבור המבחן:

3. שערך את השגיאה ע"י מיצוע של השגיאות על כל הדוגמאות ב D.

סה"כ ישנם m מודלים שיחושבו (כמספר הדוגמאות).

Boot Strap Method=

יכולות החיזוי והערכת השגיאה היו משתפרות אם היינו יכולים לייצר כמויות גדולות של נתונים מתויגים שאיתם היינו מייצרים הרבה קבוצות אימון. אבל, ברוב המקרים אין נתונים רבים.

בד"כ לא ריאלי לייצר נתונים חדשים יש מאיין, אבל בשיטה הזו, מיצרים הרבה קבוצות אמון מקבוצת אימון אחת שאותה דוגמים רנדומית עם חזרות:

לדוגמא, בהינתן סט נתונים של 100 דוגמאות, נרצה ליצור קבוצת אימון בגודל 200 דוגמאות

- ✓ באופן רנדומלי, נבחר דוגמא מתוך סט הדוגמאות הקיים ונכניס אותה אל קבוצת האימון. כך נעשה 200 פעמים
- ✓ קיים סיכוי $\frac{1}{3}$ שחלק מהדוגמאות בסט לא יופיעו בקבוצת האימון, אותן ניקח כולידציה
- ✓ במידה ובחרנו יותר מקבוצת אימון אחת, נבצע ממוצע על הולידציה שיצרנו

מסתבר שיכולות השערוך משתפרות ככל שניצור עוד מידגמים ואפילו שלא נוסיף שום דוגמה חדשה.

נוכיח כי לא כל הדוגמאות יופיעו בקבוצת האימון, וחלקן אכן יהיו בולדיציה:

- ✓ בהינתן סט דוגמאות בגודל N , הסיכוי של דוגמה להיבחר לקבוצת האימון הוא $\frac{1}{N}$ והסיכוי שלה לא להיבחר הוא $1-\frac{1}{N}$
- ✓ נרצה לדגום קבוצת אימון גם כן בגודל N
- ✓ מכיוון שאנחנו דוגמים N פעמים, אז הסיכוי של דוגמא לא להיבחר בכל הדגימות היא $\left(1-\frac{1}{N}\right)^N$ לפי נוסחה של משתנה גאומטרי (שסופר מספר ניסויים עד להצלחה) בהסתברות
- ✓ בהשאפה לאינסוף נקבל $\lim_{n\to\infty}\left(1-\frac{1}{n}\right)^n=\frac{1}{e}\approx 0.368$ לפי גבול אוילר

הקטנת שגיאת השונות:
הדוגמאות בקבוצת האימון כוללות מידע על תבניות וחוקים, אבל גם כוללות רעש. עקב כך עלולה להיווצר טעות דגימה = sampling error שזה בעצם זיהוי שגוי של תבניות רנדומליות שנובעות רק מהבחירה של הדוגמאות לקבוצת האימון.

כאשר מתאימים מודל, אין דרך לזהות מי מהתבניות היא תבנית "אמיתית" ומי תבנית שגויה שנוצרה בגלל טעות דגימה. אם המודל מאוד גמיש, הוא יכול למדל במדויק את התבניות השגויות ונרצה להימנע מכך.

זוהי כאמור שגיאת השונות, variance error . נרצה להימנע מהתאמת יתר ונלמד שיטות "הכבדה" על אלגוריתם הלמידה שיעזרו בכך.

<u>בחירת פרמטרים feature selection</u>:
לפי עקרון התער על אוקאם, " אין להעמיד ריבוי ללא צורך" , כלומר העדפת ההסבר הפשוט יותר על פני המורכב. במקרה הזה, העדפה למודלים בעלי פחות פארמטרים.

דרך מקובלת לפשט את ההיפותזות היא להוריד מימדים באמצעות בחירת מאפיינים. יתרונותיה:
1. הפחתה של שגיאת השונות (מטרה)
2. יותר קל להסביר מודל בעל פחות מימדים
3. מודל בעל פחות מימדים מהיר יותר לחישוב

ברמת העל, בהינתן n מאפיינים, D סט דוגמאות, A אלגוריתם למידה , אלגוריתם בחירת פרמטרים יחזיר את המאפיינים הטובים ביותר.

נכיר מספר שיטות של בחירת מאפיינים.

השיטה האקזוסטיבית=
השיטה המעייפת, שבודקת את כל האפשרויות.
1. לכל תת קבוצה V של מאפיינים, בנה מודל בעזרת אלגוריתם הלמידה A ובדוק את שיערוך השגיאה שנותן המודל cross validation
2. בחר את תת הקבוצה V שעבורה שיערוך השגיאה הוא המינימלי ביותר

יווצרו 2^n מודלים, וכן 2^n שיערוכי שגיאה (כמספר תתי קבוצות האפשריים לבדיקה).

יוריסטיקה כללית טובה יותר=
לכל $k=1\dots n$ מבצעים $C(n,k)$ אימונים כדי לבחור תת קבוצה של k מאפינים שנותנת באימון את שגיאת האימון הקטנה ביותר. ואז כדי לבחור במספר המאפיינים הטוב ביותר k, מבצעים CV לכל $k=1\dots n$.
יווצרו 2^n מודלים, אבל רק n שיערוכי שגיאה cross validation.

=Forward feature selection
שיטה חמדנית מבוססת על היוריסטיקה הכללית.
ניצור n קבוצות של מאפיינים כאשר גודל של כל קבוצה הוא $k = 1, 2, \dots, n$. בכל פעם נחפש את המאפיין בעל השגיאה הטובה (הקטנה) ביותר ונוסיף אותו לקבוצות המאפיינים הרצויים.
בסוף נבחר בין כל הקבוצות את זו שיוצרת שגיאת שיערוך מינימלית.

1. Start with empty feature set
2. For $k = 1, \dots, n$:
    a. For all features f_k not used by model:
        i. Try adding each of the missing features, train and save the loss
        ii. Create model M_k by adding the feature that provides the best loss
3. Choose the best $M_k$using cross validation

יווצרו n^2 מודלים (סדרה חשבונית), n שיערוכי שגיאה cross validation.

=Backwards feature selection
שיטה חמדנית מבוססת על היוריסטיקה הכללית.
ניצור n קבוצות של מאפיינים כאשר גודל של כל קבוצה הוא $k = 1, 2, \dots, n$. בכל פעם נחפש את המאפיין בעל השגיאה הרעה (הגדולה) ביותר ונוריד אותו מקבוצות המאפיינים הרצויים.
בסוף נבחר בין כל הקבוצות את זו שיוצרת שגיאת שיערוך מינימלית.

1. Start with full feature set
2. For $k = 1, \dots, n$:
    a. For all features f_k not used by model:
        i. Try removing each of the missing features, train and save the loss
        ii. Create model M_k by removing the feature that provides the best loss
3. Choose the best $M_k$using cross validation

יווצרו n^2 מודלים (סדרה חשבונית), n שיערוכי שגיאה cross validation.

=Hybrid feature selection
שיטה חמדנית מבוססת על היוריסטיקה הכללית.
ניצור n קבוצות של מאפיינים כאשר גודל של כל קבוצה הוא $k = 1, 2, \dots, n$. בכל פעם נחפש את המאפיין בעל השגיאה הטובה (הקטנה) ביותר ונוסיף אותו לקבוצות המאפיינים הרצויים. לאחר מכן, נחפש את המאפיין בעל השגיאה הרעה (הגדולה) ביותר ונוריד אותו מקבוצות המאפיינים הרצויים.
בסוף נבחר בין כל הקבוצות את זו שיוצרת שגיאת שיערוך מינימלית.
נועד לפצות על החמדנות של השיטות הקודמות.

1. Start with full feature set
2. For $k = 1, \dots, n$:
    a. add a feature in Forward selection
    b. remove a feature in Backward selection
    c. save M_k
3. Choose the best $M_k$using cross validation

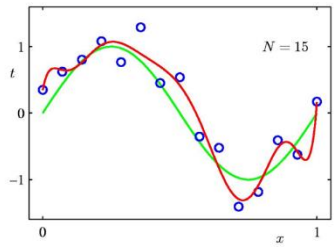
יווצרו n^2 מודלים (סדרה חשבונית), n שיערוכי שגיאה cross validation.

מרצה : ד״ר גדי פנקס
נכתב על ידי : יעלי בר גימלשטיין

רגולריזציה:
הענשת משקולות.
מטרת השיטה, כאמור, היא להקטין את שגיאת השונות high variance על ידי העקרונות הבאים:

- ✓ מקדמי פולינום קטנים אינם מאפשרים פיתולים עזים ולכן מונעים התאמת יתר
- ✓ כמקרה פרטי, אם המקדים מגיעים ממש לאפס, שיטה זו שקולה לשיטת בחירת מאפיינים

הענשת משקולות משמשת בד"כ אגוריתמי למידה פאראמטריים וגורמת להחלקת ההיפותזות.
הרעיון הוא שאיננו יודעים על איזה משקולת לוותר ולכן נענשים את כולן (חוץ ממשוקלת הביאס). את משקולת הביאס לא נהוג להעניש כי יש לה תפקיד עקרוני = למשל, ברגרסיה הוא הערך החיזוי כאשר הקלטים 0 והוא ממוצע התחזיות כאשר כל הקלטים מנורמלים סביב ממוצע 0. ברגרסיה לוגיסטית הוא קובע את ממוצע ההסתברות כאשר הקלטים מנורמלים סביב ממוצע 0.



גם אם נשתמש בפולינום מדרגה גבוהה, ע"י הענשת משקולות, נוכל ל"בקש" ממנגנון הלמידה ל"השתדל" לא להקצות מקדמים גדולים ונקבל מודל חיזוי חלק יותר. למשל בתמונה, פולינום מדרגה 15 עם ובלי עונשים

נגדיר γ קבוע רגולריזציה, ונוסחת שגיאה חדשה, מעונשת. ככל שמגדילים את העונש, המשקולות קטנות.
נשים לב כי הביאס עולה עם החמרת העונש ואילו השונות קטנה (הגיוני כי מורידים מהגמישות של המודל).

נכיר שתי שיטות של הענשת משקולות.

= Ridge regularization (L2)
נעניש את הפונקציית השגיאה על ידי הוספת הריבוע של הנורמה מסדר שני (שהם ריבועים):

$$\|w\|_2 = \sqrt{\sum_{i=1}^{n} w_i^2}$$

ואז, נוסחת השגיאה המעונשת תהיה $regMSE(w) = MSE(w) + \frac{\gamma}{2}\sum_{j=1}^{n} w_j^2$.
אם כך, כלל עדכון המשקולות משתנה בהתאם (גזירה):

$$\Delta w = -\lambda \frac{\partial loss}{\partial w_i} - \gamma w_i$$

חשוב לציין כי שיטה זו היא רגישה למדד של המאפיינים, ולכן עדיף לבצע נורמליזציה לנתונים.
חשוב לעשות למאפיינים Scale normalization ע"י חלוקה בסטית תקן. סטיית התקן לאחר הנירמול היא 1 לכל המאפינים שנורמלו. כאשר סטיית התקן זהה בכל המאפינים, גם המשקולות תהינה באותו סדר גודל
והעונש γ ישפיע באותה מידה. דוגמא: המשקולות של - גיל בשנים ומשקל בגרמים הם בסדרי גודל שונים. לא הגיוני להעניש אותם במידה שווה.

יתרונות שיטה זו על פני בחירת מאפיינים:

1. יעילות חישובית
2. מאפיינים בעלי תרומה קטנה, לא יסולקו בהכרח אלא ישתתפו בחיזוי (משקולות קטנים)
3. עמידות טובה יותר לרעשים: רעש בהרבה מאפיינים מתמצע וכן מופחת
4. ניתן להרחבה לכל אלגוריתם למידה פארמטרי (למשל רשתות נוירונים)

החסרון בשיטה זו הוא שנשארים הרבה מאפיינים וקשה יותר להסביר מודל כזה.

מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

= Lasso regularization (L1)

לפעמים יש יתרון ברגורליזציה המתבססת על ערכים מוחלטים (במקום על ריבועי המשקולות).

$$\|w\|_1 = \sqrt{\sum_{i=1}^{n} |w_i|}$$

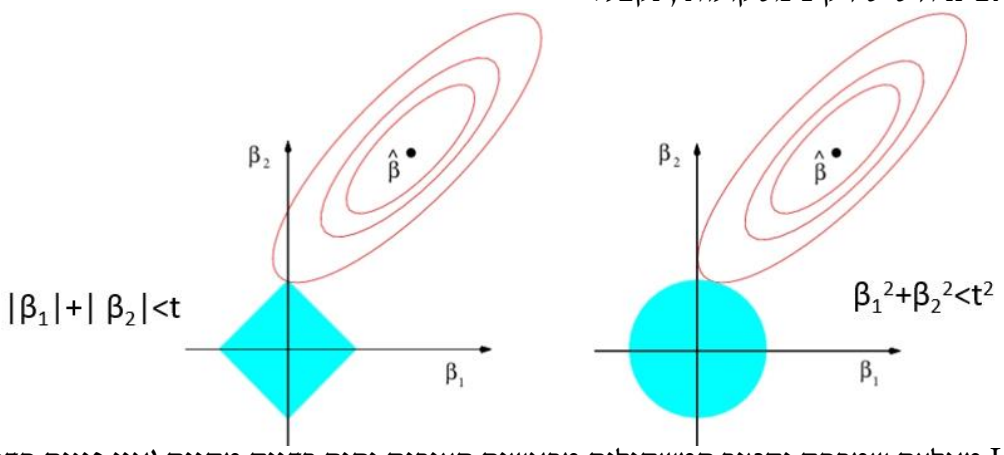
ואז, נוסחת השגיאה המעונשת תהיה $regMSE(w) = MSE(w) + \gamma \sum_{j=1}^{n} |w_j|$.

אם כך, כלל עדכון המשקולות משתנה בהתאם (גזירה):

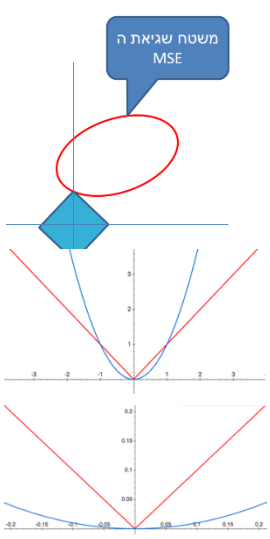
$$\Delta w = -\lambda \frac{\partial loss}{\partial w_i} - \gamma$$

שיטה זו מסוגלת לבצע בחירת מאפיינים. ככל שנגדיל את ה"עונש" משתנים יעלמו (יקבלו משקל אפס).
בשיטת Ridge המשתנים יקבלו משקולות נמוכים אבל לא יעלמו לגמרי.

נשים לב כי מזעור $regLoss$ זה כמו למצוא את המינימום של השגיאה כפוף לאילוץ שנורמה קטנה מגודל מסוים. אם נניח שיש רק 2 משקולות , נקבל:



נורמה L2 מאלצת שמרחק וקטור המשקולות מראשית הצירים יהיה רדיוס מסוים (אין פינות חדות) ואילו הנורמה L1 יוצרת מעין יהלום, פינות חדות על הצירים. מפגש היהלום עם אליפסת MSE מקבל מינימום, לרוב, על הצירים.



כאשר הפינה של היהלום פוגעת באלפיסת השגיאה, נקודות אחרות על היהלום, כאלה בעלות אותו עונש, מתרחקות ממינימום השגיאה עוד יותר.

לעומת זאת, ברידג', שאין פינות על מעגל האילוץ, נקודות יתקרבו עוד יותר למרכז האליפסה (ולכן לא תבחר פינה).

הגרדיאנט של רידג' גדול יותר משל לאסו, ולכן כשהמשקולות נמוכות, לאסו מקטינה אותם חזק יותר מרידג'.

מקובל להניח שבהרבה מיקרים רידג' חוזה קצת יותר טוב מלאסו במיוחד כאשר יש הרבה מאפיינים רועשים ממקורות שאינם קורלטיביים. רידג' ייעזר בכולם ויתגבר על רעשים בנתונים טוב יותר.לאסו טוב יותר מאספקט הפרשנות (interpretability).
וכאשר רוצים שהמודל הסופי יהיה עם כמות מוגבלת של מאפיינים (כאלטרנטיבה ל feature selection היקר חישובית). תלוי בבעיה ובדאטה, לפעמים לאסו יעבוד טוב יותר, במיוחד כאשר יש מעט מאפיינים חשובים (ושאר המשתנים לא משפיעים הרבה על החיזוי אך מאידך מכניסים רעש) .

ניתן להחליט על סוג הנורמה בעזרת CV.

מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

= Elastic net regularization
שילוב בין שתי הנורמות. במשקולות הקטנים לאסו משפיע ואילו במשקולות הגבוהים, הרידג' (אבל הלאסו ממתן קצת). הנוסחה $\frac{lasso+ridge}{2}$.

אנסמבלים:
אנסמבלים הם צירופים של הרבה מודלים שונים, כאשר מתוכם יבחר המודל הטוב ביותר. זו גישה נוספת למניעת התאמת יתר.

כאשר כמות המידע מוגבלת, והמודל גמיש מידי, עלולים לקבל low bias, high variance (גמישות יתר).
אם יש ברשותנו מספר מודלים כאלו שהם לא קורלטיביים אחד עם השני:
- ✓ ברגרסיה ממצעים את התחזיות של כל המודלים
- ✓ בקלסיפיקציה אפשר למצע את ההסתברויות או אם מעונינים בהחלטה משתמשים בהצבעת הרוב

התאוריה מאחורי שימוש באנסמלים היא ע"י מיצוע של מודלים עם Bias נמוך ושונות גבוהה, מעלימים את השונות שנובעת מכך שהמודלים מותאמים יתר לנתונים עליהם התאמנו.

שיטות ליצירת אנסמבלים:
1. שיטת Bagging = יצירת אנסמבלים על ידי דגימות שונות מתוך סט האימונים. כלומר, מאמנים מודלים שונים על תת קבוצות שונות של הנתונים, ואת הפלט ממצעים או החלטת הרוב. את המודלים השונים ניתן ליצור במספר דרכים:
   a. שיטת boot strap = שיטת דגימה בה מייצרים הרבה קבוצות אימון שונות ע"י דגימה אקראית עם חזרות מתוך קבוצת אימון יחידה. לכל קבוצה מותאמת קבוצת ווליידציה הכוללת דוגמאות שלא נלקחו לדגימה.
   b. שיטת k fold = מיצרים k קבוצות אימון (בדומה ל k-fold cross validation) ע"י שמוציאים בכל פעם 1/k מהנתונים כווליידציה, ומשאירים בקבוצת האימון את היתר k-1/K. קבוצות הווליידציה הן זרות זו לזו. ניתן כמובן "לנצל" k-fold cross validation (שמטרתו לשערך שגיאה ולכוונן היפר-פארמטרים) על מנת לבנות bagging Ensemble בעזרת K המודלים שנוצרו.
2. שיטת boosting = אנסמבלים של מודלים מאומנים על קבוצות אימון שונות כך שכל מודל מתמקד בדוגמאות שהקודמים לו שגו בהן. כלומר, משתמשים בסדרה של מודלים "חלשים" (עם שגיאת ביאס גבוהה) וכל מודל מתרכז בלמידה של המקרים שהמודלים האחרים לא הצליחו לסווג נכון:
   a. משתמשים באותם נתונים, אך מגדילים את משקל הדוגמה (תדירות הופעתה) במידה והמודלים הקודמים לא סיווגו אותה נכון.
   b. מקבלים סידרת מומחים (שכל אחד "חלש" מידי – ולא יכול לעשות Overfitting) שנותנים תחזיות שונות למיקרים קשים.
   c. את המודלים השונים שמתקבלים ניתן לקבל ע"י למשל הצבעת הרוב או מיצוע (גיאומטרי) של הסתברויות– (אפשר לשקלל מודל על פי חשיבות השגיאות שמטופלות על ידו) או על פי מידת הצלחתו על ווליידציה
   d. חיסרון: רגיש ל outliers ועלול לגרום ל high-variance אם המומחים אינם כל כך חלשים.

חשיבות הנתונים:
ישנם ניסויים המראים כי אלגוריתמים שונים נותנים תוצאות דומות, וכולם משתפרים כתוצאה מעוד נתונים, אבל זה נכון רק תחת ההנחות הבאות:
- ✓ המאפיינים בקלט אכן מספיקים עבור הבעיה (כוללים מספיק מידע) (בדיקת שפיות: האם מומחה אנושי יוכל להסתדר עם מאפיינים אלו?)
- ✓ המודל עשיר בפארמטרים (low-bias) למשל רשת נוירונים עם הרבה נוירונים נסתרים
- ✓ כאשר משתמשים במעט דוגמאות אימון: שגיאת האימון קטנה

בקיצור: אם האבחנה היא High-bias, יש להשתמש באלגוריתם יותר "גמיש" (למשל: בעל יותר פארמטרים) או להוסיף מאפיינים , אם האבחנה היא high variance, תוספת דוגמאות אימון תעזור, רגולריזציה, Feature Selection.....

מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

אלגוריתמים שונים:

אלגוריתם K – nearest neighbors:

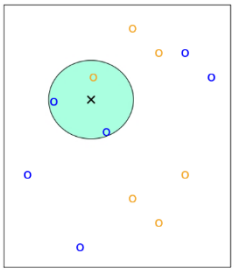
אלגוריתם ה-(KNN) k-nearest neighbors הוא אלגוריתם למידה מונחת פשוט וקל ליישום, שניתן להשתמש בו כדי לפתור בעיות קלסיפיקציה וגם בעיות רגרסיה. אלגוריתם זה הוא ותיק ופשוט שידוע כבר עשרות שנים הוא לא לינארי והוזנח מחוסר בסיס תיאורתי על אף שמוצלח בהרבה מיקרים פרקטיים.

עבור קלסיפיקציה:

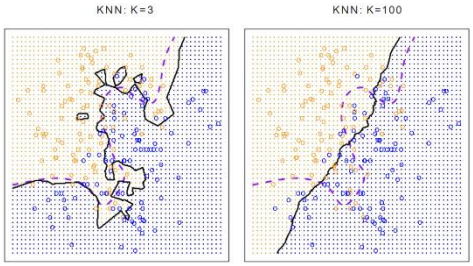
בהינתן אוסף דוגמאות (אימון) D, ודוגמת מבחן חדשה x , נשערך את ההסתברות האפוסטריורית (=המותנית) של כל הקטגוריות $p(y=j|x,D)$ ונבחר בקטגוריה שההסתברות שלה היא המקסימלית.

השיערוך באלגוריתם KNN הוא על פי הצבעת השכנים:

3-Nearest Neighbors (KNN)



- ✓ נמצא את הקבוצה N_x המכילה את K הדוגמאות מתוך D ה"קרובות" ביותר לדוגמא x
- ✓ ההסתברות לקטגוריה j מחושבת על פי מספר הדוגמאות מקבוצה N_x שסווגו לקטגוריה j באופן הבא: $p(y=j|x,D)=\frac{1}{k}\sum_{i\in N_x} I(y_i=j)$
- ✓ התחזית עבור הדוגמה x היא לכן על פי "הצבעת" הרוב מתוך קבוצת N_x השכנים באופן הבא: $I(y_i=j)=\begin{cases}1 \; if \; y_i=j \\ 0 \; else\end{cases}$



K=3 is too flexible, k=100 is almost linear
K=N is the null hypothesis: y= est_prior(+)

גבול ההחלטה (המפריד בין הקטגוריות השונות) אינו חייב להיות לינארי.

הגמישות יורדת ככל שמשתתפים יותר שכנים בהחלטה.

עבור רגרסיה:

בהינתן אוסף דוגמאות (אימון) D, ודוגמת מבחן חדשה x , עבור על k הדוגמאות הכי קרובות והחזר את ממוצע הערכים על פני k הדוגמאות השכנות.

ההחלטה מי קרוב:

עלינו להחליט על סמך איזו שיטה נחליט מי השכנים הקרובים לדוגמא.

1. כאשר המאפיינים נומרים בלבד:
   - ✓ מרחק אוקלידי, $L=2$, $d(q,p)=\sqrt{\sum_{i=1}^{n}(q_i-p_i)^2}$
   - ✓ מרחק מנהטן, $L=1$, $d(q,p)=|q|+|p|$
   - ✓ מרחק נורמת האינסוף $L=\infty$, $d(q,p)=\max(|q_i-p_i|)$
2. מרחק קוסינוס: מרחק זה מודד קרבה לפי דמיון, אורנטציה ולא לפי גודל.

$$cosineSim(A,B)=\frac{A\cdot B}{|A||B|}=\frac{\sum_{i=1}^{n}A_iB_i}{\sqrt{\sum_{i=1}^{n}A_i^2}\sqrt{\sum_{i=1}^{n}B_i^2}}$$

כזכור, הקוסינוס של הזווית אפס הוא 1, והוא פחות מאחד עבור כל זווית באינטרוול $[0,\pi]$.
לכן, שני וקטורים מאונכים אחד לשני הדמיון שלהם יהיה אפס, ואילו שני וקטורים עם אותה אורינטציה (זווית אפס), הדמיון שלהם יהיה אחד.

מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

3. מרחק קורלצית פירסון: מרחק זה הוא בעצם מרחק קוסינוס -1 כאשר הוקטורים מנורמלים מבסיס לממוצע שלהם. $d(A,B)=\frac{\sum_{i=1}^{n}(A_i-\overline{A})(B_i-\overline{B})}{\sqrt{\sum_{i=1}^{n}(A-\overline{A})_i^2}\sqrt{\sum_{i=1}^{n}(B-\overline{B})_i^2}}$. במקרה זה, המרחק נקבע על ידי הדמיון בצורה של פרופילים (למשל פרופיל לקוח).

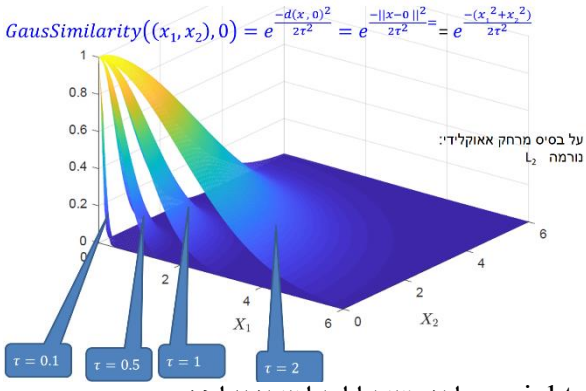
4. עבור וקטור בעל מאפיינים קטגורים: נעזר במרחק hamming = ספירת מספר הביטים השונים של 2 הוקטורים.

כאמור, דמיון יכול להיות קשור למרחק. ניתן לבצע גם את התהליך ההפוך- הפיכת מדידת מרחק לדמיון בין שתי נקודות:

1. ההופכי למרחק בריבוע: $GravityLaw(x_i,x)=\frac{1}{d(x_i,\ x)^2}$.

2. דמיון גאוסיאני: $GausSimilarity(x_i,x)=e^{\frac{-d(x_i,\ x)^2}{2\tau^2}}$.

טאו שולט על רוחב הגאוסיאן (בדומה לסטית התקן בהתפלגות נורמלית), טאו קטן: רק נקודות מאוד קרובות יקבלו ערך דימיון גדול, נקודות שאינן קרובות יונחתו לאפס.



שכלול האלגוריתם:
לפעמים נרצה להביא משקל רב יותר לשכנים הקרובים ביותר, weighted k-NN. לכן, נשכלל כל שכן על פי מידת הדמיון לשאילתה.

$$\hat{f}(x_q)\leftarrow\frac{\sum_{i=1}^{k}w_i f(x_i)}{\sum_{i=1}^{k}w_i}$$

where

$$w_i\equiv\frac{1}{d(x_q,x_i)^2}$$

כדי לטפל במקרה שבו המרחק הוא אפס, לוקחים רק את הנקודות שמרחקן 0 כי משקלן אינסופי.
לחילופין, פונקצית משקל אלטרנטיבית: ללא חלוקה באפס. בעזרת דימיון גאוסי, מאפשרת שליטה בגמישות באמצעות tau. טאו משפיע כמו סטית התקן על רוחב צורת הגאוסיאן: כשטאו קטן, נקודות רחוקות מהתוחלת מקבלות משקל קטן.
יש הגיון להשתמש בכל הדוגמאות ולשקללם לפי המרחק (מזכיר מאוד Locally Weighted Linear Regression).

$$w_i=e^{\frac{-||x_i-x||^2}{2\tau^2}}$$

בעיה באלגוריתם KNN:
המרחק מחושב על סמך כל המימדים (בניגוד למשל לעצי החלטה). למשל: אם יש 20 מימדים ורק 2 רלוונטים: דוגמאות שזהות ב 2 המימדים הרלונטים אבל רחוקים ב 18, יהיו רחוקות זו מזו.
אי-הדימיון מטעה! KNN רגיש מאוד ל"קללת המימדים". ניתן להראות תיאורתית שאם ערכי המאפיינים מתפלגים באופן אחיד או נורמלי, מימד גבוה מפריע להערכת המרחקים ופוגע לכן בחיזוי.

פתרונות אפשריים לבעיה זו:
- ✓ Feature selection, דחיסת מימדים,
- ✓ בחישוב המרחק: משקל שונה לכל מאפיין: כל מימד נמתח או מתכווץ עי מכפלה בקבוע כיווץ (נחליט על הקבועים בעזרת CV ו/או מהיכרות עם הנתונים)

למרות קללת המימדים, למזלנו בסוגי נתנוים מסוימים קימת ברכת חוסר האחידות: בהתפלגויות אמיתיות שאינם אחידות או נורמליות. למשל, על תמונות וקול, KNN עובד לפעמים לא רע, למרות המימד הגבוה.

יתרונות האלגוריתם:

- ✓ אימון מהיר
- ✓ המודל לומד את הסיבוכיות של המטרה בקלות
- ✓ אין איבוד מידע

חסרונות האלגוריתם:

- ✓ זמן שאילתה איטי
- ✓ איחסון מידע רב
- ✓ מושפע רבות ממאפיינים לא חשובים (קללת המימדים)

מרצה: ד״ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

למידה בייסיאנית Bayesian learning:
הרעיון של למידה בייסיאנית הוא לחשב את התפלגות ההסתברות הקודמת של מאפייני המטרה של דוגמה חדשה המותנה בתכונות הקלט שלה ובכל דוגמאות האימון.הלמידה הבייסיאנית מבוססת באופן מוקפד על תורת ההסתברות ובפרט על משפט Bayes (הסתברות מותנית).

התחום יוצא מנקודת ראות תיאורטית, אך הפיק אלגוריתמי למידה פרקטיים:

- ✓ LDA ,Naïve Bayes Classifier
- ✓ למידה ברשתות Bayes: שילוב ידע מוקדם בצורה גרפית ; שילוב ידע מוקדם עם נתונים ניצפים

למידה בייסיאנית מהווה מסגרת רעיונית שימושית. היא התפתחה להיות סטנדרט "הזהב" בלמידת מכונה.

חוק בייס לאירועים בדידים בלתי תלויים:
חוק בייס הוא תוצאה בתורת ההסתברות המאפשרת לחשב הסתברות מותנית של מאורע כאשר יודעים את ההסתברויות המותנות ההפוכות.

באופן פורמלי, ההסתברות המותנית של מאורע A בהינתן מאורע B היא הסיכוי להתרחשותו של A בהנחה ש- B אכן התרחש. קיום ההנחה מצמצם את מרחב המדגם. מרעיון זה נובעת הנוסחה להסתברות המותנית הבאה: $P(B|A) = \frac{P(A\cap B)}{P(A)}$. הסתברות זו נקראת "אחורית" (posterior). לעומת זאת, ההסתברות למאורע כללי $P(B)$ נקראת "קודמת" (prior).

ואילו, חוק בייס מאפשר לחשב את ההסתברות המותנית ההפוכה- ההסתברות המותנית של B בהינתן A:
$P(A|B) = \frac{P(B|A)\cdot P(A)}{P(B)}$.

מתוך נוסחת ההסתברות השלמה, במידה והמאורעות A_i זרים אז $P(B) = \sum_i^k P(B|A_i)P(A_i)$ ואז:
$P(A_i|B) = \frac{P(B|A_i)\cdot P(A_i)}{\sum_i^k P(B|A_i)P(A_i)}$.
כלומר, ניתן לחשב הסתברות posterior של A_i בהינתן B בעזרת priors וכן ההסתברות המותנית ההפוכה לה.

חוק בייס עבור קלסיפיקציה:
נגדיר מאורעות זרים שהם הקטגוריות השונות בקלסיפיקציה $1, \dots, k$. ואז, למשל, בהינתן דוגמא x, ההסתברות שלה להיות מסווגת בקטגוריה k היא

$$P(k|x) = \frac{P(x|k)\cdot P(k)}{P(x)} = \frac{P(x|k)\cdot P(k)}{\sum_i^k P(x|i)P(i)} = \frac{P(x\cap k)}{P(k)}\cdot\frac{P(k)}{\sum_i^k P(x|i)P(i)} = \frac{P(x\cap k)}{\sum_i^k P(x|i)P(i)}$$

את ההתסברות שהיא priors של הקטגוריות בד"כ קל לשערך. אם היינו יודעים לשערך גם הסתברויות הקלט בהינתן קטגוריה $p(x|k)$, או לחילופין את $p(x\cap k)$ אז היינו יכולים לחשב את ההסתברות הפוסטריורית $p(k|x)$ לכל לקטגוריה k.

לדוגמא $x = fever$ הינו סימפטום (מאפיין), ורוצים לחשב את ההסתברות של להיות חולה בקורונה או בריא כתלות בסימפטום שהוא חום $p(k = corona|fever);\ p(k = healthy|fever)$.

מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

הערכה מקסימלית להסתברות אחורית MAP = Maximal A-Posteriori Estimation:
נרצה לחזות את הסיווג על פי הקטגוריה בעלת ההסתברות המקסימלית מבין כל הקטגוריות האפשריות.

$$y_{MAP} = argmax_k \{p(k\,|x)\}$$

כאשר:

- ✓ k היא קטגורית סיווג (מתוך קבוצה K של קטגוריות)
- ✓ x הוא וקטור של ערכי קלט (מאפיינים) אותו רוצים לסווג

כלומר, אם נשתמש במדד map נצטרך לחשב:

$$argmax_k \{p(k\,|x)\} = argmax_k \left\{\frac{P(x|k)\cdot P(k)}{\sum_i^k P(x|i)P(i)}\right\} = argmax_k\{P(x|k)\cdot P(k)\}$$

לשמחתנו כדי לסווג מהי הקטגוריה בעלת ההסתברות המקסימלית לא צריך לחשב את המכנה. בד"כ המכנה קשה לשיערוך היות וצריך את ההסתברות של כל צירוף מאפיינים. אבל אם נרצה לחשב את ההסתברות הפוסטריורית ממש, נצטרך לחשב את המכנה.

הערכה מקסימלית להסתברות סבירות ML= Maximal Likelihood Estimation :
נרצה לחזות את הסיווג על פי הקטגוריה בעלת ההסתברות המקסימלית מבין כל הקטגוריות האפשריות. אבל, כאשר לא יודעים לשערך את ה Priors של הקטגוריות, בהרבה מיקרים מניחים שהסתברות prior של הקטגוריות שווה זו לזו ולכן ניתן למקסם רק את $P(x|k)$, כלומר, בשונה מהערכת map :

$$y_{ML} = argmax_k \{P(x|k)\}$$

ניתן לחזות את הקטגוריה בקלסיפיקציה ביזיאנית בעזרת ML אך אם הנחת השיוויון ב Priors שגויה, נקבל הבדלים מול חיזוי על פי MAP.

נשים לב כי נתוני recall, precision מספקים מידע הסתברותי חשוב עבור החישובים שמעלה.
דוגמא:
מטופל נבדק במעבדה והבדיקה חוזרת חיובית למחלת הסרטן. האם תבחר לעשות לו ניתוח מורכב כאשר יודעים את הרגישות והספציפיות של הבדיקה?

- ✓ recall: ידוע כי בדיקת מעבדה חיובית תתקבל ב 98% מהמיקרים בהם ישנה המחלה
- ✓ true negative rate: תוצאה שלילית נכונה תתקבל ב 97% מהמיקרים בהם המחלה איננה
- ✓ ידע מוקדם: 0.8% מהאוכלוסיה חולים במחלה

מנתונים אלה אנו מבינים:

- ✓ $P(-|cancer) \approx 0.02, P(+|cancer) \approx 0.98$
- ✓ $P(-|\neg cancer) \approx 0.97\,, P(+|\neg cancer) \approx 0.03$
- ✓ $P(cancer) \approx 0.008, P(\neg cancer) \approx 0.992$

ברור כי לסיווג ישנן שתי קטגוריות $cancer$, $\neg cancer$ וכן ישנו מאפיין אחד בוליאני שהוא תוצאת הבדיקה $+$, $-$.

כעת, נוכל לחשב את הסיווג על פי הקטגוריה בעלת ההסתברות המקסימלית מבין כל הקטגוריות האפשריות, כאמור, בשני אופנים:

1. לפי MAP :
   I. $y_{cancer} = P(cancer|\,+) = P(+|cancer)\cdot P(cancer) = 0.98\cdot 0.008 = 0.00784$
   II. $y_{\neg cancer} = P(\neg cancer|\,+) = P(+|\neg cancer)\cdot P(\neg cancer) = 0.03\cdot 0.992 = 0.0298$
2. לפי ML :
   I. $y_{cancer} = P(cancer|\,+) = P(+|cancer) = 0.98$
   II. $y_{\neg cancer} = P(\neg cancer|\,+) = P(+|\neg cancer) = 0.03$

ואז, נוכל לבחור היפותזה בהתאם לכל אחת מהגישות:

1. לפי MAP : $0.0298 > 0.00784$ כלומר, הסיווג שיבחר הוא $y_{\neg cancer}$, לא חולה
2. לפי ML : $0.98 > 0.03$ כלומר, הסיווג שיבחר הוא y_{cancer} , חולה

נשים לב כי דוגמה זו לא שיערה את ההסתברות עצמה, אלא רק נתנה סיווג שיבחר.

מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

במידה והיינו רוצים לחשב את השערוך להסתברויות, היינו מחשבים:

1. עלינו לחשב את ההסתברות האחורית של לקבל בדיקה חיובית, ולכן:

I. $P(+\cap cancer) = p(+|cancer)p(cancer) \approx 0.98 \cdot 0.008 = 0.00784$

II. $P(+\cap \neg cancer) = p(+|\neg cancer)p(\neg cancer) \approx 0.03 \cdot 0.992 = 0.0298$

III. $P(+) = P(cancer \cap +) + P(\neg cancer \cap +) \approx 0.00784 + 0.0298 = 0.03764$

2. כעת, על פי הנוסחה להתברות מותנית ניתן לחשב את השערוך:

I. $P(cancer|+) \approx \frac{P(+\cap cancer)}{P(+)} = \frac{0.00784}{0.03764} \approx 0.21$

II. $P(\neg cancer|+) \approx \frac{P(+\cap \neg cancer)}{P(+)} = \frac{0.0298}{0.03764} \approx 0.79$

נשים לב כי החישוב הנ"ל מאמת את גישת MAP.

וחזרה לשאלה בתחילת הדוגמה- האם תבחרו לעשות ניתוח להוצאת הגידול? התשובה היא שזו החלטה סובייקטיבית, כלכלית, כמה אתם מעריכים את חייכם? וכמה את הסבל הסיכון והעלות הכרוכים בניתוח?

שערוך הסתברויות מתוך מדגם הנתונים:

אם יש מדגם גדול, נוכל לספור ולשערך הסתברות.

למשל, בהינתן מדגם גדול המשקף את האוכלוסיה של חולים ובריאים עם תוצאות הבדיקה, נוכל לספור:

סהכ נבדקים n	כמה חולים n_c	כמה בריאים n_h	כמה חיוביים n_+	כמה שליליים $n - n_+$	לכמה חולים יש בדיקה חיובית $n_{c\cap+}$	לכמה בריאים יש בדיקה חיובית $n_{h\cap+}$	

ואז, נוכל לשערך הסתברויות: $P(+|c) = \frac{n_{c\cap+}}{n_c}$, $P(+) = \frac{n_+}{n}$.

הנחות בייס:

שיטות אלו מצויינות כאשר וקטור המאפיינים הוא חד מימדי כפי שראינו בדוגמה, אבל איך נשערך את ההסתברות לקטגוריה בעבור וקטור רב מימדים?

אם x הוא ממימד קטן והמאפינים הם בעלי מספר קטן של ערכים אפשריים, לעיתים יהיו מספיק נתונים בסט הנתונים כדי לשערך ישירות ע"י ספירה $P(x\ |k) = \frac{n_{x\cap k}}{n_k}$. אך כאשר למאפינים יש הרבה ערכים, או כשהמימד גבוה, לא יהיו לנו הרבה דוגמאות בסט הנתונים שזהות לוקטור הקלט x (אם בכלל), השערוך הישיר להסתברות לא יהיה מדויק.

נעזר בהנחות שונות.

הנחת בייס נאיבית:

1. הנחה נאיבית חזקה (אגרסיבית): המאפיינים אינם תלויים זה בזה בהינתן קטגוריה k, כלומר ההסתברות לכל וקטור הקלט הוא פשוט כפל ההסתברויות $p(v_1, v_2, \dots v_n\ |k) \approx \prod_{j=1}^{n} p(x_j = v_j|k)$. בנוסף, ידוע כי $p(k) \approx \frac{n_k}{n}, p(x_j = v_j|k) \approx \frac{n_{v_j,k}}{n_k}$.
ואז, הסיווג הינו

$$y_{MAP} \approx argmax_k \{p(k) \cdot \prod_{j=1}^{n} p(x_j = v_j|k)\}.$$

2. סדר המילים אינו חשוב, ההסתברות של מילה מסוימת להופיע בכל פוזיציה היא שווה (איננה תלויה בפוזיציה במסמך) $p(a_i = w_j|k) = p(a_m = w_j|k)$.

מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

תחת הנחה זו, נלמד את אלגוריתם הקלספיקציה הנאיבי של בייס:

1. הערכת כל ההסתברות הנחוצות – התסברויות אחוריות, הסתברויות של מאפיינים בהתאם לקטגוריה
2. קבלת קלט וחישוב הסיווג שמוחזר.

באופן פורמלי, האלגוריתם נראה כך:

Naïve Bayes Classifier (D, K, x):
For each category k=1…K:
- $P(k) \approx \frac{n_k}{n}$
- For each value v_j of each attribute x_j : $P(v_j|k) = p(x_j = v_j|k) \approx \frac{n_{v_j \cap k}}{n_k}$

Return $predict = argmax_k \{P(k) \cdot \prod_{j=1}^{n} P(v_j|k)\}$

אלגוריתם זה פשוט מאוד, קל לחישוב והבנה, בעל הנחות מפשטות שגויות (בד"כ), ומצד שני, באופן מפתיע, נותן לפעמים תוצאות טובות למדי, גם כאשר הנחות האי-תלות לא מתקיימות.

אלגוריתם זה שימושי בקלסיפיקציה רפואית ועיבוד שפה טבעית. למשל, גוגל השתמשה בהנחות ביזיאניות נאיביות בתחילת דרכה (ואולי עדיין) עבור קלסיפיקציה של מסמכים.

ישנה בעיתיות בהנחת הנאיביות. במקרים רבים מקבלים הערכות להסתברויות אפוסטריוריות קרובות לאחד או לאפס באופן לא ריאלי (בגלל תלות סטטיסטית בין המשתנים או בגלל מידגם קטן מה שמוביל לשיערוך לא מדויק). מכיוון שהסתברות לא יכולה להיות אפס, נהוג לתקן שיערוכים של ערכי הקטגוריות הנדירים.

תיקון לשיערוכי הסתברות עבור ערכים נדירים $m - estimate$:

נהוג לתקן שיערוכים של ערכי הקטגוריות הנדירים בהסתמך על הסתברות הקודמת של ערכים אלו- "כאילו" שקימות עוד מספר קטן m של דוגמאות בקטגוריה שבהן מופיע הערך v בהסתברות של $prior(v)$.

כאשר אין מספיק דוגמאות מקטגוריה $y = k$ ושבהן גם $x_i = v_i$, מוסיפים m דוגמאות וירטואליות מקטגורה k שמתוכם $P(x_i = v_i)$ הם בעלי ערך $x_i = v_i$. $P(x_i = v_i)$ הינו השערוך הקודם של v_i. m הינו היפר פאראמטר שמצפה על גודל המדגם או על התלויות.

$$\hat{P}(x_i = v_i|k) \approx \frac{n_{i \cap k} + m \cdot P(x_i = v_i)}{n_k + m}$$

אם לא ניתן לשערך את ההסתברות הקודמת $P(x_i = v_i)$ כי למשל סט הנתונים אינו מייצג אוכלוסיה כללית או שהערכים מאוד נדירים, נבחר שערוך אחר.

תיקון לשיערוכי הסתברות עבור ערכים נדירים $Laplace\ Smoothening$:

אם לא ניתן לשערך את ההסתברות הקודמת $P(x_i = v_i)$, אפשר להניח התפלגות אחידה בין הערכים $v_1 \dots v_n$ ולשערך $P(x_i = v_i) \approx \frac{1}{n}$. ואז ההסתברות תהיה

$$\hat{P}(x_i = v_i|k) \approx \frac{n_{i \cap k} + m \cdot \frac{1}{|v|}}{n_k + m}$$

בפרט, אם נוסיף $m = |v|$ דוגמאות פיקטיביות נקבל: $\hat{P}(x_i = v_i|k) \approx \frac{n_{i \cap k} + |v| \cdot \frac{1}{|v|}}{n_k + |v|} = \frac{n_{i \cap k} + 1}{n_k + |v|}$

למשל, מילים נדירות שקיימות במילון אך לא נמצאות כלל בקטגוריה k בסט הנתונים ישוערכו $\frac{1}{n_k + |v|}$.

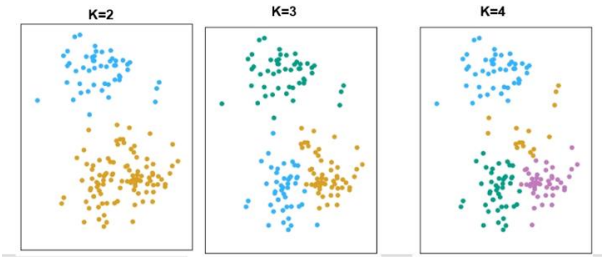
למידה בלתי מונחת:
נלמד שתי שיטות של למידה שאינה מונחת עבור למידת מכונה.

שיטת אשכולות Clustering:
זוהי שיטה בלתי פרמטרית מיועדת עבור קלסיפיקציה. בהינתן קבוצה D של דוגמאות אימון (ללא תיוגים) רוצים לחלק לקבוצות זרות (=אשכולות, קלסטרים) שמכסות את D ומכילות דוגמאות "דומות" כך שהנקודות שבתוך קלסטר תהינה דומות זו לזו ואילו נקודות בקלסטרים שונים תהינה שונות זו מזו.

משתמשים בשיטה זו עבור:
1. גילוי ידע= הבנת קשרים ודימיון בין נקודות הנתונים
   - מציאת דימיון בין נקודות – למשל זיהוי דוברים לא ידועים, ויזואליזציה, סגמנטציה
   - מדידת מרחק בין קבוצות של נקודות: למשל כמה רחוקה קבוצת הפרימטים מהלטאות
2. איתור חריגות מהתנהגות רגילה (למשל בתחנת כוח, תעבורת רשת), איתור שגיאות

שיטת האשכולות לא מוגדרת במדויק כמו Supervised learning. לא תמיד ברור מה רוצים להשיג:
- ✓ כיצד מודדים "דימיון" בין קבוצות (לא ברור, תלוי אפליקציה). שיטות תלויות אפליקציה. דימיון אינו תמיד יחס שקילות.
- ✓ כיצד מערכים את התוצאות. ישנן המוני שיטות לקלסטרינג, אין שיטה אחת שמתאימה לכל.

K-Means Clustering:
נתונה קבוצת דוגמאות D ממימד n. נתון מספר ה clusters (אשכולות) K.



- ✓ נחלק את סט הנתונים ל K קבוצות זרות המכסות את כל הסט
- ✓ נרצה כי הדימיון בין הנקודות בתוך כל cluster יהיה הגדול ביותר, או במילים אחרות, השוני (אי-הדימיון) בין הנקודות שבכל אשכול יהיה מינימלי

לשם כך, נזדקק:
- ✓ לפונקציה "אי דמיון" שמהווה מרחק בין וקטורים, למשל מרחק אוקלידי $d(x_{i,}, x_{i'})^2 = ||x_i - x_{i'}||^2 = \sum_{j=1}^{m}(x_{i,j} - x_{i',j})^2$
- ✓ למדד לשוני בין נקודות בתוך האשכול, למשל Within-Cluster-Variation, סכום הממוצעים של מרחקי נקודה משאר הנקודות בקלסטר $WCV(C_k) = \frac{1}{|C_k|}\sum_{i,i' \in C_k} d(x_{i,}, x_{i'})^2$
- ✓ לפונקציה שמוצאת אשכולות שהסכום הכולל של השוני WCV של כל אחד מהם הוא מינימלי, למשל $TWCV = \sum_{k=1}^{K} WCV(C_k)$

כמובן שנרצה להעזר באלגוריתם הקליסטור מבוסס על אופטימיזציה $minimize_{c_1 \dots c_k,}\{TWCV\}$.

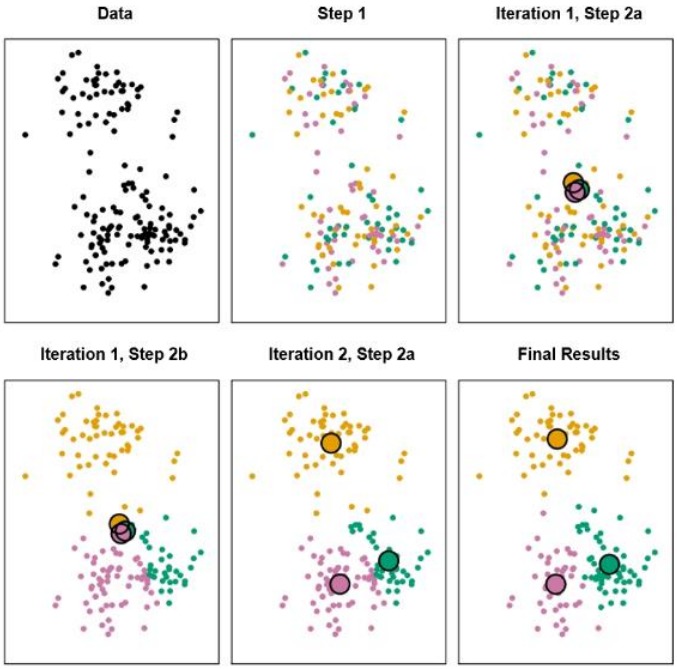
האלגוריתם הינו:
1. מצרפים רנדומית כל דוגמא לאחד מתוך K הקלסטרים
2. חוזרים שוב ושוב עד שאין שינוי בקלסטרים:
   א. מחשבים את המרכז לכל קלסטר
   ב. מצרפים מחדש את הנקודות על פי קירבתם למרכזי הקלסטרים.

מרכז הקלסטר הוא הווקטור של ממוצעי המאפיינים של הדוגמאות שבקלסטר.

אם נשתמש בנורמה d לחישוב המרחק בין 2 וקטורים, ניתן להוכיח כי בכל איטרציה, סכום המרחקים $TWCV$ אינו גדל ולכן בכל צעד שבו מעבירים דוגמא מקלאסטר לקלאסטר (מבצעים שינוי), משפרים.

מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

כך, למשל, ניתן לראות הרצה של אלגוריתם זה על מידע כללי כלשהו :



ניתן לראות כי בסוף האיטרציות, ישנם אשכולות ברורים ומופרדים ולכל אחד מהם יש את המרכז שלו.

אלגוריתם K-means מבצע ירידה בפונקצית המטרה WCV עד לקבלת מינימום מקומי. ניתן לראות שקיים קשר בין ה WCV לבין סכום המרחקים ממרכז הקלסטר :

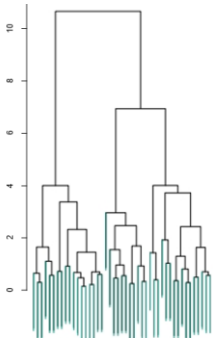
המרחק של נקודה מהסנטרואיד

$$\frac{1}{|C_k|}\sum_{i,i\prime\in Ck}\sum_{j=1}^{m}(x_{ij}-x_{i'j})^2=2\sum_{i\in C_k}\sum_{j=1}^{m}(\mathrm{x_{ij}}-\bar{x}_{kj})^2$$

בכל איטרציה המרחק בין הנקודות יורד מונוטונית. מחשבים סנטרואידים חדשים ואז מכניסים לקלסטר את הנקודות הקרובות לכל כל סנטרואיד. המרחק של הנקודות מהסנטרואיד יכול רק להשתפר- אם התווספה נקודה לקלסטר חדש הרי שהנקודה יצאה מקלסטר ישן ואז מרחקה ממרכז הקלסטר החדש קטן ממרחקה ממרכז הקלסטר הישן, ולכן ה WCV של הקלסטר החדש גדל פחות מההקטנה ב WCV שקרתה בקלסטר הישן. מתבצעת לכן ירידה בפונקצית המטרה TWCV עד למציאת מינימום מקומי.

פונקצית המטרה אינה קמורה בהכרח ולכן נקודת המינימום שנמצא אינה בהכרח מינימום גלובאלי, לכן רצוי לנסות את האלגוריתם כמה פעמים (התחלה רנדומית שונה) ולבחור WCV מינימלי.

חיסרון גדול של K-means : יש לקבוע מראש את מספר הקלסטרים. קשה מאוד לדעת כמה קלסטרים מראש נצטרך. הפתרון לבעיה זו הוא שימוש בהיררכיה של קלסטרים.



קלאסטרים היררכים :
השיטה המקובלת היא היררכיהמלמטה למעלה : מתחילים כשכל דוגמא בסט הנתונים היא קלסטר (עלה בעץ)
בכל איטרציה מאחדים 2 קלסטרים שהם הכי קרובים עד שמקבלים בקלסטר אחד את כל הסט שוב.

ברגע שינו גרף Den do Grams נוכל ע״י חיתוך מלמעלה בגובה המתאים, לקבל כל מספר של קלסטרים שנרצה.

אלגוריתם Bottom-up לקליסטור היררכי:

נתונה פונקצית מרחק $d(x,y)$ בין 2 ווקטורים ובעזרתה נגדיר "אי דימיון" בין קלסטרים וכן נתונה $Inter-Cluster-DisSimilarity\ (C1,C2)$ המודד כמה 2 קלסטרים שונים זה מזה.

1. הכנס כל דוגמא $x \in D$ לקלסטר נפרד. חשב את מדד אי-הדימיון לכל $\binom{n}{2}$ זוגות הדוגמאות.
2. עבור $i = n, n-1, \dots 2$: (עד שכל הדוגמאות מאוחדות בקבוצה אחת)
   א. בחר זוג קלסטרים בעלי אי-דימיון (ICD) מינימלי בין הזוג ומזג אותם לקלסטר אחד
   ב. חשב את אי הדימיון (ICD) של הקלסטר הממוזג מול כל אחד מהקלסטרים שנותרו
   ג. גובה הקלסטר הממוזג בדנדוגרם הוא ה WCV(C) בין הדוגמאות ששויכו לקלסטר

$$WCV(C_k) = \frac{1}{|C_k|} \sum_{i,i' \in C_k} ||x_{i,} - x_{i'}||^2$$

בכל איטרציה מספר הקלסטרים יורד באחד ואילו אי הדימיון בקלסטר האחרון שמוזג גדל.
ישנן כמה וריאציות כיצד נמדד אי דימיון בין קבוצות (כדי למזג):

1. מקסימום אי דימיון:
   חשב את כל אי הדמיון עבור כל זוג d(x,y) בין x∈C1 ל-y∈C2 ותן כפלט את השונות המקסימלית.
2. ממוצע:
   חשב את כל אי הדמיון עבור כל זוג d(x,y) בין x∈C1 ל-y∈C2 ותן כפלט את ממוצע השונות.
3. מינימום אי דמיון:
   חשב את כל אי הדמיון עבור כל זוג d(x,y) בין x∈C1 ל-y∈C2 ותן כפלט את השונות המינימלית.
   נשים לב כי שיטה זו בעלת נטייה ליצור עץ לא מאוזן.
4. מרכוז:
   תן כפלט את השונות d(center(c1),center(c2)) בין המרכזים של שני האשכולות. בדרך כלל מרכז אשכול הוא הנקודה הממוצעת. נשים לב כי שיטה זו בעלת נטייה לייצר בעיות בויזואליזציה.

בנוסף, ישנן כל מיני שיטות למדוד אי דמיון בתוך קבוצה:
1. מרחק אוקלידי
2. מרחק מבוסס קורלציה

כלומר, על מנת להשתמש בשיטת האשכולות, עלינו להחליט:
1. באיזה מרחק אי דמיון נשתמש
2. האם וכיצד ננרמל את המאפיינים
3. כמה אשכולות נרצה לאתר

חשוב לציין כי שיטת האשכולות לא תמיד עובדת, כאשר:
- ✓ כאשר אין הפרדה טובה בין תת הקבוצות
- ✓ כאשר נפח תת הקבוצות שונה משמעותית
- ✓ כאשר תת קבוצה אחת דלילה משמעותית עלולים לא למצוא אותה כלל ולעוות את הקלסטרים הפחות דלילים
- ✓ קימים מספר קטן (ולא ידוע) של תת קבוצות אמיתיות אך יש מספר קטן של outliers שלא שיכים ללשום תת-קבוצה אמיתית, הקליסטור עלול להקצות קלסטרים גם ל outliers ובכך לעוות את תת הקבוצות האמיתיות

מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

הפחתת מימדים : Principle Component Analysis

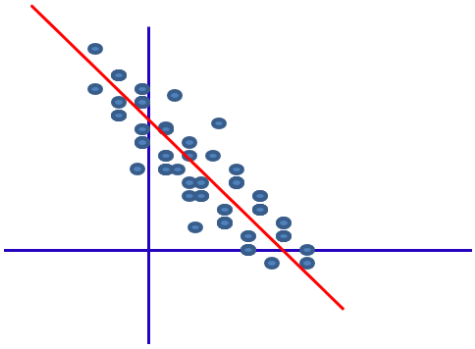
שיטה זו היא בעצם ביצוע טרנספורמציות לינאריות ממימד n למימד $p < n$ המנסות לשמר כמה שיותר אינפורמציה באמצעות קומבינציות לינאריות של המאפיינים המקוריים.

הפחתת ממדים נחשבת גם כטכניקת למידה unsupervised מכיוון שמשתמשת רק בנתונים עצמם (ולא בתוויות).

שימושים של שיטה זו :

- ✓ שימוש עבור ויזואליזציה. נוכל להציג את נקודות המדגם D כתמונה דו/תלת ממדית.
- ✓ שימוש עבור הפחתת ווריאנס : לאחר שהיתמרנו למרחב מממד קטן, נוכל להריץ כל אלגוריתם Supervised, בדומה ל Feature Selection.
- ✓ דחיסה לצורך ריחסון יעיל בזיכרון

ועוד.



כאמור, נשאף לאובן מידע מינימלי ולכן נבצע הטלות. למשל, עבור הגרף הבא, רוצים לצייר את כל הנקודות במימד אחד, ונשאלת השאלה היכן יעבור הציר האופטימלי. ההטלות על הציר האופטימלי הן הנקודות החדשות. המרחק הממוצע בין ההטלות ממוקסם (מיקסום השונות) ואילו המרחק האוקלידי הממוצע בין הנקודות לציר ממוזער (מיזעור ה MSE).

1st Principle Component :

מיקסום השונות בו זמנית עם מזעור המרחק, כלומר, המרחקים האוקלידים של הנקודות מהציר מתמזערים באותו רגע בו מתמקסמת השונות על הציר.

Principle Component Analysis באופן פורמלי :

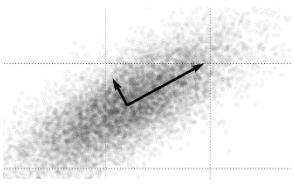
נתון מדגם D של נתוני קלט n מימדי : $x = (x_1, x_2, \dots, x_D)$. מחפשים טרנספורמציות לינאריות ששיעבירו את הקלט המקורי x ממימד n ל z בממימד m (קטן מ n) תוך שימור של המידע "החשוב".
הטרנספורמציות שמחפשים הם $\emptyset_1, \emptyset_2, \dots \emptyset_m$. וקטורים ממימד n ,
$z = (z_1, z_2, \dots, z_m)$ הם m קומבינציות לינאריות של n המאפיינים המקוריים של דוגמא x בעזרת הטרנספורמציות. z_i הוא ההטלה של דוגמא x על הציר ה- i .

The score of the ith PC

$$z_i = \sum_{j=1}^{n} \emptyset_{j,i} x_j$$

$z = (z_1, z_2, \dots, z_m)$ הוא וקטור ההטלות של x על מערכת צירים חדשה בה יש m צירים אורתוגונלים.

נרצה למצוא m<<n טרנספורמציות $\emptyset_i$ שתשמרנה אינפורמציה "חשובה". כל אחת מ m הטרנספורמציות נקראת Principle Component. $\emptyset_i$ הוא הקומפוננט (הציר) ה i. Principle component הראשון הוא הכי משמעותי, ה PC השני הוא שני בחשיבותו וכן הלאה.

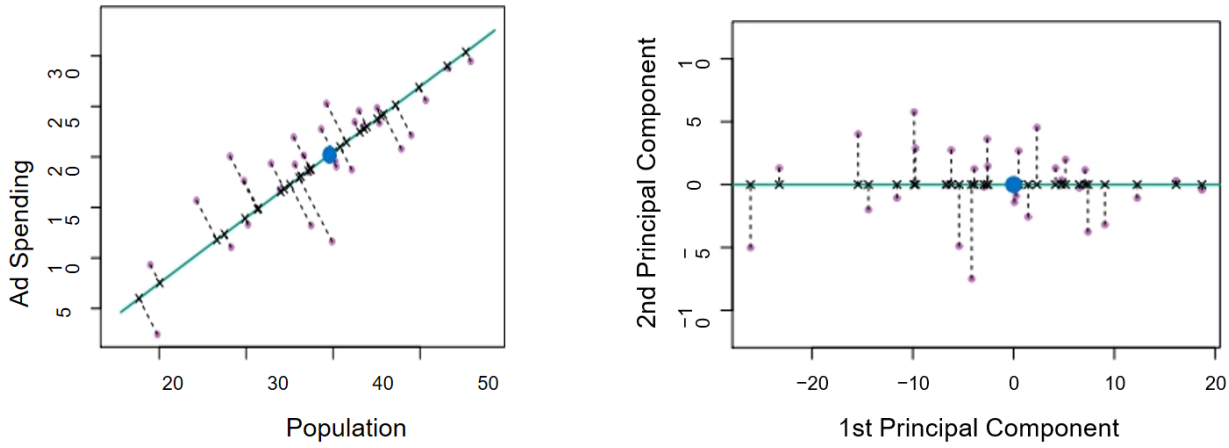


הצירים החדשים הם הוקטורים העצמיים של מטריצת ה Covar של D. הציר שנותן את השונות הכי גדולה בין ההטלות של הנקודות על הציר הוא ה Principle Component , הוקטור העצמי עם הערך העצמי הדול ביותר של מטריצת ה covar של הנקודות ב D. הווקטור עם הערך העצמי הבא הכי גדול, הוא הציר השני. מאונך לווקטור העצמי העיקרי.

מרצה : ד"ר גדי פנקס
נכתב על ידי : יעלי בר גימלשטיין

כלומר, מחפשים טרנספורמציה לינארית למערכת צירים שממקסמת שונות.הכיוון העיקרי שממקסם את האינפרמציה הטמונה בנתונים- ממקסם את השונות בין נקודות וכן ההיפר מישור הכי קרוב לנקודות, ההתאמה הלינארית הכי טובה לקשר בין המאפיינים.

הטרנספורמציה z_1 של 1st PC תחשב את ההטלה של כל נקודה על הציר החדש.



אם נרצה לדחוס את 2 הקלטים למספר אחד, אוסף ההטלות על הציר של הרכיב הראשון, יתן שונות מקסימלית וגם יבטא קורלציה בין 2 המאפינים אם קימת (בדוגמא יש קשר לינארי רועש בין 2 המאפינים).

שינוי מערכת צירים:

1. הווקטורים מנורמלים מסביב לממוצע 0 (ע״י הפחתת הממוצע)
2. הקומפוננט הראשי הראשון הוא טרנספורמציה לינארית המתקבלת ע״י אלגוריתם הממקסם את השונות של ההטלות
3. שונות ההטלות על כל ציר אחר תהיה קטנה יותר
4. ברגע שנוסיף את הציר ההשני (והאחרון), לא נאבד שום אינפורמציה

ייצוג הרכיבים:

פירוש נוסף לרכיב הראשון הוא שדוחסים 2 מספרים למספר אחד שמיצג הכי טוב את 2 המספרים המקוריים. למשל אם יש קשר לינארי בין 2 המאפינים, הרי המספר הדחוס לא יאבד שום אינפורמציה.

הרכיב השני z2 הוא קומבינציה לינארית של המאפינים המקוריים שאיננה קורלטיבית (אורתוגנלית) עם הרכיב הראשון ויש לה שונות מקסימלית (בכפוף לאילוץ האורתוגנליות).
ב 2 מימדים, הרכיב השני אינו תורם להפחתת המימד, אך במימדים גבוהים, יש לבחור ציר מאונך לציר הראשון כך שהשונות של ההטלות עליו ממוקסמת.

על פי הבניה, הרכיב הראשון מכיל את מקסימום האינפורמציה, הרכיב השני מכיל פחות אינפורמציה, הרכיב השלישי עוד פחות....וכך הלאה.

PCA כבעיית אופטימיזציה:

ממקסמים variance (שקול למיזעור מרחק L2). ללא הגבלת כלליות נניח שהתוחלת של כל מאפיין היא 0. נוכל להסב כל משתנה מקורי כך שיהיה מסביב ל 0. מכיוון שישנן אין סוף צירים שקולים (למשל מכפלה בקבוע) נבקש שלצירים תהיה נורמה 1:

$$\sum_{j=1}^{n} \emptyset_{j,p}{}^{2} = 1$$

זו גם בעיית מיקסום, מציאת ה PC הראשון.
ב D יש m דוגמאות שנורמלו מסביב ל 0. כיצד נמצא את ה- PC הראשון: $\emptyset_1 = (\emptyset_{1,1} \ \dots \emptyset_{n,1})$
שיבצע טרנספורמציה על ווקטורים $x_i = (x_{i1}, x_{i2}, \dots, x_{in})$ בסט הנתונים.

- $z_{i,1} = \sum_{j=1}^{n} \emptyset_{j,1} x_{i,j}$ the scores of the 1st PC
- $Maximize_{\emptyset_1} \left\{ \frac{1}{m} \sum_{i=1}^{m} (z_{i,1})^2 \right\}$ מיקסום הוואריאנס
- $subject\ to\ \sum_{j=1}^{n} \emptyset_{j,1}{}^2 = 1$

הבעיה ניתנת לפיתרון ע״י פירוק מטריצת הקו-וריאנס של D לווקטורים עצמיים . ה PC הראשון הוא הוקטור העצמי עם הערך העצמי הכי גבוה.

אם ניקח את מטריצת ה co-variance בין כל הנקודות ונחשב eigenvectors,נמצא מערכת צירים אורתוגונית שממקסמת את ה variance . הווקטור עם השונות הכי גדולה, הוא הווקטור העצמי עם ערך העצמי הגדול ביותר, הווקטור עם הערך העצמי הבא הכי גדול, הוא הציר השני.

מבט נוסף:
אלגוריתם PCA מבצע דחיסה באמצעות קומבינציות לינאריות שממקסמות את יכולת שיחזור הנקודות ע״י מזעור ריבוע ההפרשים (MSE) בין נקודה לשיחזור שלה. השיחזור הוא שוב ע״י מכפלה במטריצה . הPC הראשון הוא ישר שממזער את ריבועי המרחקים של הנקודות לקו. ה PC הראשון והשני פורשים מישור שממזער את ריבוע מרחקי הנקודות מהמישור. המרחב הנפרש ע״י m הPC הראשונים ממזער את ריבוע מרחקי הנקודות מהמרחב שנפרש. כלומר, המישור הנפרש ע״י 2 ה PC הראשונים ממזער את סכום ריבועי המרחקים של הנקודות מהמישור.

נרמול הנתונים:
בד״כ עבור PCA ממרכזים את המאפיינים מסביב ל 0. נירמול כזה ניתן לעשות ע״י הפחתת הממוצע של כל מאפיין. אין חובה לנרמל גם את סטיית התקן. אבל התוצאות של PCA יהיו מאוד תלויות בScale של המאפינים: אם השונות של אחד המאפינים גבוהה במיוחד, PCA ישקלל מאפיין זו יותר מאחרים ויתן תוצאות שונות לפני/אחרי scaling (הכפלה במספר) להבדיל מרגרסיה לינארית שאיננה רגישה להכפלות.

כדי ש PCA לא יהיה תלוי שרירותית בסקאלת המדידה, נהוג ללבצע Scaling כדי ליצור סטית תקן 1.
אבל במידה ו 2 מאפיינים נמדדים באותה סקאלה, יתכן מאוד ולא נרצה לבצע scaling שונה לכל אחד וזאת כדי שההבדלים ב scale יבואו לידי ביטוי.

אין תשובה נכונה לכמה מימדים נרצה להפחית את המידע, אלא מספיק כדי לשמר אינפורמציה חשובה. עבור ויזואליזציה נרצה 2-3 מימדים.

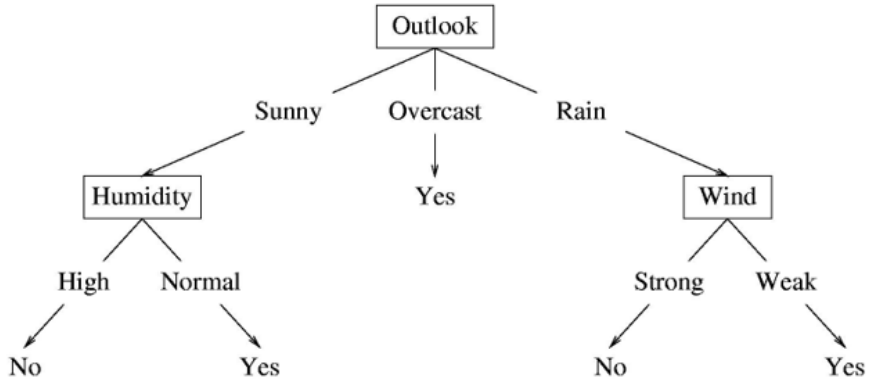
מרצה: ד״ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

עצי החלטה ויער רנדומי:

גישה אלגוריתמית זו טוענת כי ההיפותזה היא עץ החלטה. כלומר, נתונה קבוצת אימון עם דוגמאות ממספר קטגוריות. נחפש שאלה על אחד המאפיינים שתפצל את קבוצת האימון לשתיים (או יותר) קבוצות הומוגניות ככל האפשר. ניצור צומת הכולל את השאלה שנבחרה ונפצל את קבוצת הדוגמאות לקבוצות (ילדים בעץ) בהתאם לשאלה. בצורה רקורסיבית, נמשיך לפצל את כל קבוצות הצאצאים עד שנצליח ליצור עלים הומוגניים (הפרדה מלאה) או שנעצור קודם על פי קריטריון אחר.

דוגמא: אברהם נוהג להזמין את שרה למשחקי טניס. שרה נוהגת להיענות בחיוב או לסרב כתלות בתנאי המשחק. אברהם מעונין לחזות את תשובתה של שרה מתוך ניסיונותיו בעבר. הוא משוכנע שלשרה יש מודל של עץ החלטה בראשה. קבוצת האימון הינה כל אותם דוגמאות שבהם הזמין אותה בעבר. המאפיינים עבור מודל זה הם: $outlook = (sunny, overcast, rain), Wind = (strong, weak), Humidity = (High, normal)$.

היפותזה שהיא עץ החלטה תראה כך:



כל צומת מכיל בדיקה על מאפיין ומתפצל על פי ערכיו. העלים מכילים החלטה על סיווג. בד"כ ישנו שיערוך להסתברות ההחלטה.

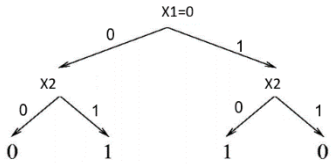
פיצול לפי סוג מאפיין:

מאפיין הינו קטגורי, ניתן לפצל בשתי דרכים:

1. פיצול שהוא יהיה כל הקטגוריות האפשריות
2. פיצול שהוא בינארי על ידי בחירת תת קבוצה של ערכים, למשל, קטגוריה אחת מול כל השאר או לקבוצת ערכים כנגד הקבוצה המשלימה

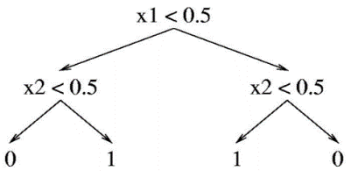
מאפיין נורמי, ניתן לפצל, גם כן, בשתי דרכים:

1. ניתן לפצל אותו באופן בינארי על ידי בחירת סף $x_i < threshold$
2. ניתן להגדיר טווחי ערכים (Bins) ולפצל את העץ כאילו היו קטגוריות בדידות



ייצוג פונקציות בולאניות בעזרת עץ החלטה:

כל פונקציה בוליאנית ניתנת לייצוג בעזרת עץ החלטה. למשל בתמונה, עץ החלטה עבור פונקצית xor.



כל פונקציה בוליאנית ניתן גם לממש בעזרת החלטות על משתנים נומריים וגבול החלטה.עץ ההחלטה שנקבל יוכל להכליל גם כשמאפייני הקלט שאינם רק 0 או 1. גבול החלטה לא לינארי וגמיש מאוד.

עצי החלטה הם מודלים גמישים מאוד. לכל טבלת אמת של n מאפיינים ניתן לבנות עץ החלטה (בינארי שלם) שבו הענפים מייצגים שורות בטבלת האמת. מספר העלים כמספר השורות בטבלה. בהרבה מיקרים ניתן למצוא עצים בעלי פחות עלים ממספר השורות בטבלה. במקרה הגרוע גודל העץ יהיה אקספוננציאלי במספר המאפיינים (משתנים).

מספר עצי ההחלטה:

מרחב ההיפותזות של כל העצים האפשריים הוא עצום. בהנחה שישנם n משתנים בוליאנים ושהעצים הם מלאים, מספר העצים הוא 2^{2^n} . זוהי, כמובן, הערכת חסר כי יתכנו עצים לא מלאים וכן כאשר מרחב הקלט איננו בוליאני, מקבלים אף יותר עצים. באופן תיאורטי, אם המאפיינים מכילים שדות נומריים, מספר העצים הוא אינסופי.

אימון בהיפותזת עצי החלטה – פרוצדורת הלמידה של עצי החלטה:

נחפש את העץ (ההיפותזה) הטוב ביותר. כדי לחפש אותו במרחב החיפוש הענק, נשתמש באלגוריתם חמדני שיאפשר לנו בכל איטרציה להוסיף צומת אחד שנראה לאלגוריתם "הכי טוב".

אלגוריתם זה הוא אלגוריתם למידה לא פרמטרי (ללא משקולות).

מהלך האלגוריתם עבור היפותזת קלסיפיקציה בינארית:

בהינתן סט נתונים, נבנה עץ שמפריד את הקטגוריות כך שהעלים הם הומוגנים (או כמעט הומוגניים).

1. נתונה קבוצת אימון עם דוגמאות חיוביות ושליליות.
2. נניח שמוצאים שאלה אחת על מאפיין מסוים שתפצל את קבוצת האימון לשתי הקבוצות הרצויות. אז סיימנו.
3. אם לא הצלחנו למצוא כזו שאלה, נבדוק שאלות פוטנציאליות ונבצע
   a. נחליט מי יהיה שורש העץ: השאלה שמפצלת את סט הנתונים "הכי טוב"
   b. אחרי הפיצול של השורש, נבדוק בכל ענף איזה מאפיין הוא "הכי טוב" לפצל
   c. בעזרת המאפיין "הכי טוב", נפצל את סט הנתונים לכל ענף ונבנה עץ (באופן רקורסיבי) לכל אחד מהילדים
   d. כך נמשיך הלאה: בכל איטרציה רקורסיבית נוסיף עוד צומת. בצורה רקורסיבית, נמשיך לפצל כל קבוצה עד שנצליח להפריד את שתי קטגוריות (או ע"פ קריטריון עצירה אחר)

האלגוריתם ניתן להכללה גם עבור קלסיפיקציה שאיננה בינארית וכן עבור רגרסיה.

קריטריוני עצירה אפשריים:

1. כאשר ההפרדה מושלמת (תוויות הומוגניות – לא ניתן לפצל יותר)
2. לעיתים לא נוכל להגיע להפרדה מושלמת ואז נעצור אם לא ניתן לשפר את ההומוגניות
3. קריטריונים אחרים – (כמו גודל העץ) - על מנת למנוע התאמת יתר- נעצור כאשר העץ גדול/עמוק מידי

באופו פורמלי:

נתונה קבוצה S של דוגמאות עם מאפיינים בינארים ותווית מטרה בינארית $y = 0/1$.

Grow Tree(S):

if (t=0) for all $< x, t > \ \epsilon\ S$, return (new leaf(0))

אם הקבוצה היא הומגנית מבחינת y, ניצור ממנה עלה, ונסמנו על פי התווית. סיימנו

Else if (t=1) for all $< x, t > \ \epsilon\ S$, return (new leaf(1))

Else

choose "best" feature x_j

אחרת (S איננה הומגנית), נבחר מאפיין שיפצל "הכי טוב" לשתי קבוצות נפרדות. כמה שיותר הומוגניות ב y

$S_0 \ = all \ < x, t > \ \epsilon\ S \ with\ x_j = 0$

$S_1 \ = all \ < x, t > \ \epsilon\ S \ with\ x_j = 1$

return (new node ($x_j < 0.5$,Grow Tree(S_0), Grow Tree(S_1))

נחזיר תת עץ שהשורש שלו הוא השאלה שנבחרה כדי לפצל, ויש לו 2 ילדים שבצורה רקורסיבית ממשיכים להתפצל לעוד קבוצות יותר ויותר הומוגניות עד שהקבוצות "מספיק" הומוגניות או שלא ניתן יותר לפצל

אם ניתן להפריד את קבוצת האימון לקבוצות הומוגניות, האלגוריתם יבנה עץ שיפריד אותם.
לחילופין, ניתן להפסיק לפצל אם השיפור בהומוגניות (או בשגיאה) איננו משמעותי או שיש מגבלות גודל לעץ.
כך מסתפקים בעצים קטנים יותר שנותנים שגיאת וואריאנס קטנה (בהשואה לעצים גדולים).

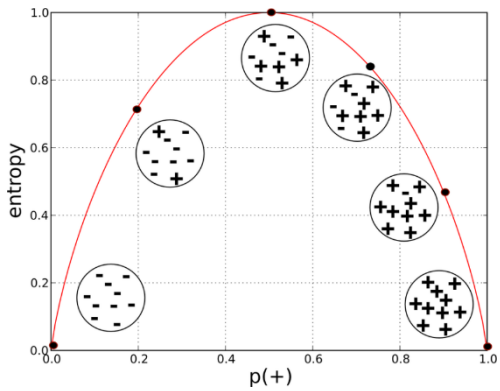
בחירת היוריסטיקה - אנטרופיה:

כיצד נשפוט אם מאפיין קלט מסוים מכיל אינפורמציה חשובה לגבי המטרה (Target label)? כיצד נוכל לדרג את מאפייני הקלט לפי מידת האינפורמציה שהם מספקים? נבצע שימוש בתורת האינפורמציה.

אנטרופיה הוא מדד טוהר/הומוגניות שמודד את כמות אי הוודאות (הפתעה, אי-סדר).

אנטרופיה גבוהה	אנטרופיה נמוכה
✓ הטרוגניות גבוה	✓ הומוגניות גבוהה
✓ אי סדר גדול	✓ סדר
✓ חוסר ודאות גדול	✓ ודאות
✓ הפתעה	

כלומר, אנטרופיה הוא מדד להומוגניות של קבוצה ביחס לקטגוריות המופיעות בה. ככל שיש יותר ערכים בקבוצה, אי הסדר גדל.



באופן פורמלי:

בהינתן משתנה אקראי V (בינארי או לא), האנטרופיה שלו היא:

$$H(V) = \sum_{v=0}^{1} -P(V=v)\log P(V=v)$$

זה הוא ההפתעה הממוצעת בתיאור המשתנה. במילים אחרות, מדד לאי ודאות.

כאשר המשתנה המקרי הוא בוליאני האנטרופיה מקסימלית כאשר הסיכוי $V=1$ הוא 0.5. האנטרופיה מינימלית כאשר הסיכוי הוא 1 או 0 (קבוצה הומגנית). כאשר יש K ערכים, האנטרפיה המקסימלית היא $log(\frac{1}{k})$.

דוגמה לחישוב אנטרופיה ובחירת מאפיינים על סמך אנטרופיה:

נבנה עץ החלטה עבור ההיפותזה האם לקוח יקבל הלוואה מהבנק. בהינתן המידע הבא:

Name	age	gender	Balance ($)	Employed	Default
Mike	42	M	200,000	Yes	No
John	37	M	35,000	No	Yes
Mary	40	F	115,000	No	No
Robert	23	M	72,000	Yes	No
Dora	31	F	29,000	No	Yes

האנטרופיה של חיזוי ברירת המחדל הוא:

$$H(default) = -P_{yes}\log P_{yes} - P_{no}\log P_{no} = -\frac{2}{5}\log\frac{2}{5} - \frac{3}{5}\log\frac{3}{5} \approx 0.97$$

Employed	Default
Yes	No
No	Yes
No	No
Yes	No
No	Yes

כעת, הוחלט כי שורש עץ ההחלטה יהיה מאפיין $employed$. נחשב מחדש את האנטרופיה.

מתוך כל מי שהינו מועסק, אף אחד לא יקבל הלוואה, לכן:

$$H(D, E=yes) = -\frac{0}{2}\log\frac{0}{2} - \frac{2}{2}\log\frac{2}{2} = 0$$

מתוך כל מי שאינו מועסק, שניים יקבלו הלוואה ואחד לא יקבל, לכן:

$$H(D, E=no) = -\frac{1}{3}\log\frac{1}{3} - \frac{2}{3}\log\frac{2}{3} = 0.92$$

לפני הפיצול האנטרופיה הייתה 0.97 ואחרי, קיבלנו שתי אנטרופיות $0.92, 0$. נשאלת השאלה האם המאפיין הנבחר הוא מאפיין טוב לפיצול. לא ברור. לשם כך נעזרת באנטרופיה משוכללת.

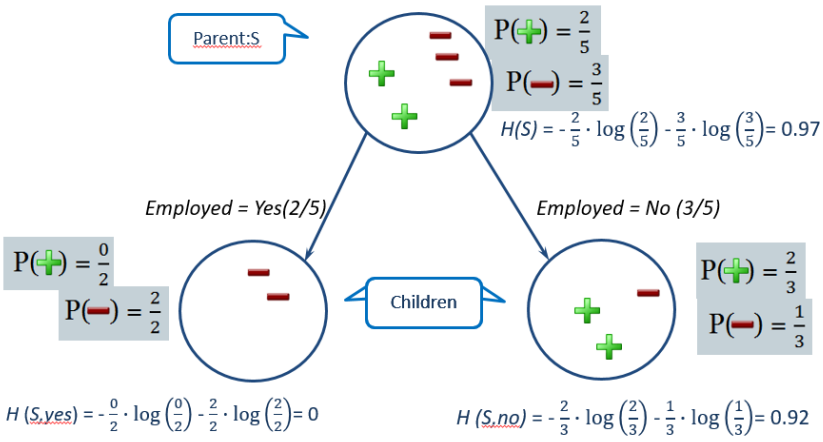
מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

אנטרופיה משוכללת:
המאפיין שנבחר הוא זה שגורם לירידה הגדולה ביותר באנטרופיה המשוכללת.
האנטרופיה המשקללת של מספר קבוצות היא סכום האנטרופיות המשוקללות עי ההסתברות של דוגמא להגיע לכל אחת מקבוצות

$$\sum_{i=1}^{\#children} P(child_i) \cdot H(child_i)$$

את ההסתברות של $P(child_i)$ משערכים באופן הבא:

$$P(child_i) = \frac{|child|}{|parent|}$$



אם נחזור לדוגמה, כעת, נוכל לחשב את האנטרופיה המשוכללת:

$$P(D, E = yes) \cdot H(D, E = yes) + P(D, E = no) \cdot H(D, E = no) = \frac{2}{5} \cdot 0 + \frac{3}{5} \cdot 0.92 = 0.552$$

נשים לב כי חישוב זה הוא בעצם לכפול את האנטרופיה בהסתברות של ההסתעפות בעץ.

Information gain = רווח המידע:
רווח המידע מודד את כמות הפחתת האנטרופיה או כמות הוספת הומוגניות על ידי פיצול מערך נתונים לפי ערך נתון של משתנה אקראי. רווח מידע גדול יותר מצביע על קבוצת אנטרופיה נמוכה יותר או קבוצות של דגימות, ומכאן פחות הפתעה.

לאירועים בסבירות נמוכה יותר יש יותר מידע, לאירועים בסבירות גבוהה יותר יש פחות מידע. אנטרופיה מכמתת כמה מידע יש במשתנה אקראי, או ליתר דיוק התפלגות ההסתברות שלו. להתפלגות מוטה יש אנטרופיה נמוכה, בעוד להתפלגות שבה לאירועים יש הסתברות שווה יש אנטרופיה גדולה יותר.

$$IG(Parent, children) = H(parent) - \sum_{i=1}^{\#children} P(child_i) \cdot H(child_i)$$

רווח המידע תלוי בשני גורמים- באנטרופיה של הבנים $H(child_i)$, וכן כמה הם דומיננטים $P(child_i)$.

עבור הדוגמא, רווח המידע שיחושב הוא $0.97 - 0.552 \approx 0.42$ כלומר, מאפיין "מועסק" הוסיף 0.42 כמות של מידע. או במילים אחרות, המאפיין הוריד את כמות חוסר האי וודאות (האנטרופיה) במשתנה המטרה מכמות 0.97 של אנטרופיה אל 0.552 כמות של אנטרופיה.

בתהליך בניית העץ: לפני הוספת צומת בדיקה נבדוק את רווח המידע של כל המאפיינים שעדיין לא השתמשנו בהם בענף שלו מוסיפים את הצומת החדש. מבין כל המאפיינים, נבחר לפצל את המאפיין עם רווח המידע הכי גדול. כעת ניתן לחדד את האלגוריתם הרקורסיבי של **Grow Tree(S):**. בחירת מאפיין שיפצל הכי טוב לשתי קבוצות שונות יהיה מאפיין שרווח המידע שהוא תורם הוא הגדול ביותר, כלומר, הכי תורם להומוגניות.

בנוסף, ניתן להוסיף קריטריון עצירה לאלגוריתם: שום פיצול אינו גורם לרווח מידע חיובי.
נחשב באופן זהה עבור מאפיינים רב קטגוריים או נומריים אחרי שעברו פיצול מתאים (תואר מעלה).

דוגמה עבור אנטרופיה משוכללת ועץ החלטה עבור משתנים רב קטגוריים- האם נשחק טניס:

Day	*Outlook*	*Temperature*	*Humidity*	*Wind*	*PlayTennis*
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

TABLE 3.2
Training examples for the target concept *PlayTennis*.

נבדוק מה יהיה רווח המידע אם נחליט לפצל לפי $wind = \{strong\ ,weak\}$.

נספור ונראה כי:

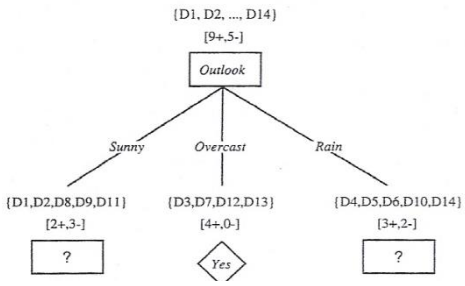
$$S = \{9\ yes, 5\ no\}$$
$$S_{weak} = \{6\ yes, 2\ no\}$$
$$S_{strong} = \{3\ yes, 3\ no\}$$

ואז נוכל לחשב את רווח המידע:

$$H(S) = -\frac{9}{14}\log\frac{9}{14} - \frac{5}{14}\log\frac{5}{14} = 0.94$$
$$H(S_{weak}) = -\frac{6}{8}\log\frac{6}{8} - \frac{2}{8}\log\frac{5}{8} = 0.811$$
$$H(S_{strong}) = -\frac{3}{6}\log\frac{3}{6} - \frac{3}{6}\log\frac{3}{6} = 1$$
$$IG = 0.94 - \frac{9}{14}H(S_{weak}) - \frac{5}{14}H(S_{strong}) = 0.048$$

נוכל באופן זה להחליט מי המפצל הטוב ביותר של S. נבדוק את רווח המידע עבור כל אחד מהמאפיינים ונקבל:

$IG_{temp} = 0.029$ | $IG_{humidity} = 0.151$ | $IG_{outlook} = 0.246$ | $IG_{wind} = 0.048$



נראה כי מזג האוויר משפיע בצורה הכי טובה להומוגניות, תורם הכי הרבה סדר, ולכן נבחר בו לשורש העץ.
כעת, עבור כל צומת פנימית, נחשב רווח מידע לכל שאר המאפיינים ונישקול פיצול נוסף לפי המאפיינים שנשארו $wind, humidity, temp$.

למשל נבצע את החישוב עבור צומת $S_{sunny} = \{2\ yes, 3\ no\}$.

$IG_{temp} = 0.57$ | $IG_{humidity} = 0.97$ | $IG_{wind} = 0.019$

לכן נבחר לפצל צומת זה לפי $humidity$.
לעומת זאת, האנטרופיה של צומת $S_{overcast}$ היא 1, הומוגנית מושלמת, ולכן לא נפצל צומת זה יותר.

נשים לב כי קיימת בעיתיות עבור מאפיינים שהם נומרים רציפים, למשל טמפרטורה. במקרה כזה, נחפש סף שלפיו ניתן לפצל. נבדוק את רווח המידע עבור מספר מקומות סף שבהם ניתן לפצל פיצול בינארי. נסדר את ערכי הערכי המאפיין בסדר עולה (וברשימה נפרדת את ערכי התוויות המתאימים). נבדוק את רווח המידע info gain (=ממוצע הערכים במעבר) רק בנקודות שבהם מתחלפת התווית. מובטח לרווח המידע המקסימלי להיות באחד המעברים. פיצול שאינו במעבר תמיד ייתן פחות הומגניות (אנטרופיה גדולה) מאשר פיצול במעבר.

Temp:	40	48	[	60	72	80	]	90
PlayTennis:	No	No	[	Ye	Ye	Ye	]	No

מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין

ההסתברות (וגם score) לעלה בעץ החלטה :

עבור עץ החלטה לקלסיפיקציה, לכל עלה נקבעת :

1. קטגורית העלה (ע״פ רוב הדוגמאות שבעלה)
2. ההסתברות לגילוי נכון Accuracy של העלה (ע״פ ספירת מספר הדוגמאות החיוביות מתוך סה״כ הדוגמאות שבעלה)

רצוי להחליק את ההסתברות המחושבת כדי להימנע מהסתברויות של 0 או 1. אם אין מספיק דוגמאות וחוששים מהסתברויות 0 או 1, מבצעים תיקונים של ההסתברות ,למשל, בעזרת Laplace Smoothening ואפשר כמובן לקבל את h או G על העלה ולהשתמש במדד לציון או להסתברות.

עצי רגרסיה :

עד כה התעסקנו עם קלסיפיקציה בעצים. ניתן לאפשר רגרסיה בעץ באופן הבא :

- ✓ ממוצע התוויות של הדוגמאות הנופלות בקבוצה (צומת) נלקח כחיזוי y של הקבוצה. כל עלה, לכן, חוזה את הערך הנומרי הממוצע של הדוגמאות שבעלה.
- ✓ $sum\ squared\ error$ של הדוגמאות בקבוצה נלקח כמדד לשיפור (במקום אנטרופיה). כלומר, סכום ריבועי ההפרשים בין התווית לממוצע של הדוגמאות בצומת יהיה מדד השיפור. הרווח של כל פיצול מחושב ע״פ השיפור במדד SSE בעקבות הפיצול.

היוריסטיקות אלטרנטיביות לבחירת מאפיין לפיצול- מדד $Genie$:

ממד זה דומה למדד האנטרופיה וגם הוא מודד אי סדר.

עבור משתנה אקראי בינארי, המדד יחושב כך $G = \sum_{i=1}^{N} p_i \cdot (1 - p_i)$.

כאשר הקבוצות הומוגיות, $G = H = 0$.

בשונה ממד האנטרופיה, כאשר קבוצה היא הטרוגנית, $\max G = \frac{1}{2}$.

כאשר הסתברויות לקטגוריות קרובות לאחד או לאפס, שני המדדים יתנו מספר נמוך (קרוב ל 0 מלמעלה). שני המדדים מקבלים ערך גדול מאפס כאשר יש הרבה קטגוריות בהסתברות זהה,אולם G חסום ב 1 ואילו H יכול להגיע למספרים גבוהים.

יתרונות וחסרונות עבור עצים :

יתרונות :

1. קל להבין ולהסביר ל״לא מומחים״ ישנם המאמינים כי עצי החלטה משקפים טוב את אופן קבלת ההחלטות האנושי
2. ניתן להפיק חוקים (די מובנים לבני אנוש) מענפי העץ ולישם קבלת החלטות שקל להסבירה ע״י הפעלת החוקים בזה אחר זה
3. שגיאת ביאס קטנה : גמישים מספיק כדי לקרב כל פונקציה
4. ניתן לדרג (rank) את ה features לפי חשיבותם (כמה אינפורמציה הם תורמים)
5. ניתן להפעיל על נתוני עתק
6. היסטוריה ארוכה והרבה הרחבות שימושיות

חסרונות :

1. בקלות מבצע התאמת יתר, שגיאת ווריאנס (שליטה מוגבלת בעזרת רגולריזצית מורכבות – על עומק העץ)
2. שינויים קטנים בנתונים עלולים לגרום לשינויים בעץ
3. עלולים להשתמש במאפינים לא רלוונטים ולהיות גדולים שלא לצורך
4. ההיוריסטיקות לבחירת מאפיין, כמו : IG, Genie , לא תמיד מיצרות את העץ הכי טוב
5. האלגוריתם מעדיף עצים קטנים. ולא תמיד מייצר לכן את העץ האופטימלי
6. שינוי קטן בקבוצת האימון יכול להשפיע דראסטית על מבנה העץ

למזלינו, אנסמבל של עצי החלטה פותרים לפעמים חלק מבעיות אלו. אלגוריתם Random Forest הינו היום אחד מאלגוריתמי הקלסיפיקציה הטובים ביותר.

שיטות להפחת שגיאת השונות:

1. עצירה מוקדמת - מפסיקים להצמיח עלים לפני שהגענו לעלים הומוגניים
   - כאשר השיפור באנטרופיה אינו משמעותי
   - כאשר השיפור בשגיאת הווליד‌ציה איננו משמעותי
   - רגולריזציה Complexity: הענשת השיפור באנטרופיה לעצים גדולים או צמתים עמוקים למשל ע"י חלוקת ה IG בגודל העץ: InfoGain/TreeSize.
2. Ensemble: Bagging, Boosting, Random Forest

נתמקד בשיטות להפחת שגיאת השונות בעזרת אנסמבלים.

Bagging using Bootstrap Aggregation=

נבצע דגימה עם חזרות (=bootstrapping) כדי לגדל B עצים שונים המורכבים B מדגמים שונים קצת זה מזה.
נמצע את תוצאות החיזוי של העצים השונים:

1. ברגרסיה ממצעים את החיזוי
2. בקלסיפיקציה משתמשים בהחלטת הרוב או מיצוע ההסתברות של העלה אליו הגענו בכל אחד מהעצים

אפשר לגדל עצים לא מקוצצים שעלולים לעשות התאמת יתר, לכל עץ תיתכן שגיאת שונות גבוהה אשר תתמצע ע"י ה ensemble. בפרקטיקה מגדלים אפילו אלפי עצים ומחברים לפרוצדורת חיזוי אחת.

חסרון בשיטה זו הוא שהעצים דומים מידי אחד לשני ולכן שגיאת השונות אינה יורדת משמעותית
מכיוון שדגימות ה bootstrapping הינן קורלטיביות והצמתים (המשמעותיים) בסמוך לשורש, נשארים לרוב קבועים במקומם בעץ.

יערות רנדומיים random forests=

עצים שגדלים על מדגמים דומים עלולים להיות דומים מבחינת המאפיינים שבהם משתמשים. נרצה לגדל עצים הנבנים מקבוצות שונות של מאפיינים.

1. נבנה עץ לכל מדגם בשיטת bootstrap
2. בכל פעם שבוחרים צומת לפיצול, נגביל את המאפיינים לבדיקה ל $m' < m$ מאפיינים הנבחרים רנדומית מתוך m המאפינים שלרשותנו. כלומר, אם $m' \ll m$, בכל פעם שבוחרים צומת לפיצול, האלגוריתם לא מורשה להסתכל על רוב המאפינים

העצים שנגדל יהיו שונים, גם בגלל שנוצרו מקבוצות אימון שונות במקצת ובעיקר בגלל שאולצו להשתמש במאפינים שונים.

ללא מגבלה על בחירת המאפיינים, ובהנחה שקבוצות האימון דומות, סביר שברוב העצים היה נבחר אותו מאפיין (דומיננטי ברווח המידע) עבור השורש. לעומת זאת, ביער רנדומי, המאפיין ה"חזק" ביותר לא ילקח בחשבון בסיכוי של $\frac{m-m'}{m}$.

באלגוריתם Random Forest העצים פחות קורלטיביים ככל $m' \ll m$.
m' הוא כמובן היפר פארמטר שיש לכוונן.

בהינתן קבוצת דוגמאות S, רשימת מאפיינים מותרים והיפר פרמטר m' , להלן האלגוריתם מגבלה אקראית על מאפיינים הניתנים לבחירה:

<u>Choose Best Feature (S, Feature set, m'):</u>

If Feature set is empty return (Null,Null)

Randomly select m' features from Feature set (if $|Feature\ set| < m'$, select all features in Feature set.

Find the feature Best Feature with the best IG

remove from Feature set,

return(Best Feature, Feature set)

להלן האלגוריתם המלא לבניית עץ בינארי אקראי :

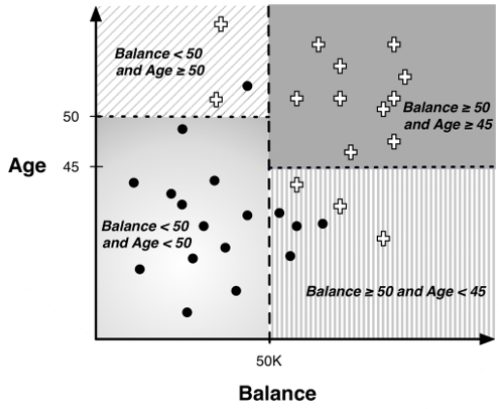
Grow Random Best Tree (S, Feature set, m′):

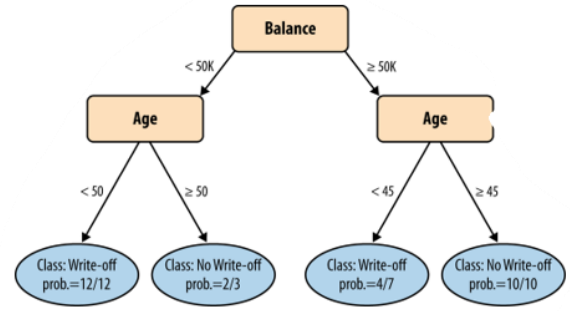
```
If t= 0 for all <x,t> in S, new leaf(0)
Else if t=1 for all <x,t> in S , new leaf(1)
Else
  Best Feature, Feature set =Choose Best Feature(S, Feature set, m')
  if Best Feature ==Null, return
  S0=all <x,t> in S, where XBest=0
  S1=all <x,t> in S, where XBest=1
Return New node(XBest,Grow Random Bset Tree(S0, Fset, m'), Grow Random Best Tree(S1,
Fset, m'))
```

קל להכליל עבור מאפיינים קטגוריים ונומרים.

גבול ההחלטה של עצים :

עץ מחלק את מרחב הקלט למלבנים (היפר קופסאות) לא חופפים.





אם המאפיינים נומרים, ניתן להגיע להפרדה מושלמת.

מסיבה זו, שגבול ההחלטה של עצים יכול לחתוך אופקית או אנכית, נשאלת השאלה האם עצים יכולים ללמוד גבול החלטה לינארי (קו ישר). ואכן, בעזרץ עצי החלטה ניתן לקרב כל פונקציה (להפרדה מושלמת) בעזרת חיתוכים אופקיים ואנכיים. החסרון בקירוב כזה הוא שבהרבה מיקרים אינו מכליל היטב , כלומר מייצר שגיאת שונות. למשל, אם גבול ההחלטה האמיתי הוא לינארי, נצטרך הרבה נתונים כדי שהחיתוכים יתקרבו לקו ישר.

SVM = Support Vector Machines :

אלגוריתם SVM הוא בעצם קלסיפייר בינארי. לשם הבנת האלגוריתם לעומק נבדיל בין מספר מושגים :

1. Maximum Margin Classifier =
קלסיפייר בינארי לינארי על בסיס מיקסום המרווח בין קטגוריות. דורש כי הנתונים יהיו ניתנים להפרדה לינארית (בדומה לפספטרון).
2. Support Vector Classifier =
הרחבה לקלסיפייר בינארי לינארי שאינו דורש הפרדה לינארית מוחלטת ומאפשר גם חריגים (outliers). בנוסף, ישנו פרמטר $capacity$ המאפשר שליטה על Bias – Variance tradeoff.
3. Support Vector Machines =
הרחבה לקלסיפייר בינארי **לא** לינארי על ידי טריק הגרעין (=kernel).

Maximum Margin Classifier:

בהינתן קבוצת אימון D של דוגמאות מסווגות בינארית $\{+1,-1\}$ ממימד n, הניתנת להפרדה לינארית, רוצים למצוא היפר-מישור ממימד $n-1$ המפריד בין הדוגמאות.

במימד n, ההיפר מישור המפריד הוא אילוץ מהצורה :

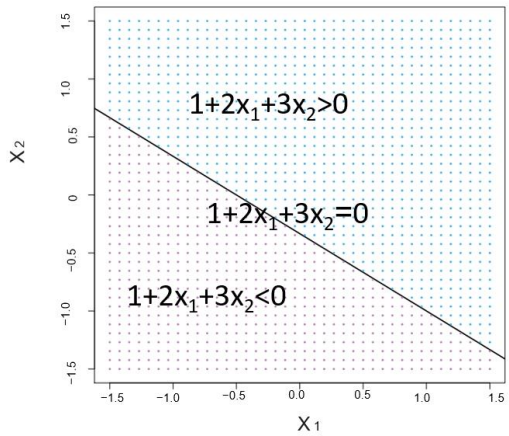
$$h(x) = w_0 + w_1x_1 + w_2x_2 + \cdots + w_nx_n = 0$$

כל נקודה שמקיימת את האילוץ, היא נקודה על המישור המפריד :

- ✓ נקודות מעל למישור יסווגו 1 : $w_0 + w_1x_1 + w_2x_2 + \cdots + w_nx_n > 0$
- ✓ נקודות מתחת למישור יסווגו 0 : $w_0 + w_1x_1 + w_2x_2 + \cdots + w_nx_n < 0$

הסיווג של X הוא לכן : $sign(w_0 + w_1x_1 + w_2x_2 + \cdots + w_nx_n)$.

דוגמא= בהינתן היפר מישור $1 + 2x_1 + 3x_2 = 0$ הסיווג של כלל הנקודות יראה :



היפר מישור מפריד לינארי :

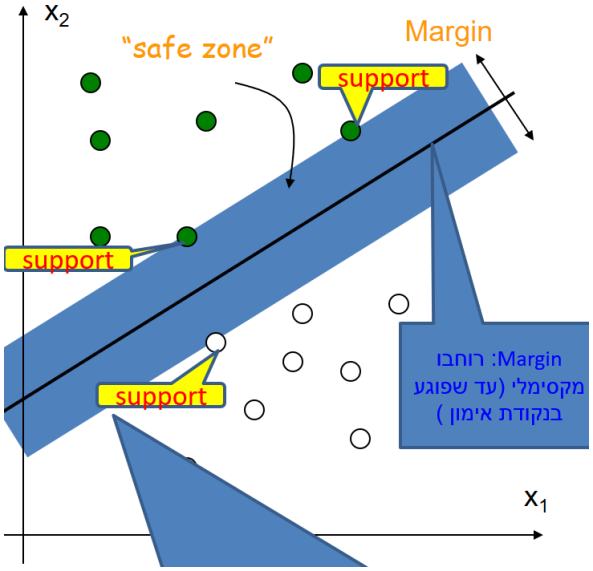
בהינתן קבוצת אימון D של דוגמאות $x_i = (x_{i1}, x_{i2}, \dots, x_{in})$ מסווגות בינארית $y_i \in \{-1,+1\}$ הניתנת להפרדה לינארית, למישור המפריד יש את התכונה :

$$y_ih(x_i) = y_i(w_0 + w_1x_{i1} + w_2x_{i2} + \cdots + w_nx_{in}) > 0$$

המרחק של נקודה x מהמישור, נותן אינדיקציה לרמת הבטחון בסיווג :

$$\frac{y_i(w_0 + w_1x_{i1} + w_2x_{i2} + \cdots + w_nx_{in})}{\sqrt{\sum_{j=1}^{n} w_j^2}}$$

ישנם אינסוף מישורים המקיימים את האילוצים (לכל נקודה). נשאלת השאלה באיזה מישור נבחר.
למשל, בקלסיפיקציה, בחרנו את מישור מפריד שממזער את CE .



עבור SVM נפעיל לוגיקה אחרת:
המישור המפריד בעל השוליים (= Margin) המקסימליים, הוא הטוב ביותר, היות ויש לו יכולת הכללה טובה יותר עבור נקודות מבחן המתקרבות למישור המפריד. בנוסף הוא אינו רגיש למיקום רוב הנקודות שמחוץ לשוליים ולכן פחות התאמת יתר.השוליים מוגדרים כרוחב שבו ניתן להרחיב את ההיפר-מישור המפריד לפני שנתקל בנקודת אימון.

כאשר ממקסמים את השוליים, רק היפר- מישור אחד יכול להיות עם שוליים מקסימליים.

- ✓ אם נקודות האימון מחוץ לשוליים יזוזו, המישור המפריד לא יושפע
- ✓ אם נקודות האימון התומכות יזוזו, המישור המפריד ישתנה

נדמה את מישור ההפרדה (כולל השוליים) כשרוול / צינור ממימד n .

בניית Max Margin Classifier כבעית אופטימיזציה קוודרטית:

$$Maxsimize\ M \quad M,\ w_0, w_1, w_2, \dots, w_n$$
$$subject\ to$$
$$1)\ y_i\ (w_0 + w_1 x_{i1} + w_2 x_{i2} + \cdots + w_n x_{in}) \geq M \quad for\ every \quad (x_i, y_i) \in D$$
$$2)\ \sum_{j=1}^{n} {w_j}^2 = 1$$

משמעות התנאי השני הוא לבחור מבין כל המישורים, מישור כזה עם נורמל היחידה, כלומר המישור מקבל משמעות של מרחק הנקודה מהמישור. הבעיה מנוסחת כך שהפתרון הוא גלובלי ויחיד. כשנקודות אינן ניתנות להפרדה לינארית, אין פיתרון לבעיית האופטימיזציה, הקלסיפייר הזה פשוט לא קיים.

תוספת של נקודה אחת עלולה לגרום לתזוזה דרסטית של המישור המפריד ולשוליים צרים מאוד. זוהי אינדיקציה להתאמת יתר. שוליים צרים מורידים את הבטחון (=המרחק מהמישור) בסיווג של נקודות התמך.

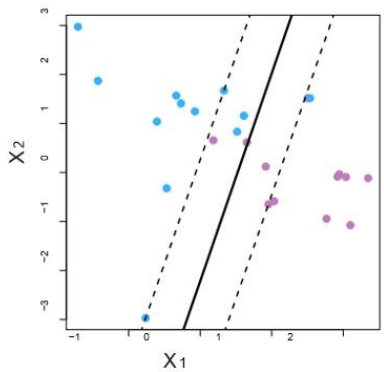
חסרונות בשיטה זו:

1. אם הנתונים לא ניתנים להפרדה לינארית, אין פתרון לבעית האופטימיזציה.
2. הגדרת הבעיה אינה מאפשרת חריגים outliers בכלל.
3. גם אם ניתן להפריד, ישנה רגישות גבוה לנקודות שבחזית (בסמוך למישור המפריד). שינוי במיקום וקטורי התמיכה, כמו תוספת נקודות בגבולות השוליים, יכול לגרום לשינוי דרסטי במישור המפריד ולשוליים צרים מאוד.

Support Vector Classifier:
קלסיפייר שכזה אינו דורש הפרדה מלאה של דוגמאות האימון. כלומר, נחפש soft margin classifier שאינו מכריח כל דוגמא להיות מעבר לשוליים, וזאת על ידי כך שנאפשר שמיעוט דוגמאות תוכלנה להופיע בצד הלא נכון של השוליים מבלי להצר אותו וכן, מיעוט דוגמאות תוכלנה אפילו להופיע בצד הלא נכון של מישור ההפרדה מבלי שישפיעו על מישור ההפרדה ואפילו על השוליים (כלומר, לא נדרוש הפרדה לינארית מושלמת).

על מנת לבצע קלסיפיקציה טובה בנוכחות חריגם ורעש, רצוי לאפשר פרדיקציה שגויה לפעמים. כך נרוויח: שוליים רחבים יחד עם קלסיפיקציה טובה של רוב דוגמאות האימון, סבילות איתנה לחריגים בודדים ונוכל לאפשר מספר רב יותר של נקודות תמיכה. התוצאה: פחות רגישות לתזוזות ונקודות חדשות.

מרצה: ד"ר גדי פנקס
נכתב על ידי: יעלי בר גימלשטיין



הקלסיפייר ה"רך" יסווג את רוב הדוגמאות ברמת ביטחון גבוה (מחוץ לשוליים הרחבים),חלק מן הדוגמאות יסווגו נכון אבל בתוך השוליים ואילו חלק מהדוגמאות יסווגו באופן שגוי.

על כן, יש לנסח מחדש את בעית האופטימיזציה – רוצים למקסם את השוליים אבל עבור שוליים רחבים מוכנים מוכנים "לשלם" שמיעוט נקודות יכנסו לתוך הצינור ואפילו יסווגו כשגויים. הפתרון הוא להוסיף משתנים לבעית האופטימיזציה.

נוסיף משתנה רפיון $Slack\ variable = \varepsilon_i$ לכל דוגמה, אשר מאפשר לה להופיע בצד הלא נכון, כאשר ε_i הוא התשלום שמשלים עבור חריגה מהשוליים. כעת, נרצה למעזר את סכום התשלומים.

- ✓ כאשר $\varepsilon_i = 0$ הדוגמא , תהיה מחוץ לשוליים ותסווג נכון
- ✓ כאשר $0 < \varepsilon_i \leq 1$ הדוגמא תהיה בצד הלא נכון של השוליים אבל בצד הנכון של המישור ותסווג נכון
- ✓ כאשר $\varepsilon_i > 1$ הדוגמא תהיה בצד הלא נכון של המישור המפריד ותסווג שגוי

יהיה לנו תקציב C שאנחנו מוכנים לשלם עבור הנקודות החורגות הללו, ככל שהחריגה גדולה יותר, התשלום ε_i גדול יותר. נרצה להרחיב את השוליים כמה שיותר, אך ובשום פנים לא לחרוג מהתקציב.

הבעיה תנוסח, אם כך, באופן הבא:

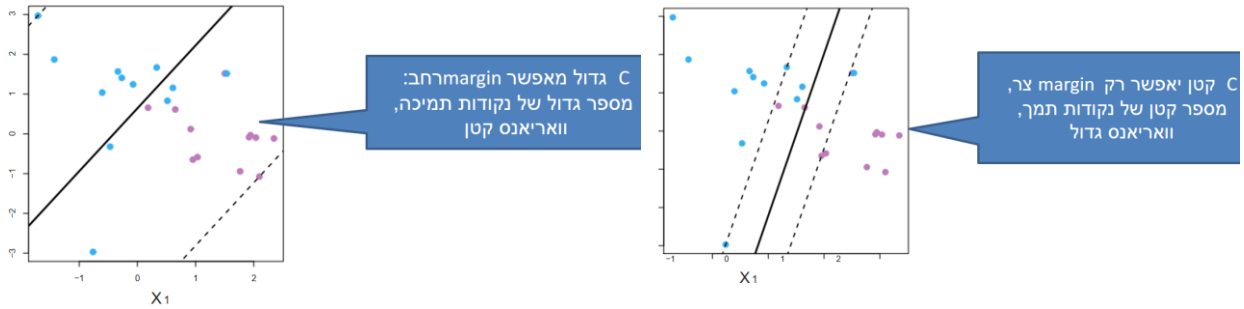
$$Maxsimize\ M \quad M,\ w_0, w_1, w_2, \dots, w_n, \varepsilon_i, \dots, \varepsilon_m$$
$$subject\ to:\ for\ every\ \ (x_i, y_i) \in D$$
$$1)\ y_i\ (w_0 + w_1 x_{i1} + w_2 x_{i2} + \cdots + w_n x_{in}) \geq M(1 - \varepsilon_i)$$
$$2)\ \varepsilon_i \geq 0$$
$$3) \sum_{i=1}^{m} \varepsilon_i < C$$
$$4) \sum_{j=1}^{n} {w_j}^2 = 1$$

בתנאי הראשון, המכפלה $M(1 - \varepsilon_i)$ מקטינה בפועל את השוליים ואפילו הופכת אותם לשליליים אם הדוגמה בצד הלא נכון של המישור.

C הינו היפר פרמטר התקציב המגביל את סכום החריגות מהשוליים M . הוא גם שולט Bias – Variance tradeoff . לא יתאפשרו יותר מאשר C חריגות מהמישור המפריד $\varepsilon_i > 1$:

- ✓ $C = 0$ משמע אין כלל תקציב, כל הדוגמאות צריכות להימצא מהצד הנכון של השוליים. ועל כן חייב להיות $\varepsilon_i = 0$, אם D אינה ניתנת להפרדה לינארית לא יהיה פיתרון
- ✓ נקודה בצד הנכון של המישור אבל בתוך השוליים תיקח מהתקציב $0 < \varepsilon_i < 1$ בהתאם למרחקה מהמישור
- ✓ נקודה בצד הלא נכון של המישור תיקח מהתקציב נתח גדול יותר $\varepsilon_i > 1$ ותסווג לא נכון

תקציב גדול, מאפשר להרבה נקודות לחרוג מהשוליים וכל נקודה שחורגת היא וקטור תמך. כשיש הרבה וקטורי תמך יש פחות רגישות לתזוזות בנקודות אלו, או להוספת נקודות תמך חדשות (שגיאת שונות קטנה יותר). מצד שני, תקציב גדול מאפשר סיווג לא נכון של דוגמאות אימון ולכן עלול להעלות את שגיאת ביאס.

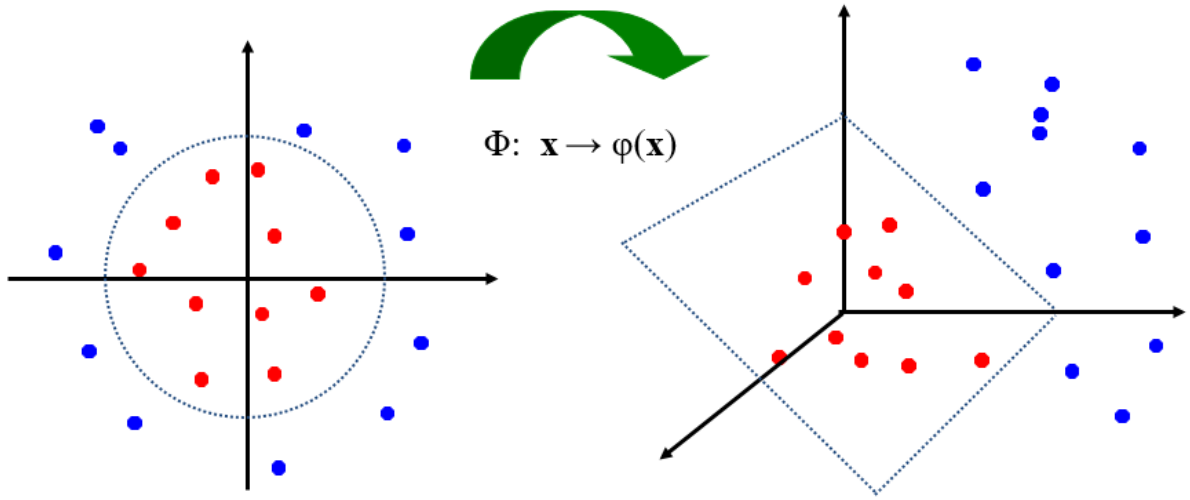


במילים אחרות, התקציב $capacity = c$ הינו היפר פארמטר הקובע כמה נתפשר בדוגמאות המפרות את השוליים על מנת לאפשר שוליים רחבים

- ✓ תקציב מאפשר שוליים רחבים הבנויים על הרבה וקטורי תמך למרות שיהיו הרבה דוגמאות שיפרו את השוליים. מספר וקטורי התמך גדול ולפיכך פחות רגישות לשינויים בוקטורי התמך (פחות שגיאת שונות).
- ✓ תקציב קטן, יצור שוליים צרים, הבנויים על מעט וקטורי תמך. גבול ההפרדה יהיה רגיש יותר לחריגים.כל שינוי בוקטורי התמך ישפיע על המישור המפריד ויוביל להתאמת יתר.

העובדה שרוב הנקודות נמצאות מחוץ לשוליין ולכן אינן משפיעות על קביעת המישור נותנת יתרון בהורדת שונות מול שיטות הרגישות לכל הנקודות כמו עצים. ברגרסיה לוגיסטית ניתן להראות שמצב דומה מתקיים וגם שם נקודות רחוקות כמעט ולא משפיעות על גבול ההחלטה וכנ"ל לגבי KNN : נקודות רחוקות אינן משפיעות. יש דימיון רב גם בהתנהגות האמפירית והתיאורטית של 3 השיטות : KNN, SVM ורגרסיה לוגיסטית.

גבולות החלטה לא לינאריים SVM:
כזכור, הסבת מרחב המאפיינים (למשל לקואורדינטות פולריות, או למרחב פולינומיאלי) יאפשר גבולות החלטה לא לינארים (על המרחב המקורי). פחות יעילים ברוב המקרים. נרצה באופן כללי, למפות את מרחב הקלטים למרחב ממימד גבוה יותר שם הנתונים כן ניתנים להפרדה לינארית.



שיטת SVM משתמשת בטריק kernel שמאפשר הרחבת המרחב בו משתמש הקלסיפייר באופן שהוא בהרבה מיקרים יעיל יותר מבחינה חישובית וללא המרה בפועל של הווקטורים.

מרצה : ד"ר גדי פנקס
נכתב על ידי : יעלי בר גימלשטיין

טריק הגרעין The Kernel Trick :
ההיפותזה :

$$h(x) = \alpha_0 + \sum_{i=1}^{m} \alpha_i \langle x, x_i \rangle$$

מכפלה פנימית היא סוג של פונקצית דמיון בין שתי נקודות, לכן נוכל להחליף אותה בהכללה שלה :

$$h(x) = \alpha_0 + \sum_{i=1}^{m} \alpha_i K(x, x_i)$$

גרעין kernel = פונקציה המכמתת דמיון בין שני וקטורים. הפונקציה צריכה לקיים תכונות מתימטיות של גרעין ואולם לצרכים פרקטיים כמעט כל פונקצית דימיון תהיה בעלת תכונות אלו. קיימים המון סוגי גרעינים, אלו הנפוצים ביותר :

1. גרעין ליניארי, דמיון קורלטיבי

$$Linear: K(x, x_i) = x \cdot x_i$$

2. גרעין פולינומיאלי, גם בצורתו הפשוטה ביותר $d = 2$, נותן הרבה כוח

$$Polynomial: K(x, x_i) = (x \cdot x_i)^d$$

3. גרעין גאוסיאני, נותן דמיון בעל צורת פעמון. הווקטורים דומים אם מרחקם האוקלידי קטן. ככל שהסיגמה קטנה, הגאוסיאן צר, ואילו ככל שהסיגמה גדולה, הגאוסיאן רחב

$$Gaussian: K(x, x_i) = e^{\left(-\frac{\frac{1}{2}\|x - x_i\|}{\sigma}\right)}$$

בטריק הקרנל ניתן להשתמש עבור הרבה אלגוריתמי למידה. מסיבות היסטוריות, חבילות תוכנה כוללות אותו בעיקר באלגוריתם SVM לכן הוא פופולרי בעיקר שם.

דמיון בין אלגוריתם SVM לאלגוריתמים אחרים:

1. דומה לרגרסיה לוגיסטית עם רגולריזצית $ridge$
2. כשמשתמשים בגרעין גאוסי מקבלים קירוב לאלגוריתם KNN

תנאים לשימוש באלגוריתם זה:

אלגוריתם SVM פופולרי במיוחד כאשר :

1. הבעיה ממימד גבוה
2. יש מעט נתונים וחוששים משונות גבוהה
3. רוצים לבדוק מהר גבולות החלטה לא לינארים, זמינות גבוהה של Kernel Trick

אבל קיימות בעיות ביצועים כאשר D גדול מאוד. מטריצת ערכי K (הגרעינים) היא ריבועית במספר הדוגמאות m. אם כי, ישנן ווריאציות עבור קרנלים מסוימים שהמטריצה K לא תהיה ענקית.

אלגוריתם SVM ליותר משתי קטגוריות:

האלגוריתם נבנה עבור קלסיפיקציה בינארית ולא ניתן להרחבה בקלות ליותר מאשר 2 קטגוריות. הדרכים בו ניתן להרחיב הן :

1. One VS One= בנה $\frac{k(k-1)}{2}$ קלסיפיירים, אחד לכל זוג קטגוריות, בהינתן דוגמא x הפעל את כל הקלסיפייר. פלט החיזוי : הקטגוריה בעלת רוב החיזויים
2. One VS. Rest= בנה k קלסיפיירים שמבדילים אם קלט x שייך לקטגוריה i או לכל יתר הקטגוריות. פלט החיזוי : הקלסיפייר שנותן את הביטחון המקסימלי לקטגוריה שלו : $h_i(x)$ המקסימלי.

מרצה : ד״ר גדי פנקס
נכתב על ידי : יעלי בר גימלשטיין