קורס מבוא למדעי נתונים

תוכן

קורס מבוא למדעי נתונים – "אם אתה מבין - זה לא משעמם" 1
שיעור 1 3
פתיחה 3
בסיס ספירה 3
מבוא להסתברות 4
מושגים בmachine learning 4
שיעור 2 4
הסתברות מותנית 4
חוק בייס 4
פול התמנון 6
בעיית מונטי-הול 6
בעיית יום ההולדת המשותף 6
שונות וממוצע 7
שיעור 3 7
המשך סטטיסטיקה והסתברות 7
התפלגות נורמלית 8
משפט הגבול המרכזי – 8
גאוסיאן 9
שיעור 4 11
קורלציה 12
סיבתיות – הרלוונטיות של המודל 12
אינטליגנציה מלאכותית 13
שיעור 5 14
אינטליגנציה מלאכותית ,OUTלמידת מכונה IN 14
תחומי האינטליגנציה המלאכותית 15
data categorizing 16
הציפייה שלנו לראות במודל 17
שיעור 6 17
3 סוגים של אירוסים – הדאטה המפורסם בעולם 17
אלגוריתם -knn – k nearest neighbor 18
שיעור 7 19
שמירה על התפלגות הנתונים 19
רגרסיה לינארית – ערך מספרי 19

רגרסיה לוג'יסטית (מלשון המילה log) – סיווג הלינארית 20
שיעור 8 22
חזרה על שיעור קודם 22
Psauodo Random 22
Train-test.split 22
רגרסיה לינארית - המשך 23
שיעור 9 25
העשרה שלא קשורה לקורס – איך ללמוד במבחנים 25
אלגוריתם k-fold – cross validation 26
ההבדל בין train/test split בין cross validation 26
ייעול הknn על ידי k-fold 26
אידיאליזציה של המודל – לטוב ולרע 27
שיעור 10 27
Grid search 27
Random search 30
מטריקות דיוק 31
שיעור 11 33
Predict probability 33
מודל ROC 33
משתנים קטגוריים 34
שיעור 12 34
לקראת הפרויקט 34
Version control 36
קריאה מומלצת 36

שיעור 1

פתיחה

1. האם יש דרך לדעת מי מהאנשים עלול להיות מחבל מתאבד? אפשר להעלות הרבה השערות אבל הנקודה היא שרק כשיש לך הרבה מאוד נתונים אפשר לחלץ תובנה מאוד מעניינת וקריטית – שלאלו שעשו פיגועי התאבדות אין ביטוח חיים למשל, כי ביטוח לא משלם למי שהתאבד.
כל סטודנט שמגיע לאוניברסיטה מגיע עם "סל" פרמטרים ועובדות שהוא מגיע איתן – מין, גיל, כתובת, השכלה, תכונות, תמיכה כספית ועוד ועוד. אפשר להניח שאם מספר המקומות באוניברסיטה מוגבל אולי ניתן לפתח מודל של מהם ה"קריטריונים" המשמעותיים ביותר שהופכים סטודנט לבעל הפוטנציאל הגבוה ביותר להצליח ברמה גבוהה, ו"לסנן" בצורה שתביא לתוצאה טובה יותר מהדרך המקובלת. אנחנו צריכים להביא את המודל הטוב ביותר. כמובן שיש עניין ערכי למשל שצריך אולי לתעדף בין שוויון בין המינים לבין ציון פסיכומטרי אבל בסופו של דבר כאשר יש לי הרבה נתונים אני יכול להסיק ולפעול איתם בצורה יותר מושכלת כך שגם אם המודל הוא לא מושלם בכל הפרמטרים אני עדיין מסוגל לומר: "אלה הפרמטרים שמשפיעים על המודל לאור השיקול הזה והזה"

בסיס ספירה

1. הבסיס שאנו סופרים הוא בסיס עשרוני (המספרים מ0 עד 9) – אלו עשרה סמלים שונים שמייצגים 10 מספרים, שמהם מרכיבים את כל המספרים האחרים, אין עוד "סמל" שמייצג ספרה נוספת. (המקור של זה – מ10 אצבעות שיש לנו).

הצרפתים סופרים בבסיס 20, (20 אצבעות, כולל הרגליים 😊 את המספר 95 מבטאים בצרפתית כארבע עשרים ועוד 15)
הבבלים למשל ספרו בכפולות של 60 (כי מתחלק ברוב המספרים, לכן גם הרכיבו ממנו את יחידות הזמן – שעה, דקה וכו')
נחזור לבסיס העשרוני – כל מספר ניתן להרכיב מ10 המספרים ולבטא כבסיסים של 10. למשל המספר 321 הוא למעשה $3*10^2+2*10^1+1*10^0$.
נעבור עכשיו לאופן שבו מחשבים סופרים: הספירה הבינארית, שבה היא על בסיס 2, כלומר רק באמצעות 2 מספרים – 0 ו1. הרעיון הוא שאם אני יכול להגדיר "יש מתח" אני יכול להגדיר ע"י 1, ו"אין מתח" ע"י הספרה 0 - אני יכול לייצר מכונה (טרנזיסטור) שתדע לתרגם אותם למספרים ולפעולות שאני מעוניין.
בספירה בינארית זה דומה רק בבסיס 2 בחזקת האינדקס של הספרה בסדר מ0 והלאה, כפול 0 או 1. כלומר המספר **10011** הוא בעצם: $\underline{1}*2^0+\underline{1}*2^1+\underline{0}*2^2+\underline{0}*2^3+\underline{1}*2^4$
והחזקה ממוספרים מ0 וצפונה כך שמה שהמספר הראשון הוא 1, השני 2, השלישי 4, חמישי 8 וכן הלאה, ועם זה ניתן ליצור את על המספרים
10 בבסיס ספירה דצימלי זה 10 מספרים
10 בבסיס ספירה אקסהדצימלי יש 16 סימונים שונים, 1 עד 9 ואז A,B,C,D,E,F, 10 ספירה זו היא למעשה הדרך להתמודד עם הספירה הבינארית כשהיא מגיעה למספרים גדולים יותר. למשל במקום להתחיל לפרש מה זה 1010001101101 ע"י 0/1 בחזקת האינדקס כפול 2, הבסיס האקסה דצימלי בעצם מחלק את המספר לקבוצות של רביעיות מימין לשמאל כך שכל רביעייה יכולה

להיות בשפה בינארית עד 15 (1111 = 15, כך שלמשל 1010=10, 1101=13 שלפי הספירה האקסה דצימלית זה למעשה D).

10 בבסיס ספירה בינארי זה 2

בדיחה לסיום – העולם מתחלק ל10 סוגים – אלו שמבינים בינארית ואלו שלא.

מבוא להסתברות

1. הסתברות להוציא בקוביה מאוזנת 4 היא שישית, אבל הרעיון של הסתברות הוא לא שפלוס מינוס שישית מהפעמים ייצא 4, כי יכול מאוד לקרות שייצא הרבה יותר או הרבה פחות.

המשמעות של המשפט: הסתברות להוציא בקוביה מאוזנת 4 היא שישית – כלומר אם נחזור על הניסוי הזה הרבה פעמים, ככל שנחזור עליו יותר – הערך יתכנס להסתברות. ניתן לתכלל זאת למשוואה: $P(A) = \lim \frac{\#A}{n} = \frac{1}{6}$ (# = מספר הפעמים שיצא A)

2. המודל של הסתברות מתבסס על 2 הנחות: א. חזרתיות (הניסוי מבוצע מספר של פעמים) ב. אובייקטיביות (לא משנה מי יעשה את הניסוי).

מושגים בmachine learning

1. למידת מכונה – יש הבדל בין משהו שיש לו פתרון אנליטי כמו פתרון משוואה מסוימת, וניתן להכניס למחשב למשל נוסחה למשוואה ריבועית והוא יפתור אותם בצורה מדויקת,

בשונה מכך אצל בני אדם יש מודל אחר וזה שככל שהניסיון שלי בתהליך מסוים גדל – כך יש לי יכולת לבצע את המשימה בצורה טובה יותר. למידת מכונה באה לפתור את ההבדל הזה ולגרום למחשב להצליח ללמוד מהניסיון, לשפר את הביצועים, וללמוד מטעויות ככל שיש יותר ויותר נתונים.

שיעור 2

הסתברות מותנית

1. הסתברות מותנית היא הסתברות של אירוע מסוים בהנחה שתנאי אחר קורה.

למשל – מה ההסתברות שקובייה תראה מספר זוגי, בהנחה שהמספר גדול מ3. התשובה היא שני שליש, כי מראש אנו מניחים שזה זוגי, ובתוכו יש 2 אפשרויות – 4/6.

2. מרחב המדגם הוא - קבוצת כל התוצאות האפשריות בניסוי (למשל בהטלת קוביה מרחב המדגם הוא $\{1,2,3,4,5,6\}$ ואנו נכנה אותו אומגה (Ω) P(A) בהינתן P(B).

חוק בייס

3. חוק בייס: $P(A,B) = \frac{p(B/A)\cdot P(A)}{P(B)}$ החוק הזה מטפל בהסתברות על אירועים יחידניים שלא באמת ניתן לנבא. ויש שכותבים את המשוואה הזו כך: $P(H,E) = \frac{p(E/H)\cdot P(H)}{P(E)}$

במצב זה, H=hypothesis E=evidence.

H זה רעיון מסוים שיש לי, למשל היכולת שלי לסיים תואר במדעי המחשב.

E זה העובדות ודברים שיקרו כדי שאבין אחרת, למשל כמה מבחנים אני צריך להיכשל כדי שאני אבין שאני לא מסוג לסיים את התואר הזה?

אם לא קיים E מסוים שישנה את הרעיון אז למעשה זה מוכיח שהרעיון לא מחובר למציאות אלא סתם מחשבה ילדותית.

מה הסיכוי שטראמפ ירוץ שוב לבחירות – אין תשובה חד משמעית אבל גם אם מישהו מאוד בטוח שזה יקרה חייבות להיות עובדות מסוימות שיקרו שישנו את דעתו, למשל אם טראמפ ימות.

דוגמא לחוק בייס: יש בדיקה שבודקת מחלה מסוימת. הבדיקה מדייקת ב99% (לחיוב או לשלילה). מישהו נבחר באקראי מהאוכלוסייה, נבדק ויצא חיובי. מה הסיכוי שהוא חולה, בהינתן שהמחלה נפוצה ביחס של 10,000 :1 באוכלוסייה?

התשובה האינטואיטיבית היא להגיד ש99%. אילו הבדיקה הייתה מדייקת ב100% אז בוודאות הוא היה חולה, אבל העובדה שהיא לא מדייקת משנה את התשובה.

נציב בחוק בייס:

H = האיש חולה

E = הבדיקה יצאה חיובי

אותנו מעניין מה הסיכוי שהאדם חולה בהינתן שהבדיקה אמינה?

מה הסיכוי הא-פריורי (=לא נסיוני - לפני הבדיקה)

נציב במשוואה: $P\left(\frac{H}{E}\right) = \frac{p(E/H)\cdot P(H)}{P(E)}$

P(E/H) – נתון לנו שהאיש חולה מה הסיכוי שהבדיקה אכן חיובית? 99%

P(H) – מה הסיכוי שהאיש חולה? 10^{-4}

P(E) – הבדיקה יצא חיובי, מה הסיכוי שהבדיקה אמינה? לפי המשוואה - נחבר את כל האופציות, ספציפית במקרה שלנו יש רק 2 מקרים:

א. הבדיקה יצאה חיובי - הוא באמת חולה והבדיקה צדקה,

ב. הבדיקה יצאה חיובי – הוא לא באמת חולה והבדיקה טעתה

לצורך זה אנו צריכים את חוק ההסתברות השלימה - $P(A) = \sum_{b_i \in B} P(A/bi) \cdot P(bi)$

יש לי מאורע A שקשה לחישוב, אני פורס את כל מרחב המדגם (ללא חזרות) כך ש-i זה מספר האופציות בשילובים השונים.

P(E) = p(סיכוי שחולה מתוך האוכלוסייה) * p(אכן חולה בהינתן שיצא בבדיקה חיובי) +

p(סיכוי שבריא מתוך האוכלוסייה) * p(לא חולה בהינתן שיצא בבדיקה חיובי)

$= (0.99 * \frac{1}{10,000}) + (0.01 * \frac{9999}{10,000}) = 0.0010098 \approx \frac{1}{100}$

כך שסך הכל נקבל: $\frac{\frac{9999}{10000} * \frac{1}{10000}}{\frac{1}{100}} = 0.00999$

למעשה הסיכוי שהאדם אכן חולה הוא פחות מ- 1%!!

2 דגשים: א. התגלית המשמעותית של בייס היא שניתן להגדיר לגבי אמונה מסוימת/ רעיון שיש לי – **עד כמה** (ברמת מספרית/אחוז) אני מאמין שזה אכן נכון.

ב. להצליח לשים לב האם בחירה מסוימת היא אקראית באמת או לא. רוב הבחירות שאנו עושים

הן לא באמת אקראיות – אם מישהו חזר עכשיו מהמזרח – ועושה בדיקה למחלה מסוימת, לשאול "אם הבדיקה לא אמינה לחלוטין מה הסיכויים שאכן חולה?" זה לא באמת אקראי.

פול התמנון

4. במונדיאל 2014 היה תמנון בשם פול בגרמניה שהצליח לנחש 8 משחקים ברציפות את התוצאה.
היפוטזת 0 הוא למעשה המודל הכי "טיפש" – פול ניחש. שזה הסתברות של 1 ל512.
היפוטזת 1 לפול יש כוח לדעת.
היפוטזת 2 אחד העובדים התערב בבחירה, דבר שאפשר לפסול כי הבחירה הייתה לפני המשחק.
בפועל כולם מסרבים לקבל את העובדה שזה כוח שהיה לו, כי ההתפלגות הא-פריורית היא מאוד מאוד קטנה (=שההסתברות שהעובדה שלפול יש כוח אמיתית לפני שעשיתי ניסוי היא קלושה)
מאחר וזו טענה (H) כל כך משמעותית, אני אצפה לראות תוצאות (E) הרבה יותר משמעותיות.

בעיית מונטי-הול

5. בעבר היה משחק טלוויזיה בשם "עשינו עסק", לפני המתמודד הוצבו 3 דלתות, מאחורי 2 מהן היה דחליל, ומאחורי 1 יש רכב חדש. המתמודד בחר למשל את דלת 3, לאחר מכן הוא היה פותח למשל את דלת 2 ושואל אותה האם הוא מעוניין לעבור לדלת 1. האינטואיציה אומרת שלא משנה כי היחס עכשיו הוא 50/50. אבל האמת היא שזה לא נכון.
ננסה לחשוב הפוך – במידה ומראש לא היו חושפים לי איפה הדחליל, אבל היו נותנים לי אפשרות לבחור 2 דלתות – קל להבין למה הסיכוי שלי גדל פי 2. אבל מה עוד אני יודע? שבוודאות באחת מהדלתות האלו יש דחליל, רק שאני לא יודע באיזה מהן. כעת אם נפתח את אחת מהן ונראה דחליל - הפתיחה של הדלת לא הוסיפה לי מידע, היא רק חשפה לי מבין הדלתות באיזה מהן הדחליל שאנחנו כבר יודעים ששם בטוח.
נחזור עכשיו לדוגמא הראשונה –
האם שווה לעבור לדלת 1? כן, אם בחרתי את 3 וחשפו לי את 2, ואני עובר ל1 למעשה אני עובר לדלת 1 ו2! ש66% שב1 ו2 יש מכונית, ו100% שיש דחליל, לכן גם הסיכויים שלי ב1 הם שני שליש.

בעיית יום ההולדת המשותף

6. אם יש לנו 40 אנשים בכיתה מה הסיכוי שיש 2 עם אותו תאריך לידה? התשובה האינטואיטיבית תהיה שסיכוי לא גדול, אך האמת היא שכמעט ואין סיכוי שלא יהיו 2 עם אותו תאריך.
הבעיה היא שאנחנו מסתכלים על זה באופן פרטני – " מה הסיכוי שמישהו נולד ביום שלי" אם אנו מסתכלים על זה כך זה באמת לא סביר להניח אבל האמת היא שזה לא נכון להסתכל על זה.
הסיכוי ש2 אנשים לא נולדו באותו תאריך הוא 364/365, ומכאן נכפיל במונה שילך וירד ככל שנוסיף עוד בן אדם, ונכפול ביניהם (כי זה יחס של 'גם'), אם כן מהכפלה למשל של 40 אנשים:

$$\frac{364}{365}*\frac{363}{365}*\frac{362}{365}*\frac{361}{365}*\frac{360}{365}\dots\dots\frac{325}{365}\approx\frac{1087}{10000}=10.87\%$$

זאת אומרת 10.87% שזה לא יקרה. זאת אומרת כמעט 90% שזה כן יקרה!

שונות וממוצע

7. יש הבדל בין מספר ממוצע לבין מספר חציוני. אם נתאר ממוצע וחציון של משכורות בחברה מסוימת הממוצע סביר להניח יהיה הרבה יותר גבוה כי הוא מכיל משכורות של אנשים שמרוויחים המון ולכן לא מעיד על משכורת אופיינית.
לכן כדי "לפתור" את בעיית הממוצע נגדיר מושג חדש בשם "שונות" – הממוצע של המרחקים בריבוע מהממוצע.
סימון של ממוצע יהיה באופן הזה: $\bar{x} = \frac{\sum(xi)}{N}$
לכן נוסחת השונות תהיה:
$\mathrm{Var(x)} = \frac{\sum(xi-\bar{x})^2}{N}$ (הממוצע של כולם – $\bar{x}$, הערך של אינדקס מסוים – xi)
לפה נכנס מושג ה"סטיית תקן" - std ("standard deviation") שהוא שורש ה(var(x, ונסמן כך:
$\sqrt{\mathrm{Var(x)}} = std(x)$

<u>סטיית תקן למעשה היא שורש השונות</u>
סטיית התקן באה להסביר בכמה סביר להניח התוצאה שנקבל תהיה רחוקה/ שונה מהממוצע.
למשל עבור הקבוצה.
(הערה – בפייתון לא צריך לחשב את הנוסחה אלא יש פונקציה המקבלת מערך ומחשבת את סטיית התקן: np.std(array_2d))
בדיחה לסיום – סטטיסטיקאי טבע באגם שהגובה הממוצע בו הוא 90 ס"מ.

סטטיסטיקה מסתכלת על העבר, הסתברות מסתכלת על העתיד בהסתמך על נסיון העבר.

שיעור 3

המשך סטטיסטיקה והסתברות

1. שני שחקנים משחקים משחק מזל שבו יש יחס זהה לנצח ולזכות בסכום של כסף. המנצח הוא זה שמגיע ל6 ניצחונות. מסיבה מסויימת הם היו צריכים לסיים את המשחק, באותו שלב התוצאה הייתה 3:5. איך נכון יהיה לחלק את הכסף? התשובה האינטואיטיבית תהיה $\frac{3}{8},\frac{5}{8}$.
אבל אם נחשוב לעומק אולי חצי חצי? כי אף אחד לא ניצח, אבל משהו לא מניח את הדעת, כי באמת התשובה היא $\frac{1}{8},\frac{7}{8}$. כי אילו הם היו ממשיכים לשחק מה ההסתברות שהשני ינצח? אם ינצח בראשון, גם בשני וגם בשלישי. כלומר חצי בשלישית. אם נסתכל בכל סיטואציה אחרת הראשון יינצח, לכן החלוקה תהיה $\frac{1}{8},\frac{7}{8}$. היחס הזה מתאר/ מנבא נכון יותר מה היה יכול להיות בעתיד.

2. סטטיסטיקה מגיעה מהמילה state, משמע המסקנות או התובנות לניהול מדינה. לצורך זה משתמשים במדגם, לוקחים כמות מסויימת של אנשים / עצמים המבטאים באופן יחסי את התפלגות האוכלוסייה כך שלפי התוצאות מהם אדע לנבא את האמינות ל100% מהאוכלוסייה.
אלמנט נוסף שיש לקחת בחשבון היא עד איזו רזולוציה אני מעוניין לרדת בחלוקה שלי לקבוצות

השונות המייצגות.

מצד שני של בחירת מדגם עומד בחירת משתנה אקראי. משתנה אקראי הוא התהליך של לקיחת מאורע הסתברותי והצמדת (קביעת) ערך מספרי אליו.

3. משתנה אקראי – משתנה אקראי הוא למעשה התהליך של לקיחת מאורע הסתברותי והצמדת ערך מספרי אליו. נרצה להוסיף מושג נוסף שקשור למשתנה האקראי:
תוחלת, או באנגלית expected value – כך שאם נרצה למצוא את התוחלת של משתנה אקראי מסוג קובייה עם 6 פאות – נבין כי ההסתברות לקבל כל אחד מהמספרים הוא שישית, ו6 אופציות. לכן אם נחשב לפי הנוסחה של expected value: $\sum x * p(x)$ (ההסתברות כפול כל אחד מהתתי קבוצות/צירופים האפשריים)
נקבל - $1*\frac{1}{6}+2*\frac{1}{6}+3*\frac{1}{6}+4*\frac{1}{6}+5*\frac{1}{6}+6*\frac{1}{6}=3.5$
אבל זה לא הגיוני, כי המספרים הם 1 עד 6 ואיך הירך הצפוי שלי הוא 3.5? התשובה היא <u>שכאשר ניקח 2 קוביות הערך יגדל פי 2</u>, זאת אומרת שexpected value יהיה 7. ואם אני אזרוק 100 קוביות ואסכום את התוצאות שהן מראות לי אני אצפה לקבל 3.5*100 כלומר בערך 350. (כמובן שיכולים להיות מצבים שזה לא יקרה, אבל אם אני אעשה את החזרה הזו המון פעמים ובכל פעם אזרוק 100 זריקות – אוכל לראות באופן יותר ברור את הexpected value הזה. כעת נוכל להבין מושג חשוב מאוד – התפלגות נורמלית).

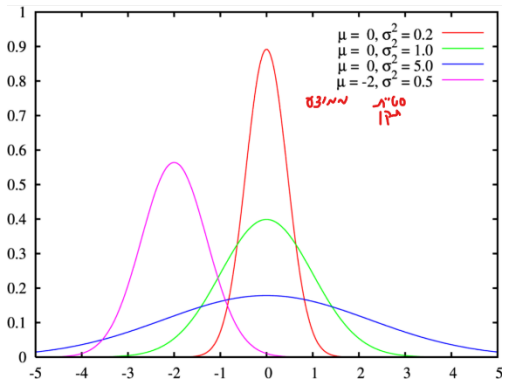
4. נחזור על 2 ערכים משיעור קודם: ממוצע יסומן בסימן מיו μ ויבוטא בנוסחה: $\bar{x}=\mu=\frac{\sum(xi)}{N}$
סיגמא קטנה σ משמעותה השונות, השונות היא למעשה הסטית תקן. $\mathrm{Var(x)}=\sigma^2=\frac{\sum(xi-\bar{x})^2}{N}$
כך שהסטיית תקן היא שורש הסיגמא קטנה.
(הערה – הסיבה שעושים בריבוע היא מ2 סיבות – 1 כדי שכאשר נצייר גרף מסוים הוא יהיה ניתן לגזירה. 2 כדי להדגיש את הסטייה כאשר יש איבר הרחוק מהקו לינארי של התפלגות התוצאות).

<u>התפלגות נורמלית</u>

5. קובייה מתארת התפלגות אחידה, הסיכוי לקבל כל אחד מהתוצאות היא שווה בדיוק לשישית. עם זאת לא כל חלוקה היא שווה - אם הגובה הממוצע של הסטודנטים זה 1.78 מ', רוב הסיכויים למצוא אנשים סביב ה1.78 מ' מאשר כאלו 2 מ' או 1.5 מ'. זה המשמעות של התפלגות נורמלית, זה שהרוב יהיו 'נורמליים', ובודדים יהיו בקצוות ("לא נורמליים" – 2 מ' או 1.5 מ')

<u>משפט הגבול המרכזי –</u>

6. אם אני לוקח מספר מסוים של התפלגויות ומחבר אותן אני אקבל התפלגות נורמלית. למשל מה משפיע על גובה של בן אדם? הגנים, כמות שעות שינה, תזונה, ועוד ועוד. חוק הגבול המרכזי קובע שגם אם המשתנים אינם תלויים – אינם מתפלגים בצורה נורמלית (כלומר שיכולים להיות כאלו בקצוות) – השילוב שלהם יוצא התפלגות נורמלית.



7. אז ראינו שבזריקה של קובייה אחת ההתפלגות שווה לכל התוצאות. אך אם ניקח 2 קוביות נוכל להבין את העקרון של משפט הגבול המרכזי. כאשר אני זורק את הקוביות הן לא תלויות לא אחת בשנייה ולא בשום נתון אחר. אך עם זאת ישנה התפלגות שאינה שווה במקרה זה, כי כדי לקבל 2 או 12 זה הסתברות של $\frac{2}{36}$ לכל אחד מהם, אבל לקבל 7 זה כבר $\frac{1}{6}$. לכן ישנה התפלגות נורמלית סביב 7, וניתן גם לייצר את זה כגרף.

למה חשוב לי התפלגות נורמלית? <u>כדי לזהות איפה הנורמה, כדי להצליח לסווג במה נכון להתחשב ומה כנראה לא מעיד</u> או אפילו הייתה טעות בחישוב/ ברישום (למשל אם ברשימת הגבהים יש מישהו 4.5 מ')

<u>גאוסיאן</u>

8. גאוסיאן הוא למעשה הייצוג המתמטי של ההתפלגות הנורמלית $f(x) = e^\wedge\ (\frac{-(x-\mu)^2}{2\sigma^2})$ (נזכיר -σ זה הסטיית תקן, σ^2 זה השונות) באופן זה יתקבל לי פעמון גאוס כך שהממוצע (μ) יהיה באמצע (בחלק המקסימום) של הפעמון, שהפעמון יתאר לי את ההסתברות לקבל כל תוצאה.

הערה נוספת לחידוד ההבנה - σ^2 משמעותה השונות – var(x), שאם נעשה לזה שורש נקבל את הסטית תקן. נבין כי ככל שהסטיית תקן יותר קטנה כך יש פחות שונות כלומר אני פחות נותן מקום ל'חריגים' להשפיע לי על המודל ואז הפעמון שנראה יהיה הרבה יותר גבוה וצר. בעוד שסטיית תקן גדולה משמעותה לתת יותר מקום לשונות ואז הפעמון שנראה יהיה הרבה יותר משוטח ופרוס)

9. הסברנו על סטיית תקן גבוהה או נמוכה ומה היא משפיעה – הם אני רוצה לתת 'ייצוג' לחריגים או לא. הרבה מאוד פעמים אנו מתארים תהליכים בהם ישנה השפעה חיצונית ואז ההתפלגות הנורמלית לא נכונה.

ההתפלגות הנורמלית עושה הרבה סדר בדברים שקשורים לטבע או לחוק ברירה טבעית וכד' אבל יש לה 2 בעיות:

א. הבעיה הראשונה שלה שהיא טוענת שרוב האחרים הם ליד הממוצע, אבל בחיים אנו מבינים כי זה לא תמיד ככה

ב. הבעיה השנייה שלה זה שהיא לא מתייחסת לקיצוניים, וגם כאשר היא מתייחסת הם משפיעים ברמה כל כך מזערית שלא באמת ניתן לומר שמשפיעה, כדי שהקיצוני ישפיע צריך שהוא יהיה פי עשרות/מאות/אלפים מהממוצע כאשר מדובר על קבוצה עם איברים רבים.

למשל הגובה הממוצע של סטודנטים באוניברסיטה הוא למשל 1.78 מ', אם יצטרף סטודנט בגובה 2.35 מ' הממוצע כמעט ולא יזוז. דוגמא נוספת –האיש המהיר ביותר בעולם – אוסיין בולט קבע את שיא העולם ב100 מ' – 9:58 שניות. אבל , הזמן הממוצע לגבר בריצת 100 מ' הוא בערך 14 שניות, כל זה כי ככה זה עם דברים שמתפלגים נורמלית – כלומר במסגרת הטבע. לעומת זאת כמה כסף יש לסטודנט ממוצע בעו"ש, לעומת הסטודנט הכי עשיר בעולם – הוספת הסטודנט הזה תעלה את הממוצע באופן שיהיה ניתן לראות, למה? כי יכול להיות שהוא עשיר פי אלף, וזה כבר קיצוני שכן משפיע.

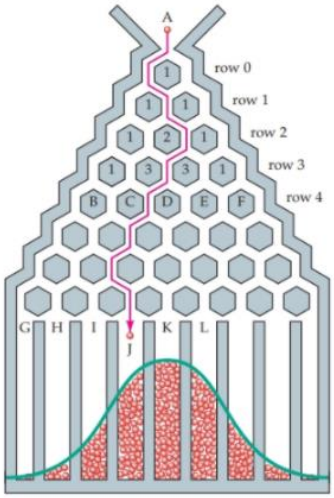
לכן כשאנחנו עובדים על הDATA שלנו אנחנו צריכים לחלק בין מאפיינים טבעיים – להם נכון להסתכל כהתפלגות נורמלית, ויש DATA – מאפיינים שאינם טבעיים (תחת מסגרת הטבע) שלא נכון להסתכל עליו כהתפלגות נורמלית.

הבהרה לסיום – חשוב להבין שלא לכל סטטיסטיקה יש מסקנה מסוימת. אם נעשה גרף של כל האנשים שמתו לאורך השנים שנחנקו במהלך השינה מהסדין, אין משמעות לכך אם הגרף הזה ייצא זהה לחלוטין לגרף של כמות הסרטים של ניקולס קייג' לאורך השנים.

שיעור 4

co-variance

1. בשיעור הקודם למדנו על משפט הגבול המרכזי, שכאשר יש לנו אוסף של משתנים בלתי תלויים – בכל זאת ניתן לייצר חלוקה מסויימת של התפלגות למרכז.



המדען פרנסיס המציא מכשיר הנקרא תיבת גולטון שנועד להראות שההתפלגות הבינומית שואפת להתפלגות הנורמלית כאשר הפרמטר n (מספר ההטלות בבינומי) גדל. המכונה מורכבת מלוח אנכי ובו שורות מופרדות של סיכות. כדורים נזרקים מלמעלה, ומקפצים לשמאל ולימין כאשר הם נתקלים בסיכות. לבסוף, הם נאספים לתוך מיכלים, כאשר כל מיכל הוא ברוחב של כדור אחד. הגובה של עמודות הכדורים בתאים בסופו של דבר ייצור עקומה בצורת פעמון גאוס. (להרחבת אופקים - משולש פסקל מתאר את מספר המסלולים האפשריים כדי להגיע לכל מיכל דרך הסיכות).

ננסה לחשוב - עבור כדור מסוים אני לא יודע להגיד מה יקרה אבל עבור המון כדורים כן – הדבר הזה נוגע גם לחברה, אני לא יודע להגיד בהכרח חיזוי על אדם מסוים אבל על קבוצה של אנשים יש יותר יכולת לדעת.

כלל שנדבר עליו בהמשך זה שבמודלים רבים שבהם פעמון גאוס בא לידי ביטוי אנו נשאף להגיע לסטיית תקן 1 ולממוצע 0.

דוגמא למודל כזה הוא הפסיכומטרי, בו הציונים בכל תחום הוא בין 50 ל150, אבל הציון בכל תחום מגיע ביחס לאנשים האחרים שניגשו איתך, ולכן אמנם ישנה איזושהי סטיית תקן מסוימת אבל בגדול ישנה חלוקה גאוסית של התוצאות.

עכשיו שאלה – האם יש קשר בין הציון המילולי לכמותי? האם בהינתן מישהו עם ציון כמותי, האם אני יכול לדעת משהו על הציון המילולי שלו?

הדבר הזה נקרא co-variance כלומר השתנות יחד, השתנות של גורם אחד המשפיע על גורם אחר.

לפעמים ל2 משתנים יש קשר כל כך קרוב עד שניתן להתייחס אליהן כדבר אחד, אז איך נחשב?

המשוואה של הco-variance היא: $\mathrm{Cov}(x,y) = \frac{\sum(xi-\bar{x})(yi-\bar{y})}{N}$

כאשר נקבל את התוצאה של המשוואה - ככל שהתוצאה יותר גדולה בחיובי זה אומר שהקשר מאוד חזק ותלוי אחד בשני.

ערך גדול של המשוואה במינוס יוצר לנו שיש קשר מאוד גדול אבל בהיפך – אם אני למשל טוב מאוד בכמותי אז אני אהיה פחות טוב במילולי.

אם נקבל 0 זה אומר שאין שום אינדיקציה בכלל לדעת מה הקשר ביניהם.

(הנוסחה בפייתון - נקבע array ל2 המשנים ונעשה np.cov(x,y))

קורלציה

2. הבעיה בco-variance זה שמתקבלים מספרים שאין לי אינדיקציה לדעת עד כמה הם גדולים או קטנים כי יכולים לצאת גם מספרים של כמה אלפים למשל, לכך משמש המושג של co-variance מנורמל – או בשמו השני קורלציה, שזה בעצם לחלק את כל ההתפלגויות בין 1 ל1-.
כמו שהסברנו קורלציה זה למעשה "עד כמה המשתנים תלויים ו'זזים' יחד", אז קורלציה חיובית משמעותה משפיעים אחד על השני ומשתנים יחד לאותו כיוון, וקורלציה שלילית משמעותה משפיעים אחד על השני ומשתנים יחד בכיוונים מנוגדים – ככל שאחד יגדל לכיוון אחד השני יגדל לכיוון השני.
3. כלל אצבע בקורלציה – כדי שתוצאה תהיה יותר ממקרה נדרש שתהיה התאמה של יותר מאשר $\frac{1}{\sqrt{n}}$ כך שn הוא מספר הנדגמים.

סיבתיות – הרלוונטיות של המודל

4. שאלה חשובה שנדרש לשאול – האם במידה ויש קורלציה חיובית האם יש קשר של סיבה ותוצאה, מאוד יכול להיות שכן אבל חשוב לשים לב כי יכול להיות גם שהתובנה/ המסקנה היא הפוכה/ לא קשורה.
נתון – אנשים שעונים על סקרים חיים יותר שנים, האם ניתן להסיק שיש קשר של סיבה בין הנתונים? שאם נענה על סקרים נחיה יותר / שאלו שחיים יותר עונים על יותר על סקרים?
התשובה היא שלא, אבל כן יש קשר מסוים שאינו סיבתי ישיר. לפני שנסביר חשוב להבין – למה חשוב לדעת שאין קשר סיבתי? כי מאוד קל להצליב נתונים, לשים אותם זה מול זה, אבל השאלה היא – האם הנתון/ המידע שיש לנו ביד הינו רלוונטי ומעיד על משהו. כי לא כל מידע הוא רלוונטי או בכלל קשור.
נחזור לדוגמא – יש אמנם קשר בין 2 העובדות הללו אבל לא קשר ישיר -כי אפשר לומר שאנשים שבאופי שלהם ובאורח החיים שלהם הם אנשים יותר רגועים, כנראה (לא בטוח, אבל אולי גם) דואגים לשמור על תזונה נכונה, ולעשות ספורט, וככל הנראה הם לא בלחץ בעבודה או בחיים באופן שיהיה להם לחץ דם גבוה לדוגמא ועוד ועוד, ובנוסף לכל אלו מאחר ואלו אנשים שיש להם יותר סבלנות - יש להם גם סבלנות לענות לסקרים.
הפילוסוף דייויד יום אמר שבטבע אין קשר של סיבתיות, יש קשר של יחס, שדבר אחד קורה אחרי דבר שני, אבל לא קשר של סיבה שדבר אחד גורר דבר אחר.
5. אם אני רוצה לבנות מודל של גובה הטיפ בהתאם לנתונים מסוימים וכך אני אדע מהם התנאים האידיאלים כדי לקבל את הטיפ הגבוה ביותר. למשל: גובה החשבון, כמות הסועדים, מעשנים/לא, שעת הסעודה, מין המלצר וכד'. יש לי המון נתונים שאני צריך להתחשב בהם, ולא נקפוץ למסקנות אם נתון מסוים התקיים. למשל אם נגלה שנשים מקבלות בממוצע יותר טיפים מגברים נסיק ש'עדיף' להיות מלצרית, אבל יכול להיות שזה בכלל לא הגורם המשפיע, אם למשל רוב הנשים עובדות במשמרת ערב, בעוד שרוב הגברים עושים משמרת צוהריים. במצב זה יכול להיות שהמסקנה היא לא שנשים מקבלות יותר טיפים אלא שכאשר אנשים מגיעים באמצע היום למסעדה יש להם אולי פחות סבלנות בגלל העומס של היום ולכן פחות נותנים טיפים. לכן חשוב הקשר הסיבתי – להבין בדיוק מה גרם למה ולשאול האם יש זווית נוספת שלא בחנתי שתשנה לי את המסקנה.

הרבה פעמים אנשים חושבים שאם מקבלים נתונים גולמיים ומזה מפיקים תאוריה, אבל זה לא נכון. כי אם אתה לא יודע מה אתה מחפש אתה יכול להגיע לכל מסקנה שבעולם. צריך להגיע עם טענה מסוימת ואז אבדוק אותה בנתונים שלי האם היא עומדת במבחן המציאות.
אני לא אבדוק ממוצע, כי יש קשר בין הטיפ לבין גובה החשבון ששולחן הזמין, יכול להיות שיצא כך ויותר בנות קיבלו באקראי את השולחנות עם החשבונות הגדולים יותר. לכן נתון נכון יותר הוא מה אחוז הטיפ ביחס לגובה החשבון ואז בחלוקה של בנות ובנים.
לכן כדי שהתובנות שאגיע אליהן יהיו אמינות אני צריך:

1. להגדיר את הטענה שלי – להחליט מה אני מחפש.

2. לבדוק האם זה בא לידי ביטוי בנתונים.

ואז או שהנתונים יפריכו את הטענה או שיאשרו אותה.
לבוא סתם לנתונים ולהריץ גרפים והשוואה של נתונים, "שהנתונים יספרו לי את הסיפור שלהם" זה אידיוטי כי או שניתן יהיה בקלות להביא מהנתונים מסקנה אחרת, או שהמסקנה שנפיק מהנתונים תהיה גולמית וצפויה. אם אתה לא מצפה לראות משהו אתה לא יכול להיות מופתע מהתגלית – שצדקת או לא. לפני שנחת בפעם הראשונה אדם על הירח יכול להיות שחשבו שגם שם יש כוח משיכה כרגיל כמו שאנו מכירים, רק בגלל שישנה הנחה – ברגע שנחת אדם על הירח וגילו את ההיפך יש מקום למסקנות ולמידה וכו'.

הבנה לסיום – תיתן לחכם מידע תהפוך אותו ליותר חכם, תיתן לטיפש מידע אתה תבלבל אותו.

אינטליגנציה מלאכותית

6. רוב האנשים שמדברים על אינטליגנציה מלאכותית מדברים על שימוש ברשתות נוירונים כמו שיש במוח האנושי ו"להלביש" אותם על מוצר מלאכותי. מחשבון שיש לכולנו מסוגל לעשות פעולות וחישובים שבן אדם לא יוכל לעשות אותם בראש, אבל אנחנו לא נאמר שלמחשבון יש אינטליגנציה או יכולת הבנה. לכן נדרש להבין מהי באמת אינטליגנציה מלאכותית.
הפער הכי גדול בנושא הוא שיש פעולות שבן אדם מעולם לא יוכל לעשות, מצד שני פעולות שאפילו ילד קטן מסוגל לעשות ומחשב לא מסוגל. מזה שאתה רואה בן אדם עושה תנועה מסוימת אתה יכול להבין מה הוא מרגיש, או אם מישהו אומר מילה לא קשורה למה שהוא מסביר אתה יודע 'לסנן' את זה בראש ולהגיד 'כנראה הוא התבלבל' או 'המילה לא קשורה'. ישנה טענה כי רק 5% מההתנהלות שבני האדם עושים נובעת בסיס חשיבה רציונלית, ו95% מההתנהלות שלנו נובע מהתנהלות אינטואיטיבית או אינסטינקטיבית. אבל הIQ של בן אדם נובע מ5% האלו.
כשמדברים על אינטליגנציה מלאכותית אנו מעוניינים "ללמד" את המחשב את ה95% האלו של האינטואיטיביות.
נשאל שאלה – כמה קומבינציות של קובייה הונגרית ניתן להרכיב? התשובה היא עצומה, עשרות מיליארדים, ועדיין יש ילדים שיכולים לפתור קובייה הונגרית. דוגמא עוד יותר פשוטה – מחשב מצלם מספר שרשום על לוח – עד לא מזמן הוא לא ידע לפענח מה כתוב שם, אבל ילד בגיל 5 כבר יודע לזהות את המספרים.
7. **מה מייחד אדם ממחשב? הלמידה מניסיון** אנו הולכים ולא באמת שמים לב שאנו מעבירים את רגל שמאל ואז ימין ושוב.. אנחנו עושים דברים שלא בהכרח אנחנו מבינים איך אנחנו עושים את זה. איך נסביר נוף לבן אדם עיוור? אנחנו יכולים לראות משהו מסוים עשרות פעמים ולא לראות

פרט מסוים בו, ואז כשיגידו לנו על הפרט ההוא אנחנו לא נוכל שלא לשים לב אליו. פעם אחת נכנס input אחד ופעם אחת נכנס אחרת – וoutput היה שונה.
הזכרנו את רשת הנוירונים שבהמשך כשנלמד על בינה מלאכותית נרצה 'לייצר' – זה בעצם ההקשרים בין דברים שונים שיכולים להיות לוגים אבל לא בהכרח, חלק מהידיעות שלנו אנחנו עושים בלי שנדע – כי למדנו מהניסיון, עשינו משהו וטעינו/ הצלחנו – ועכשיו אנחנו יודעים טוב יותר מה לעשות – כלומר נוצר לנו קשר בין 2 נוירונים שלא היה קיים לפני.
תהליך של למידה זה כשנוצרים עוד קשרים בין נוירונים, או שמתבצע עדכון של הקשרים הללו.
במוח למעשה מתבצע תהליך של כיוונון של הקשרים בין הנוירונים, אני עושה משהו אחד אבל מגלה שעומד להגיע משהו אחר אז אני משנה טיפה משהו אחר ועוד דבר ששמתי לב שמשפיע עד שאני מגיע לתוצאה הרצויה - לפעולה הזו אנו קוראים למידה.

שיעור 5

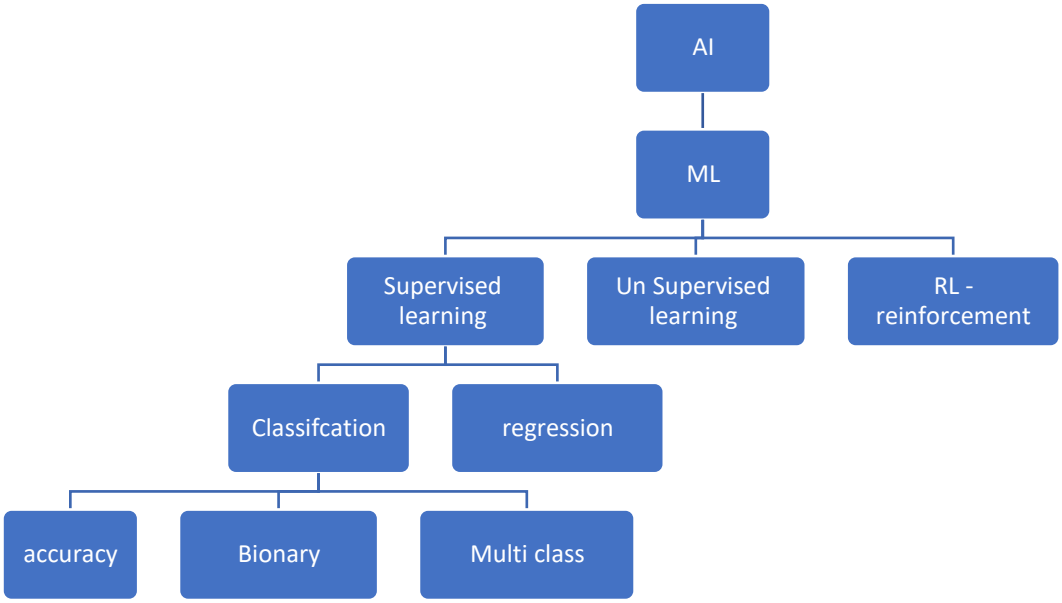
אינטליגנציה מלאכותית OUT, למידת מכונה IN

1. אלן טיורינג טען שאם אנו רוצים לדעת מתי ניתן להגיד שמכונה אינטליגנטית? – כאשר נקיים צ'ט עם מכונה ועם אדם ולא נדע להבדיל ביניהם.

כנגד הדבר הזה ישנן 2 אסקולות – הראשון הוא מפיטר נורווינג שטוען שמחשב הוא מחשב ויכול להיות שיש לו אינטליגנציה אבל לא באותו אופן שאנו מסבירים אינטליגנציה. כמו שכשהאדם רצה לעוף הוא ייצר מטוס, הוא לא ייצור מכשיר שיהיה הכי דומה לציפור, אלא כשהוא 'מתרגם' את היכולת הרצויה למושגים שלו הוא יוצר מטוס.
האסכולה השנייה נקראת החדר הסיני – מאיפה מגיעה המחשבה שלנו? מהמוח? מהלב? אין באמת מקום מוגדר, כי אם נפתח את המוח נראה תזוזות של נוירונים ולא מחשבה. כאן נכנס משל החדר הסיני – אדם יושב בחדר גדול ונכנסים אליו כרטיסיות שאומרות לו כל מיני דברים והוא בהתאם לפקודות שהוא מקבל הוא מחפש בספר לבדוק מה הוא צריך להחזיר כתגובה חזרה. הוא לא יודע שהוא קיבל מילה בסינית והחזיר סינית גם – בחוץ הבינו אותו אבל לא ניתן לומר שהוא מבין סינית. את ה'קרדיט' אנחנו לא ניתן לאדם הזה אלא למי שכתב את המילון. לכן אם מחשב יודע להגיב לכל פקודה שניתן לו – אנחנו לא יכולים לומר שיש למחשב אינטליגנציה, החוכמה לא נעצרת אצלו.

מסקנה – אנו לא נמהר לטעון ולהשתמש במונח אינטליגנציה מלאכותית (AI), אנו כן נדבר על למידת מכונה (ML) .

תחומי האינטליגנציה המלאכותית



2. לא נעמיק בכל תחום אך ננסה להבין מה אופי המודל שנבנה בעבור כל שיטת אינטליגנציה מלאכותית:

3. **Supervised learning**-מודל שבו המשתמש/המתכנת מכניס מראש למחשב נתונים ולמעשה מעין 'מאמן' אותו בקלט נתון ומה אמור להיות הפלט. למשל מודל שבו נכניס למחשב מאות תמונות של חתולים וכלבים ונרצה שבהינתן תמונה חדשה שלא ראה – יידע לקבוע האם זה חתול או כלב. השלב הראשון ייקרא שלב האימון (training) והשלב השני ייקרא שלב הבדיקה (testing). היתרון של השיטה הזו היא שניתן להגיע לרמת דיוק מאוד גבוהה אבל יש לה חיסרון מסוים - מתי נוכל לקבוע שלמודל יש אינטליגנציה? אילו למערכת יש את היכולת לתקן את עצמה באופן עצמאי וללמוד מהניסיון. ובאמת עם התקדמות הטכנולוגיה מחשב מגיע למצב בו יכולת הזיהוי שלו הוא אפילו יותר גבוה משל בני אדם. אבל הטעויות שהמחשב <u>כן עושה</u> – בני אדם בחיים לא היו עושים. למשל זיהוי של קוף בתור אדם.

4. **Un supervised learning** – הניסיון ללמוד בלי שאף אחד הגדיר לו מראש את ההגדרות. למשל נטפליקס ראה שאהבת סרט מסוים ואז הוא רוצה לתת לך הצעות לדברים שתאהב. אין חוקיות לגבי מה אתה תאהב אם ראית סדרה מסוימת אבל בכל זאת הרציונל הוא שדברים דומים נמצאים אחד ליד השני (קלסטרינג) כלומר שאם משתמשים אחרים נכנסו לסרט הזה וגם לסרט אחר – הגיוני שגם אתה תרצה את שניהם. הבעיה בתחום זה שהאלגוריתמים בו עובדים פחות טוב.

5. **Reinforcement learning** – סוג של supervised learning שבו אני לא מקבל את הפידבק האם נכון או לא נכון על המקום אלא רק בדיעבד, ואז אתה מבין בדיעבד מה הייתה הטעות ולא תחזור עליה אם תיפגש בה שוב. שיחקתי משחק מסוים והפסדתי, עשיתי בו המון החלטות, ככל הנראה שאחת או שתיים מהן הן אלו שגרמו לכך שאני אלך ואפסיד, יכול להיות שאעשה מהלך כזה ורק אחרי כמה תורות אני אבין שטעיתי – הדגש הוא שההבנה על הטעות מגיעה בדיעבד. כאשר אתה משחק מספיק פעמים או שאתה עושה פעולה מסוימת כל כך הרבה פעמים – אתה תדע לעלות ברגע שקורית טעות גורלית, זה לפתח מעין 'טביעת עין' שיודעת לזהות ולצפות כבר קדימה.

6. **Regression** - יש לנקודות נטייה לעשות רגרסיה לממוצע. אם למשל אבא מסוים מאוד גבוה, הבן שלו סביר להניח יהיה מעט מתחתיו, ואם אבא יהיה מאוד נמוך סביר להניח שהבן יהיה מעט גבוה ממנו. איינשטיין היה ממש חכם, ויכול להיות שגם הבן שלו היה חכם מאוד מעל הממוצע אבל הוא לא יוצא דופן כמו אבא שלו, הוא מעט לכיוון הממוצע. יש יוצאים מן הכלל אבל בגדול זה קורה. למערכת יש נטייה לעשות regression to the mean. רגרסיה למעשה זה לנחש ערך מספרי – והשיקול שלנו יהיה לכיוון הממוצע.
7. **Classification** – בשונה מרגרסיה עליה דיברנו למעלה – קלסטרינג זה למעשה היכולת <u>לסווג</u> דברים לפי נתונים שאינם ניתנים לכימות. אני לא יכול להגיד שיונקים גבוהים יותר מעופות, ולא ניתן לקחת מאפיין מסוים שיתאר חתך שיהיה ניתן לייחס את הנקודות לקו לינארי או דפו מסוים. בעיות מסוג אלו נקראות בעיות קלסיפיקציה כי נדרשת חלוקה לקלאסים, לסווג לקבוצות לפי סכימת נתונים שקרובים אחד לשני, אבל לא בהכרח זהים. קלסיפיקציה יוצאת מנקודת הנחה <u>שלדברים דומים יש נטייה להיות דומים או להיות קרובים אחד לשני</u>.
למשל – ניכנס לכיתה ונראה שאנשים התקבצו בחבורות, זה לא אומר שהתחביבים שלהם אותו דבר, שכולם גרים באותו אזור בארץ או כל נתון כזה או אחר המשותף להם – אבל ניתן לומר בגדול שהדבר שדומה ביניהם גרם להם להתקבץ אחד עם השני ולא עם מישהו מהקבוצה השנייה. דוגמא נוספת שאנו פוגשים הרבה – ראינו סרטון ביוטיוב, ואז קופצים לנו הצעות לסרטונים שאולי יעניינו אותנו – איך זה עובד? אנשים אחרים ראו את שני הסרטונים האלו וההיגיון אומר שאם אהבת את זה יש סיכוי שתאהב גם את זה. חשוב להבין – זה לא תפיסה מדויקת אבל יש בה מן האמת והיא עוזרת לסדר את המחשבות של מה אני מפצה לראות.
8. Accuracy - דיוק – איך אני מעריך את טיב המודל שלי? לצורך זה יש מטריקה – עד כמה המכונה צודקת. אבל גם פה צריך לשים לב כמה אני מה היחס שלי לטעויות, ואם יש טעות לאיזה כיוון – פה נכנסים 2 מושגים שעוד ניגע בהם בהמשך:
   א. False positive - שגיאה חיובית – למשל בדיקת הריון שאומרת שכן בהריון אבל לא.
   ב. False negative - שגיאה שלילית – למשל בדיקת הריון שאומרת שלא בהריון אבל כן.

   שיטה לזכור – אמרתי positive אבל זה false, או אמרתי negative אבל זה false)

בעיקרון אי אשר להגיד באופן גורף אילו מבין הטעויות הללו היא עדיפה כי זה משתנה בהתאם לצורך. לכן חשוב לשים לב בהתאם למה שאני מעוניין להציג לאיזה כיוון טעויות אני מוכן לקבל את האחוז המינימלי המסוים של חוסר דיוק ולאיזה לא. למשל מגנומטר בשדה תעופה – 'עדיף' לי שהמכונה תטעה כ False positive כלומר שתזהה אנשים שיש עליהם פצצה אבל באמת אין להם. מאשר תטעה כFalse negative כלומר אנשים שיש עליהם פצצה והמכונה לא התריעה לנו על זה.

בהקשר זה חשוב להוסיף דגש חשוב – אם תהיה לי מכונה מקולקלת שלא יודעת לומר כלום מלבד 'אין פצצה' – המכונה הזו תיתן תשובה נכונה ב99.99999% - כי באמת לרוב האנשים אין פצצה. אבל היא תטעה בבן אדם היחיד שיש לו. לכן לא להתרשם מאחוז הדיוק של מודל!

<u>data categorizing</u>

9. ספריית pandas נותנת לנו אפשרות להשתמש בנתונים של טבלה, כך שהטבלה מחולקת לדאטה ולפיצ'רים כל שורה בטבלה משמעותה נתונים של פרט יחיד (כמו שורה באקסל) וכל עמודה הינה פיצ'ר/פרמטר/נתון לגבי כל אחד מהפרטים. אנחנו נשאף להמיר פיצ'רים לערכים מספריים כדי

שתהיה לי את האפשרות לשאוב מהם את המידע. לתהליך הזה קוראים data categorizing, באופן הזה ניתן לומר שמשהו יותר קרוב למשהו – למשל כאשר יש בית עם 5 חדרים המחיר שלו יהיה קרוב יותר למחיר של בית עם 5 חדרים מאשר בית אחר עם 3 חדרים. אבל יש נתונים שלא ניתן להגיד שהם קרובים יותר למשהו אחר – למשל נגדיר אם צבע הבית ירוק נסמן 1, אדום 2, כחול 3. זה שנתתי לצבעים ערך מספרי זה רק כדי להבדיל ביניהם אבל בנושא של צבעים אי אפשר לומר שצבע 2 'קרוב' יותר ל1 מאשר 3. אם היה ניתן לראות שיותר קל למשל למכור בתים כחולים מאשר בתים אדומים וירוקים אז ניתן באמת לסווג את זה לפי רמת המשקל שהפיצ'ר משפיע.

<u>הציפייה שלנו לראות במודל</u>

10. למה אנו אמורים לצפות כאשר אנו בונים מודל לחיזוי? (הערה לפני שמתחילים - אם יש משהו שהוא יותר מידי טוב ומספק לי אחוזי דיוק כמעט מושלמים – יכול להיות שיש משהו שמסיט לי את התשובה ולכן לא להתלהב מוקדם מידי. ככל שנרצה מהמודל שלי תוצאות יותר טובות למשל 99% דיוק – אני צריך מאוד מאוד להתאמץ, הרבה מעבר למה שהיה צריך להתאמץ בשביל 90%). בנוסף, לפעמים אנחנו לא יכולים להוציא 100% דיוק כי אין לנו מספיק מידע.
11. Over fitting – מצב בו מערכת שיודעת לעבוד רק על דוגמאות שהיא כבר ראתה.

שלב למידת המכונה מתבצע ב2 שלבים – שלב האימון בו הוא מקבל נתונים כtraining set, לומד וזוכר אותן, ושלב הבדיקה בו אנו מאמתים שהמערכת מצליחה לפתור בעיות חדשות ובוחנים אותו על בסיס נתונים שלא ראה עדיין – testing set.

בשלב באימון – ככל שהמודל נעשה יותר מורכב עם יותר דוגמאות ויותר דברים שונים ומשונים, וגם מגדיר לו מה הוא רואה - כך רף השגיאה יורד. אבל בשלב הבדיקה – ככל שהמודל נעשה יותר מורכב רף השגיאה יורד, אבל בשלב מסוים הוא יעשה מעין פרבולה ורף השגיאה יעלה – כי המערכת לא יודעת להתמודד עם נתונים שלא מכירה ולא ראתה בתוך מודל מורכב ומסועף. מצב הover fitting זה מצב שבו המערכת כל כך 'מאומנת' על הdata set כך שהיא לא מסוגלת ללמוד דברים חדשים ולא יודעת לחשוב ולהסיק. אבל השאיפה שלנו תמיד תהיה שהיא תהיה מסוגלת לפענח בהתאם למה שהיא למדה דוגמאות חדשות, לכן חשוב לבנות מודל מורכב אבל לא מורכב מידי.

<u>שיעור 6</u>

1. 2 הערות – כל מי שמתעסק במדעי נתונים חייב להכיר את האתר Kaggle, זה אתר שמכיל המון דאטה סטים ובנוסף 'אתגרים' של אנשים שמעלים את הצורך שלהם ומציעים סכום כספי למי שיעשה את זה בשבילם. ניתן להשתמש באתר הזה כדי להוריד דאטה סטים גדולים ומסוגננים. האתר השני זה fast.ai שהוא מומלץ ללמידה על כל תחום הבינה המלאכותית.

<u>3 סוגים של אירוסים – הדאטה המפורסם בעולם</u>

2. הדאטה סט של האירוסים נוצר כאשר רונאלד פישר הלך ואסף 150 דוגמאות של אירוסים. הוא מדד אותם אחד אחד ואפיין אותם לפי רוחב ואורך של עלה הכותרת הראשי ועלה הכותרת המשני ל3 סוגים – סטוזה, ורסיקולור, ווירג'יניקה. במודל שלנו אני רוצה לקבל את הפרמטרים האלו ולסווג איזה פרח זה.

אנחנו מבינים שהאורך והרוחב בלבד הם לא אלו שמעידים לנו על הפרח כי זה משתנה.

נכתוב את הקוד:

ייבוא ספרייה המכילה את הדאטה סט של האירוסים הנ"ל From sklearn import load_iris( )

השמה במשתנה את הדאטה סט Iris = load_iris( )

הדפסת סוג המשתנה Type (iris)

הדפסת הדאטה עצמה Print (iris.data)

Print (iris.featurw_names)

סיווג המשתנים לפי תוויות Print (iris.target)

הדפסת כמות התוויות עם השם Print (iris.target_names)

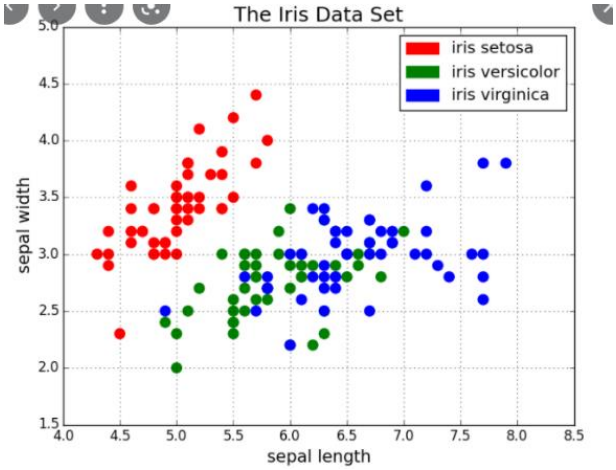
אלגוריתם - knn – k nearest neighbor

3. כלי שמשתמש ברגרסיה (ישנם 2 סוגי רגרסיה: לינארית - עבור כל ערך לקבל ערך מספרי – ואז אפשר להגיד מה קרוב יותר למשהו אחר. והשני זה לוג'יסטית- המבצעת קלסיפיקציה, כלומר אפיון, שאינו מספרי – למשל אדום או צהוב).

אנו נשאף לייצר מודל , אבל הגרף של מורכבות וכמות הנתונים הוא גרף אקספוננציאלי, זאת אומרת שאם יש לי מעט מידי נתונים וחוסר מורכבות ההיתכנות לשגיאה גדולה אבל באותה מידה גם אם יש יותר מידי נתונים ויותר מידי מורכבות אז שוב ההיתכנות לשגיאה היא גדולה, לכן צריך לזרוק חלק מהנתונים ולדעת להתמקד ברלוונטיים יותר.

ברגרסיה לינארית אני מייצר קו שהוא הכי קרוב לכל הנקודות ואז משווה רק על פיו, ובעצם מעין "זורק" את הנקודות לפח כי הקו הלינארי הוא המעיד על האופן שהגרף מתנהג, אבל יש בכך חיסרון- כאשר אני זורק את הנקודות לפח אני אמנם מייצר מודל פשוט הרבה יותר, (למשל – תגיד לי מה היה הגובה של אבא שלך ומה של אמא שלך ואגיד לך כמה יהיה שלך לפי הקו הממוצע הזה שיצרה לנו הרגרסיה הלינארית) אבל הצד הפחות טוב של מודל זה שאני מפסיד את האופציה שיתקבל בשלב המבחן תוצאה בדיוק כמו נקודה מסוימת שזרקנו אבל בגלל המודל הערך שהמודל מחזיר הוא אחר – כי הנקודה התכנסה לקו הגרף ולא למה שבאמת קרה במציאות. המודל כאילו 'לא מחויב' לנקודות שיצרו אותו.

לכן אולי נשמור את הנתונים הישנים? אבל מה יקרה אם יקבל ערך שלא נמצא בנתונים שהתאמן עליהם? הוא הולך לשכן הקרוב ביותר (nearest neighbor) ובודק מה הוא. זה המשמעות של knn – אנו מגדירים כמות (k) שכנים עליהם אנו רוצים לבדוק, ואומרים למודל לבדוק לפי k השכנים הקרובים ביותר – למשל מאיזה סוג יש הכי הרבה. זה נקרא לומד עצלן - lazy learner. משמע הוא לא עובד קשה בשלב הלמידה ואז יודע להתמודד עם מקרים שעוד לא למד, אלא הוא לא מייצר מודל וכשאומרים לו "מה הקלאס (הסוג) של הדבר הזה" הוא פשוט משווה איפה הוא ממוקם ומי הכי קרובים אליו, הוא פשוט אומר מה היחס בין הקלט לפלט, הוא אוגר ואוגר נתונים ולא עושה איתם כלום ורק כשמבקשים ממנו למצוא משהו הוא מחפש מה הכי קרוב.



כאשר אנו רוצים להציג לדברים דומים יש נטייה להתקבץ ביחד. אנחנו 'מקבלים תחושה' של מה סביר להניח שיהיה. עם זאת זה לא עובדות וחשוב לזכור את זה. אז באמת לשכן הכי קרוב זה לא בהכרח מעיד, אבל אם ניקח כמה שכנים (k) אני יכול לקבל רושם חזק יותר. כך שאם k מספיק גדול הוא מעין 'לומד' כי הוא מבין חוקיות או הסתברות מסוימת. יש פה מושג של אנומליה – מצבים בהם האלגוריתם אומר שזו נקודה חריגה והאזור הזה היה אמור להיות ירוק ולא כחול. ואז היא תישאר כחולה אבל מילימטר ימינה או שמאלה כבר נקבל ירוק בגלל ה-knn, והנקודה הזו לא הייתה אמורה להיות שייכת, גם אם במציאות יצא שכן. למה בכל זאת אני מתעלם ממנה? <u>כי לא באמת משנה לי מה קרה בעבר, אלא ללמוד מהעבר וע"י כך לנסות לחזות מה יקרה בעתיד כך שגם אם קרה בפועל משהו חריג אני יכול להתייחס אל זה כחריג.</u>

<u>שיעור 7</u>

<u>שמירה על התפלגות הנתונים</u>

1. חשוב להבין שכדי לסמוך על המודל שלנו אנו צריכים לצאת מנקודת הנחה <u>שההתפלגות של הדאטה בעתיד זהה להתפלגות הדאטה בשלב הלימוד</u>. אני יוצא מנקודת הנחה שמה שאני עומד לראות בtest דומה למה שראיתי כבר בשלב בtrain. חתולים שהיו לפני 1000 שנה לא נראים מאוד שונה מהאופן שהם נראים היום – לכן ניתן לסמוך על מודל שיתבסס על זה. אבל מודל שמשתמש בדאטה של כמות השימוש ואופן השימוש בטלפון הנייד בעידן לפני הסמארטפונים, וכיום – עצם הדאטה השתנתה. אבל כל עוד לא היה שינוי מהותי אני אצא מנקודת הנחה שאני לא אמור להיפגש עם סיטואציה חריגה שלא נחשפתי אליה מעולם או אפילו לא לדבר שדומה לה. כאשר אני עושה את זה ניתן לבצע את המודל העצלן/ קלסטרינג (סיווג)/ רגרסיה וכל השיטות שלמדנו לפני כן. כי לדברים מאותו סוג/דומים יש נטייה להיות דומים/קרובים אחד לשני. אם נחזור לדוגמה עם האירוסים – לרוב למשל לפרחים שהעלה הראשי והמשני שלו הם ביחס זהה אורך רוחב הם מהסוג הראשון בעוד שהסוג השני היחס הוא לא 1:1 אלא 1:2 למשל.

<u>רגרסיה לינארית</u>– ערך מספרי

2. רגרסיה לינארית זה כאשר יש לנו פרמטר מסוים ואני נותן לכל פרמטר ערך מספרי מסוים כך שניתן לסווג מה קרוב יותר למה מבחינת הערך. <u>הרגרסיה הלינארית תיתן לנו מספר ממשי</u>. למשל בהינתן כל הפרמטרים של בית מסוים – כמה הוא יעלה?
3. המשוואה של רגרסיה לינארית היא:

$$y=\sum wi*xi+b$$

(wi= כמות/ כמה משקל אני נותן לפרמטר הספציפי הזה

xi הפרמטר, למשל מ"ר, חדרים, מס' חדרי שירותים וכו'.

b= קבוע מסוים שאינו מתחשב בפרמטר)

איך קבלן יודע להגיד כמה יעלה לבנות בית מסוים? מטר רבוע (פרמטר מסוים) כפול כמה בממוצע עולה לבנות מטר רבוע. למה יש לנו את הb? למשל תעריף של נהג מונית – הוא מחשב את המרחק שאתה מתכוון לנסוע, כפול כמה יעלה כל ק"מ, אבל לזה תמיד מוסיפים תעריף קבוע מסוים.

<u>רגרסיה לוג'יסטית</u> <u>(מלשון המילה log) – סיווג הלינארית</u>

4. מהרגרסיה הלינארית אני מקבל מספר ממשי, כך ניתן להגיד שמשהו אחד גדול יותר ממשהו אחר, למשל מחיר של בית אחד מול מחיר של משהו אחר. אבל לא תמיד יש לי מה לעשות עם מספר ממשי, לפעמים אני צריך פירוש מילולי לרגרסיה הלינארית למשל הכנסתי את כל הפרמטרים של פרח מסוים ויצא לי 500. מה זה אומר לי?

בעיות מסוג זה אלו בעיות קלסיפיקציה (סיווג) ולצורך זה יש את הרגרסיה הלוג'יסטית. שבה אני בעצם מגדיר את טווחי הסף של כל עניין ועושה קלסיפיקציה למשל שרגרסיה לינארית בין 1 ל200 זה סוג 'סטוזה', 200 ל350 זה 'וירג'יניקה', ובין 350 ל500 זה 'ורסיקולור'.

5. איך עושים את זה?

```python
from sklearn.linear_model import LogisticRegression

# instantiate the model (using the default parameters)
logreg = LogisticRegression(solver='liblinear',max_iter=100)
#logreg = LogisticRegression(max_iter=200)

#liblinear
# fit the model with data
logreg.fit(X, y)

# predict the response values for the observations in X
logreg.predict(X)
```

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       2, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 2, 2,
```

<u>מעבר קצר על שימוש פונקציות בsklearn</u>

6. fit – מציאת נכונות המודל הכי טוב (חשוב לשים לב על מה מופעל הfit. אם יופעל על knn הוא לא יעשה כלום כי הוא מודל עצלן. אם יופעל על logistic regression הוא דווקא יעבוד קשה וכו').
הפונקציה תקבל 2 ערכים – הראשון x זה הדאטה שעליהם אני רוצה להפעיל את המודל ולבדוק לגביהם, והשני y זה הדאטה <u>הנכונה</u>. ואז לאחר שביצע את המודל x הוא משווה מול התשובות ובודק כמה צדק למשל – יש לי כרטיסיות עם מילים באנגלית ומאחור הפירוש באנגלית, זה מיוצג ע"י הy. עכשיו אני לוקח מחברת רק עם המילים באנגלית וכותב לכל אחת מהן את מה שאני יודע. פונקציית הfit עוזרת לי כי אני רוצה לדעת באיזה פרמטרים להתחשב או לא כדי לקבל את התוצאה הכי טובה. למשל הוא שם לב שכדי לקבל את הדיוק הכי טוב אין מה להתמקד ביחס העלים של הפרח אלא באורך הגבעול, ושטח החלק הלבן ביחס לסגול למשל.

predict - בהינתן ערך <u>חדש</u> שלא למדתי לפני כן – מציאת הסוג הכי קרוב אליו (יחזיר ב0, 1, 2 וכו' לפי כמות הסוגים שלימדתי מראש – למשל סוגי הפרחים). לי אין עניין רק לזכור עוד ועוד מידע למשל התנהגות לקוחות בחברה מסוימת אלא לנסות לחזות מה תהיה התנהגות הלקוחות בעתיד

score – בנוי על פונקציית הpredict ובו יש 'פירוש' של הערך הספרתי לסוג הספציפי/ יסווג מתוך הדאטה החדש כמה שייך לכל סוג. זה למעשה פונקציה שיודעת כבר את התשובה מראש אבל הוא כאילו בוחן בכל זאת האם המודל צודק או לא ויחזיר מה אחוז התשובות הנכונות שלי (מספר עשרוני בין 0 ל1).

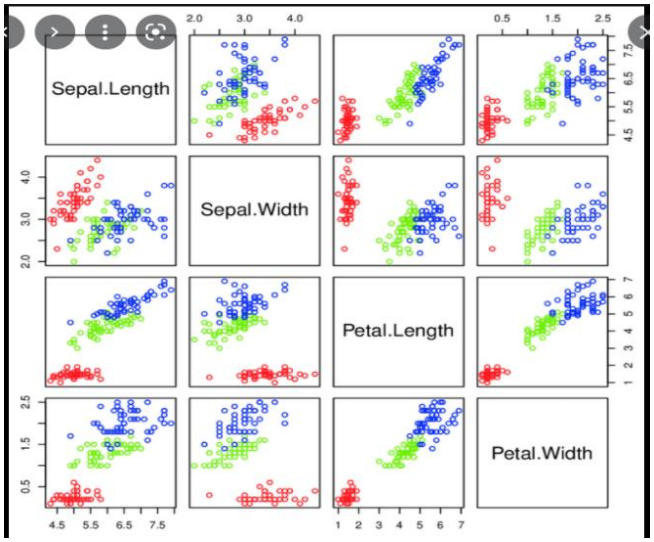
metrics.confusion_matrix – יצירת טבלה של כמה מכל סוג חשבתי שזה כל סוג, ובו האלכסון יהיה התשובות הנכונות.

[50 0, 0] במצב זה הוא צדק בכל סוג 0 – משמע כל פרח מסוג 0 שקיבל הוא צדק לגביו.

[0, 45, 5] במצב זה יש 5 פרחים מסוג 1 שסיווג כ2

[0, 1, 49] במצב זה יש פרח אחד מסוג 2 שסיווג כ1

knn.predict_proba – (קיצור של predict probability – חיזוי הסתברות/ היתכנות - יובן יותר טוב לאחר שיעור 10+ 11) הפונקציה תקבל את הערכים של הפרט (למשל 2,3,4,5 – אורך רוחב עלה משני ראשי) הוא ידפיס לי למשל: [0.375, 0.625, 0]
הפונקציה הזו למעשה מאפשרת לי לא רק לדעת איזה סוג זה יהיה כמו שעושה predict, אלא יסביר לי כמה הוא בטוח בתשובה הזו. למשל יקבל ערכים של פרח מסוים ויגיד לי בהתאם למודל מה הסבירות שלו לתת תשובה נכונה. כמו שראינו בדוגמא קודמת הוא צדק בכל הפרחים מסוג 0 שקיבל לכן פונקציה זו תחזיר 0 כי הוא לחלוטין בטוח שזו אינה התשובה. ואז לי הערכים שהכנסתי לו הוא אומר לי עד כמה סביר להניח שזו התשובה בהתאם לכמה הוא טועה בסוג/בטווח הזה.



sns.pairplot – בספרייה אחרת - seaborn ישנה פונקציה שניעזר בה בניסיון מציאת קשר בין 2 משתנים ע"י תצוגה של מטריצה (למשל אורך ורוחב של עלה כותרת משני וראשי ייתן לי טבלה של 4 על 4 טבלאות קטנות כך שאני אוכל לזהות אולי חוקיות מסוימת – שישנו פיזור או יותר נכון 'דפוס' של השונות באופן שישנה חוקיות מסוימת למשל כאשר מייחסים משקל רב יותר לרוחב ואורך העלה המשני. מה שנקבל בדוגמה זו הוא את הטבלה הבאה, נשים לב כי 2 המשבצות ליד המשבצת התחתונה ימנית הם למעשה בשימת הדגש על ייצוג של עלה כותרת משני גדול – ונראה שישנה התפלגות לינארית שממש מחלקת לי את הסוגים באופן כזה שככל שהעלה המשני גדול יותר גם האורך שלו גדול יותר – וזה ההבדל בין הסוגים. אז כשאני ארצה לבחון פרח חדש אני אסתכל קודם כל על גודל העלה המשני. הוויזואליזציה עוזרת לי להבין משהו חדש ומעניין על הגרף כך שניתן לשפר את המודל.
נחזור שוב לנקודה שאמרנו כבר – פה הצלחתי לזהות את הדפוס הזה כי השתמשתי ב4 פרמטרים

– אורך ורוחב של עלה משני וראשי. אם הייתי משתמש בעוד הרבה פרמטרים ההתפלגות שלי הייתה כל כך מפורטת שלא הייתי יכול להסיק שום דבר. לכן לא תמיד עוד מידע והוספת ממד/ מורכבות עוזרת לי, לפעמים אפילו ההיפך.

שיעור 8

חזרה על שיעור קודם

1. Confusion matrix – נקדים ונזכיר שSupervised learning זה כשיש לי את הנתונים ויש לי יכולת לדעת מה אמת ומה שקר ואז להפעיל בדיקה עד כמה המודל שלי טוב. כמו שהסברנו confusion matrix היא כלי לבדיקה זו ע"י יצירת טבלה של כמה מכל סוג חשבתי שזה כל סוג, ובו האלכסון יהיה התשובות הנכונות.

[50 0, 0] במצב זה הוא צדק בכל סוג 0 – משמע כל פרח מסוג 0 שקיבל הוא צדק לגביו.

[0, 45, 5] במצב זה יש 5 פרחים מסוג 1 שסיווג כ2

[0, 1, 49] במצב זה יש פרח אחד מסוג 2 שסיווג כ1

מכיוון שרוב הדאטה בעולם הוא ללא לייבלים (כלומר ללא הבחנה של אמת ושקר או הגדרות שניתן ללמד על ידם את המודל שלנו) לכן יש לנו את כל סוגי הלימודים האחרים של המודל, שבסופם אני חייב להסביר למה המודל נתן לי תוצאה כזו או אחרות.

Psauodo Random

2. רנדומליות שקרית. כדי לבדוק דאטה מסוים יש חשיבות מאוד גדולה לערבב את הדאטה גם בשלב הלימוד וגם בשלב המבחן. כאשר אנו מאמנים את המודל שלנו ישנה פונקציה שמערבבת את הדאטה בצורה רנדומלית (train_test_split(x, y, random_state=1)) כדי שהמודל ילמד שוב באופן אחר כדי לחדד את ההבנה – שלא נקבל מצב שבו למשל כל הראשונים הם מסוג אחד וכל האחרונים מסוג אחר. לכן מפעילים את הערבוב הזה, אלא שזו פעולת ערבוב מזויפת, כי בכל אינדקס ערבוב שאכניס הוא אמנם יערבב בצורה שונה אבל עבור אותו אינדקס הוא ישחזר לי את אותו סדר בדיוק שהיה. לכן זה שקר.

למה זה חשוב? כי יכול להיות שיש דאטה סט מסוים שטוב יותר מאחר, למשל שהוא מכיל ערכים מובחנים יותר (לדוגמה -אני לומד מתמטיקה, ברור שיש הבדל ברמת ההבנה שלי אם הנתונים שאני לומד זה: א. 8*0=0, 5*1=5, 4+0=4, 2+0=0 או ב. 11*11=121, 60/5=12, 7*6=42, 4*5=20.

לכן אני רוצה לאמן את המודל שלי בסידור שונה של הנתונים שהוא מקבל, או שיאמן את המודל ע"י שילמד את הנתונים של הבדיקה ויבחן האם צודק בנתונים של האימון! אני לא רוצה להגיע למצב שבו הדיוק שלי תלוי באיזה מודל השתמשתי.

Train-test.split

3. ראינו כבר לפני כן על המודל שלומד את הדאטה סט ואז בוחן את עצמו עליו – זהו סוג מודל מסוג supervised learning שבו הוא יודע כבר את התשובה ולכן יש לו אפשרות לבחון את עצמו ולומר בכמה צדק ובכמה טעה.

איך נכתוב אותו?

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=1)
(# default split is 75% for training and 25% for testing)

print(X_train.shape)
print(y_train.shape)
print(X_test.shape)
print(y_test.shape)
```

מה שנקבל זה:

קבלת דאטה סט של 150 על 3 – (150, 3)
בוחן את עצמו על 150 – (150,)
קבלת דאטה סט של 50 על 3 – (50, 3)
בוחן את עצמו על 50 – (50,)

רגרסיה לינארית - המשך

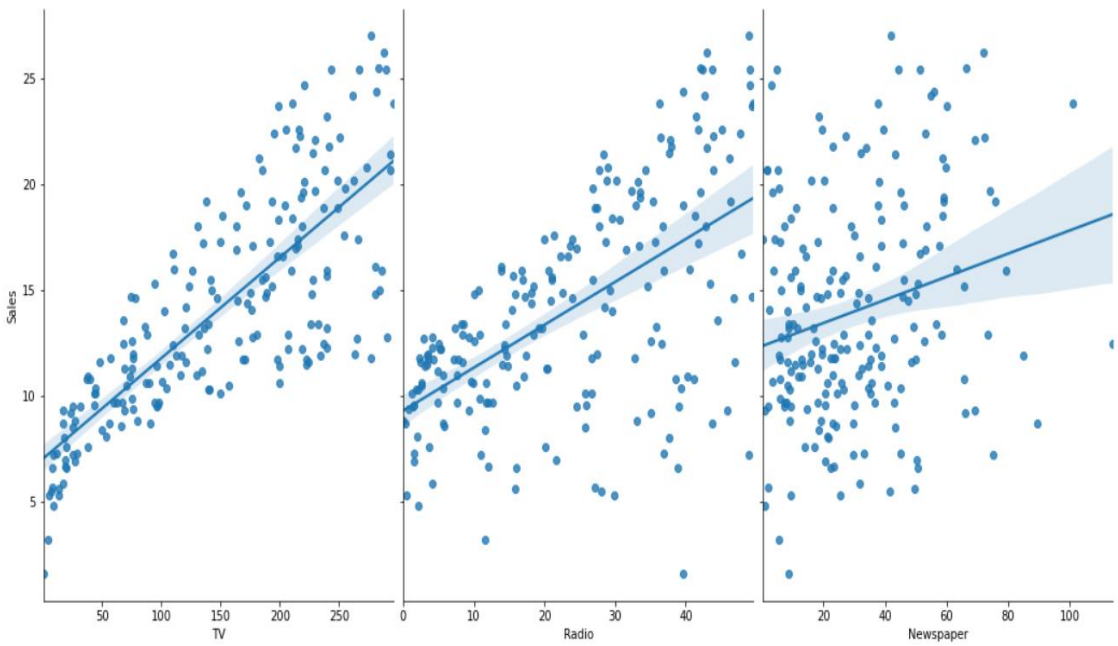
4. אחרי שנמשיך את את הקוד מסעיף קודם – אימנו את המערכת ואנחנו רוצים להציג את זה בתור רגרסיה לינארית:

```
from sklearn.linear_model import LinearRegression
linreg = LinearRegression()
linreg.fit(X_train, y_train) (# fit the model to the training data (learn the coefficients))
```

5. נתבונן במודל הזה:

```
# visualize the relationship between the features and the response using scatterplots
sns.pairplot(data, x_vars=['TV','Radio','Newspaper'], y_vars='Sales', height=7, aspect=0.7, kind='reg')
```

```
<seaborn.axisgrid.PairGrid at 0x7fb0288ec550>
```



כמו שאנו רואה במודל הזה ניתן לראות שהגרף מלמד אותי משהו – זהו גרף של כמות האזכורים של חברה מסוימת באמצעי התקשורת השונים – טלוויזיה, רדיו ועיתון. ואנו רואים שכאשר ישנו פרסום בטלוויזיה ישנו גרף לינארי יחסית ברור בין כמות הפעמים שהופיע בטלוויזיה לבין כמות המכירות שלה, אז ככל הנראה אשקיע יותר

6. אחד הדברים המייחדים רגרסיה לינארית זה שהכי קל לראות למה נתון שקיבלתי הוא נכון. נזכיר את המשוואה של רגרסיה לינארית: $y=\sum wi*xi+b$

(wi= כמה משקל חשיבות אני נותן לפרמטר הספציפי הזה
xi הפרמטר, למשל מ"ר, חדרים, מס' חדרי שירותים וכו'.
b= קבוע מסוים שאינו מתחשב בפרמטר)

למעשה ע"י הW ניתן לראות שנתינת דגש מסוים על עניין מסוים משפיעה על הגרף שנקבל כך שהדברים יהיו הרבה יותר מובחנים וניתנים לחלוקה, והדגש החשוב – לניבוי וחיזוי (!) מה יכול להיות התוצאה עבור ערך חדש שעוד לא למדתי.

עם זאת לא תמיד ניתן להתבסס על W כאשר יש מטריקה שונה לכל פרמטר. לכן חשוב מאוד לא להתבלבל! במקרים אלו W גדול יותר לא אומר משהו חשוב יותר! אלא שאני נדרש לעשות נורמליזציה של המודל כי למשל אם נמשיך בדוגמה של מציאת מחיר דירה – אם יש בדירה מסוימת חדר שירותים אחד או שלושה – זה משמעותי, לכן זה ישפיע על המחיר למרות שמבחינת כמות יש שינוי רק של 2, בעוד שאין הבדל אם המ"ר של הדירה הוא 102 או 104. לכן כדי לנרמל את המשקל שאני נותן לכל אחד אני צריך להתאים W מתאים לכל X כדי שההשפעה בכמויות תיתן output שרלוונטי למודל – לתהליך זה אנו קוראים scaling.

נשים לב למשל עמ"נ להציג את הW נרשום את המשתנה של linerg עם הפונ' coef.

Linear regression in scikit-learn

```
# import model
from sklearn.linear_model import LinearRegression

# instantiate
linreg = LinearRegression()

# fit the model to the training data (learn the coefficients)
linreg.fit(X_train, y_train)
```

```
LinearRegression()
```

Interpreting model coefficients

```
# print the intercept and coefficients
print(linreg.intercept_)
print(linreg.coef_)
```

```
2.8769666223179318
[0.04656457 0.17915812 0.00345046]
```

```
# pair the feature names with the coefficients
list(zip(feature_cols, linreg.coef_))
```

```
[('TV', 0.04656456787415029),
 ('Radio', 0.17915812245088839),
 ('Newspaper', 0.003450464711180378)]
```

$$y = 2.88 + 0.0466 \times TV + 0.179 \times Radio + 0.00345 \times Newspaper$$

שיעור 9

העשרה שלא קשורה לקורס – איך ללמוד במבחנים

סיכום - או מה ניתן לעשות בפועל

בעבודה על הפרק הנוכחי ביקשתי מאחד החברים המבריקים ביותר שלי שיחלוק אתנו את הדרך בה הוא לומד לבחינות. מדובר על אדם שממוצע הציונים שלו בתואר הראשון במסלול המשולב הנדסת אלקטרוניקה - פיזיקה הוא 98 וכרגע הוא דוקטורנט לפיזיקה בטכניון. אני לא מבטיח שאם תעקבו אחרי המתכון שלו תגיעו לממוצע 98, אבל למרות זאת שווה להציץ.

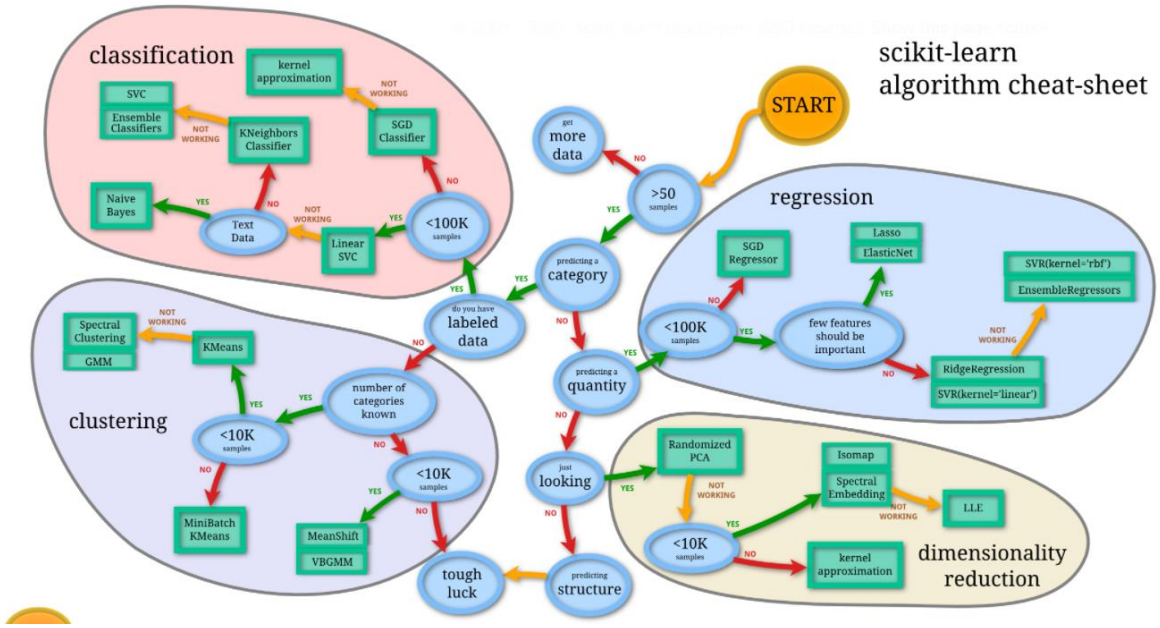
השלב הראשון הוא לעבור על ההרצאות בקריאה מהירה, לראות שאנחנו מבינים את החומר התיאורטי ומבינים את התרגילים הפתורים של המרצה והמתרגל. אם בשלב הזה החומר נראה לכם כמו סינית, חזרו שוב ושוב על ההרצאות וספרי הלימוד.

השלב השני הוא לעבור על הסיכומים שסיכמנו במהלך הסמסטר ולסכם אותם עוד יותר. קשה להסביר את החשיבות של השלב הזה. לעבור ולסכם בפעם השנייה את הסיכומים ממהלך הסמסטר עוזר להחדיר אותם בצורה אפקטיבית למוח שלנו.

השלב השלישי הוא הכנה של דף נוסחאות, כמו שכבר הסברנו. מי שיכול לתמצת את החומר לדף נוסחאות בודד מעיד על עצמו שהוא מבין את החומר.

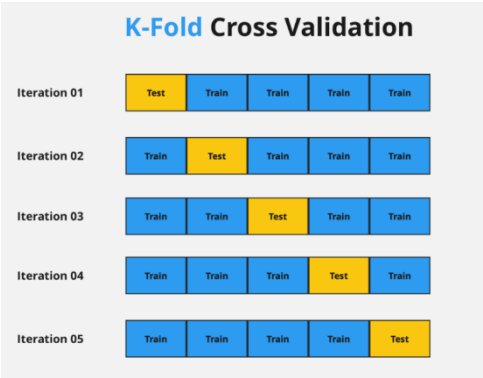
1.

2. נציג גרף מהאתר של scikit learn המתאר באיזה סוג מודל מומלץ להשתמש לפי כמות ואופי הדאטה שיש לי:



אלגוריתם k-fold – cross validation

3. השאלה 'כמה הצלחתי' במודל היא שאלה קריטית, ואנו מבינים שהיא ניתנת להגדרה רק כאשר יש לי מטריקה – כלומר יכולת למדוד, והדבר הזה רלוונטי רק בsupervised learning, לעומת זאת במודל הclustering אין מטריקה – למשל תמצא לי כתבה באינטרנט שדומה לכתבה הזו – אין דרך להבחין לגבי כל כתבה האם באמת יש דמיון או לא.
לכן בשביל לשאול כמה הצלחתי – למעשה מה הדיוק של המודל שלי – אני נדרש בעוד דאטה, מה הבעיה? שלפעמים דאטה זה עניין מאוד יקר.
4. מודל הk-fold כך שk הוא מספר שאני בוחר, (כמו knn) שלמעשה מחלק את כל הדאטה לk חלקים כך שכל פעם הוא מתייחס לאותו לחלק הספציפי בתור הtest, ובכל יתר הדאטה כtrain.



למשל עבור 5-fold ילמד את כל ה80% אחוז הראשונים, ויבחן את עצמו על ה20% האחרונים, לאחר מכן ילמד על ה60% הראשונים וה20% האחרונים ויבחן את עצמו על ה20% הנותרים וכן הלאה. כך הוא כביכול מתאמן על עוד דאטה אבל לא באמת הבאנו לו יותר דאטה.
חשוב להבין – אסור להשתמש במודל הזה בבנייה של המודל שאני בונה, כי למעשה הראית למודל כבר את הtest תוך כדי הtrain. ואז בהכרח נקבל אחוזי דיוק הרבה יותר גבוהים – אבל לא פלא. לכן נשתמש בk-fold רק כבדיקה לאשר שהמודל שלנו אכן עובד. זה כמו לראות את המבחן שיהיה בסוף שנה כבר בתחילת השנה. ואז סביר להניח שאמנם בחלקים שילמדו מהמבחן אתה תהיה מרוכז, ובמבחן תקבל 100, אבל אם תקבל מבחן אחר – על חומר שלא היית קשוב אליו (כי הוא לא למבחן) סביר להניח שתיכשל.
5. איך עושים את זה? למשל עבור דאטה סט של 25 שורות אני מחלק ל5 ואומר לו ללמוד 20 כל פעם.

```
From sklearn.model_selection import Kfold
Kf = kfold(n_splits=5, shuffle=True).split(range(20))
```

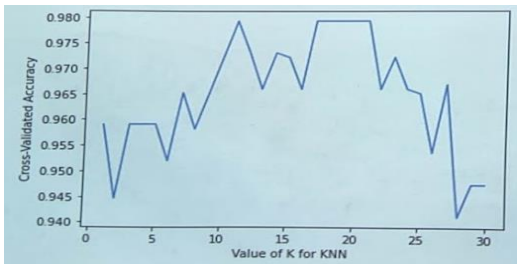
ההבדל בין בtrain/test split לבין cross validation

6. ניזכר במושג train/test split – הפעולה הזו למעשה לוקחת את כל הדאטה שאני כבר יודע, ומחלק אותו שאת הרוב ילמד, וכאילו 'ישכח' את החלק השני וכאילו יבחן את עצמו על החלק השני בהתאם למה שלמד בחלק הראשון – אבל יודע להגיד כמה צדק כי יודע גם את החלק השני. מה ההבדל אז ביניהם הוא כמה פעמים הוא עושה את החלוקה הזו ללמידה ובחינה.
train/ test split –מריץ פי k יותר מהר מאשר k-fold. ובנוסף הוא פשוט יותר בבחינת התוצאות של שלב הtest.
k-fold – נותן דיוק משוער גבוה יותר ובנוסף הוא משתמש בדאטה בצורה הרבה יותר יעילה – כי משתמש בכל פריט מידע גם בשביל הtrain וגם בשביל הtest.

ייעול הknn על ידי k-fold

7. הגודל של הדאטה של האירוסים הוא 150, שזה מידע, אבל לא באמת כמות מספיק גדולה כך שהחיזוי שלי יהיה אמין, אני צריך עוד מידע. השיטה הזו היא דרך "לייצר" עוד מידע, אבל זה לא באמת מידע חדש. לכן מה שהקוד עושה זה לבחור כל פעם k שכנים קרובים ביותר שונה ואז על

זה הוא מבצע את הk-fold ומחלק את הדאטה ל10 חלקים של 15 כל אחד, ואז מתאמן כל פעם על 135 אחרים ובוחן על 15, ובכל בחינה ובחינה שכזו הוא בודק מה אחוז הדיוק שנקבל.
למה זה עוזר לי? – כי ניתן להשתמש במודל הknn שייתן לי דיוק מסוים, אבל הפעולה הזו אפילו משדרגת את התוצאה שלי כי הוא בוחן <u>כמה שכנים עדיף לי לקחת כך שהמודל יהיה יותר מדויק</u>, כלומר מה הk האופטימלי.



למשל עבור הקוד שכתבנו למעלה נקבל את הגרף המצורף הנ"ל ונראה שב k=12/17/18/19/20/21 אנחנו מקבלים את האחוזים הגבוהים ביותר. אז נבחר ככל הנראה את 19, כי אלו שקרובים אליו גם לא 'משבשים' את הדיוק כך שאני הולך על יותר 'בטוח'.

8. חשוב להבין שיש עוד המון דברים במודל שניתן לשחק איתם ולעשות עליהם k-fold ולא רק על הknn. כך ש'מיטיבי לכת' ינסו להשתמש בפרמטרים האחרים שיש במודל ועליהם לעשות בדיקה כל מה מהם הכי 'משתלם' לעשות k-fold.

<u>אידיאליזציה של המודל – לטוב ולרע</u>

9. לא נרחיב אבל כלי לזה הוא למשל הscore וneg_mean_score, ניתן ללמוד על זה מי שרוצה.
10. ברגרסיה לינארית התרגלנו להסתכל על קו לינארי פשוט בלבד – במעלה ראשונה, אבל האמת היא שלא חייב להיות מקובעים לכך ואולי אם לא נחייב את הדאטה להתכנס לרגרסיה לינארית במעלה ראשונה בזמן שאולי היה עדיף לה במעלה שניה ואז היינו מקבלים תוצאות אלגנטיות פתאום שמעידות יותר על התנהגות הגרף.
11. נסביר את זה – אם נשאל מהו המספר האידיאלי של תלמידים בכיתה? השאלה הזו יוצאת מנקודת הנחה של רגרסיה לינארית ,זאת אומרת שככל שיש יותר או פחות – ההשפעה עולה או יורדת, למשל ככל שיש יותר תלמידים זה פחות טוב. יש בזה אולי מן האמת אבל זו הסתכלות קצת עיוורת וחד ממדית – יש עוד המון פרמטרים שצריך להסתכל עליהם, כי החיים הרבה יותר מורכבים מזה. לכן אם עשינו רגרסיה לינארית וקיבלנו תוצאות שלא מספקות לנו הבנה מסוימת או שמראש אנו מבינים שאמורה להיות מורכבות גדולה יותר – מומלץ להוסיף עוד פרמטרים או לחשוב האם יש מודל אחר יותר טוב.

<u>שיעור 10</u>

<u>Grid search</u>

1. לפני שנסביר מה אלגוריתם/פונקציית grid search עושה - ניזכר במה שראינו שיעור קודם. ראינו איך מבצעים את הk-fold (או cross validation) על הknn למציאת מספר השכנים האידיאלי כדי "לנחש" טוב יותר מה הסוג של פרח נתון מתוך על הדאטה סט של האירוסים.
למשל במקרה הנ"ל לקחנו 5 שכנים ועליהם עשינו קרוס ולידציה של חלוקה ל10, ואז ידפיס את אחוז הדיוק. במקרה זה המודל ירוץ 10 פעמים – כל פעם על חיתוך אחר מבין ה10.

```python
from sklearn.model_selection import cross_val_score
```

```python
# 10-fold cross-validation with K=5 for KNN (the n_neighbors parameter)
knn = KNeighborsClassifier(n_neighbors=5)
scores = cross_val_score(knn, X, y, cv=10, scoring='accuracy')
print(scores)
```

```
[1.         0.93333333 1.         1.         0.86666667 0.93333333
 0.93333333 1.         1.         1.        ]
```

```python
# use average accuracy as an estimate of out-of-sample accuracy
print(scores.mean())
```

```
0.9666666666666668
```

לעומת זאת אנחנו רוצים לבדוק אולי יש פרמטר אחר שניתן לקחת – למשל כמות שכנים אחרת, ולא 5 שאנחנו בחרנו, ואולי במצב זה אחוז הדיוק שלנו ישתפר. לכן נעשה לולאת for שרצה על k מ1 ל30 ולזה יעשה קרוס ולידציה של 10. במצב זה האלגוריתם ירוץ 300 פעם.

```python
# search for an optimal value of K for KNN
k_range = list(range(1, 31))
k_scores = []
for k in k_range:
    knn = KNeighborsClassifier(n_neighbors=k)
    scores = cross_val_score(knn, X, y, cv=10, scoring='accuracy')
    k_scores.append(scores.mean())
print(k_scores)
```

```
[0.96, 0.9533333333333334, 0.9666666666666666, 0.9666666666666666, 0.9666666666666668, 0.9666666666666668, 0.966666666666666
8, 0.9666666666666668, 0.9733333333333334, 0.9666666666666668, 0.9666666666666668, 0.9733333333333334, 0.9800000000000001,
0.9733333333333334, 0.9733333333333334, 0.9733333333333334, 0.9733333333333334, 0.9800000000000001, 0.9733333333333334, 0.98
```

עכשיו נוכל לגשת להבין מה זה grid search – אנו מבינים שכל פרמטר שאני רוצה לבדוק אני צריך בשבילו לולאת for, למשל אם נרצה לא רק לבדוק מה הk האידיאלי אלא גם באיזה סוג מטריקה לבחור – כלומר האם להסתמך בכמות השכנים הקרובים על אלו שהכי קרובים מבחינת מרחק, או מתוך הk ההכי קרובים – מה יש הכי הרבה, יכול להיות שמשהו מזה ישפר לי את אחוז הדיוק. ואז לא נעשה 300 איטרציות אלא 600. ואם יהיו הרבה פרמטרים? הקוד עלול להיות מאוד מאוד איטי / מאוד מסובך/ עלולה להיות טעות קטנה שתשבש לי הכל.

לכך נועד הgrid search – למעשה נועד למנוע את הfor בתוך for בתוך for, ואל תוך הפונקציה עצמה אני מכניס מהם הפרמטרים עליהם אני רוצה לרוץ והוא "מרכיב" את כל הקומבינציות האפשריות. (האופן שבו הוא עובד זה שהוא מריץ במקביל כל אחד מהפרמטרים ואז מניח אותם אחד על השני)

לכל קומבינציית פרמטרים כזו (היפר-פרמטר) נבקש שידפיס לנו את הscore, כלומר מה אחוז הדיוק, ואם אני רוצה לדעת מה הדיוק האידיאלי אני לא אעשה max אלא argmax ולמשל יחזיר לי את האינדקס של הk המקסימלי.

2. איך מיישמים את זה בקוד?

```
from sklearn.model_selection import GridSearchCV
```

```
# define the parameter values that should be searched
k_range = list(range(1, 31))
print(k_range)
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30]
```

```
# create a parameter grid: map the parameter names to the values that should be searched
param_grid = dict(n_neighbors=k_range)
print(param_grid)
```

```
{'n_neighbors': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29,
30]}
```

```
# instantiate the grid
grid = GridSearchCV(knn, param_grid, cv=10, scoring='accuracy')
```

- You can set `n_jobs = -1` to run computations in parallel (if supported by your computer and OS)

```
# fit the grid with data
grid.fit(X, y)
```

3. הערה n_jobs=-1 מאפשר להריץ כמה תהליכים במקביל. התהליך הזה נקרא parallel. הוא למעשה משתמש בכל הליבות שיש לו (כיום למחשבים יש 4/8 וכו' ליבות) אתה מקצר את תהליך החיפוש פי מספר הליבות כי כל ליבה רצה במקביל ומחפשת.

לאחר מכן אני יכול לבקש ממנו להראות לי עבור כל k מה האידיאלי ומה סטיית התקן

```
# view the results as a pandas DataFrame
import pandas as pd
pd.DataFrame(grid.cv_results_)[['mean_test_score', 'std_test_score', 'params']]
```

	mean_test_score	std_test_score	params
0	0.960000	0.053333	{'n_neighbors': 1}
1	0.953333	0.052068	{'n_neighbors': 2}
2	0.966667	0.044721	{'n_neighbors': 3}
3	0.966667	0.044721	{'n_neighbors': 4}
4	0.966667	0.044721	{'n_neighbors': 5}
5	0.966667	0.044721	{'n_neighbors': 6}

```
# define the parameter values that should be searched
k_range = list(range(1, 31))
weight_options = ['uniform', 'distance']  # יוניפורם זה הברירת מחדל של קייאנאג, הוא מסתכל כמה קרובים אליו וממה יש יותר - מבחינת כמות
    דיסטנס זה לפי המרחקים הקטנים ביותר#
```

```
# create a parameter grid: map the parameter names to the values that should be searched
param_grid = dict(n_neighbors=k_range, weights=weight_options)
print(param_grid)
```

```
{'n_neighbors': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29,
30], 'weights': ['uniform', 'distance']}
```

```
# instantiate and fit the grid
grid = GridSearchCV(knn, param_grid, cv=10, scoring='accuracy')
grid.fit(X, y)
```

```
# view the results
pd.DataFrame(grid.cv_results_)[['mean_test_score', 'std_test_score', 'params']]
```

	mean_test_score	std_test_score	params
0	0.960000	0.053333	{'n_neighbors': 1, 'weights': 'uniform'}
1	0.960000	0.053333	{'n_neighbors': 1, 'weights': 'distance'}
2	0.953333	0.052068	{'n_neighbors': 2, 'weights': 'uniform'}
3	0.960000	0.053333	{'n_neighbors': 2, 'weights': 'distance'}
4	0.966667	0.044721	{'n_neighbors': 3, 'weights': 'uniform'}
5	0.966667	0.044721	{'n_neighbors': 3, 'weights': 'distance'}

ואז מגיעה לי שורת המחץ שלשמה עשינו את כל המודל והוא להבין מה הנתונים האידיאלים:

```
# examine the best model
print(grid.best_score_)
print(grid.best_params_)
```

!!!!!!!!!

```
0.9800000000000001
{'n_neighbors': 13, 'weights': 'uniform'}
```

כלומר בשביל לקבל את הדיוק הגבוה ביותר בחיזוי של פרח חדש בדאטה סט של האירוסים אני צריך לבחור knn=13 ומטריקת המרחק הוא לפי uniform כלומר – מבין ה13 הקרובים הללו מה יש הכי הרבה (ללא התחשבות במרחק. יכול להיות שיש כאלו שממש צמודים אליו בגרף אבל הם מועטים מאוד ביחס לאלו שמעט יותר רחוקים אבל הם יותר בכמות). במצב זה אני מקבל דיוק של 98%.

Random search

4. התחלנו לעשות grid search אבל בגלל שיש לנו המון פרמטרים יכול להיות שייקח למודל הרבה זמן/ שמשהו לא יעבוד – ניתן לעשות פשוט random כך שבמקום לבדוק את כל הקומבינציות האפשרויות ניקח הרבה אקראיים (אבל עדיין לא את כל האפשרויות) וכך אני יכול 'לדגום' מכל קבוצה עם שינויים קלים כך שיהיה שינוי שאולי ישנה. (ההיגיון – לדברים דומים יש נטייה להיות דומים/קרובים אחד לשני – לדוגמה סביר להניח שעבור knn עם 5 או 6 שכנים שאחוז הדיוק יהיה קרוב, לכן ידגום למשל עבור 2 ואז עבור 5 ואז עבור 10, וכך עושה קומבינציות מדגמיות).

```
from sklearn.model_selection import RandomizedSearchCV
```

```
# specify "parameter distributions" rather than a "parameter grid"
param_dist = dict(n_neighbors=k_range, weights=weight_options)
```

Important: Specify a continuous distribution (rather than a list of values) for any continous parameters

```
# n_iter controls the number of searches
rand = RandomizedSearchCV(knn, param_dist, cv=10, scoring='accuracy', n_iter=10, random_state=5)
rand.fit(X, y)
pd.DataFrame(rand.cv_results_)[['mean_test_score', 'std_test_score', 'params']]
```

	mean_test_score	std_test_score	params
0	0.973333	0.032660	{'weights': 'distance', 'n_neighbors': 16}
1	0.966667	0.033333	{'weights': 'uniform', 'n_neighbors': 22}
2	0.980000	0.030551	{'weights': 'uniform', 'n_neighbors': 18}
3	0.966667	0.044721	{'weights': 'uniform', 'n_neighbors': 27}
4	0.953333	0.042687	{'weights': 'uniform', 'n_neighbors': 29}
5	0.973333	0.032660	{'weights': 'distance', 'n_neighbors': 10}

מטריקות דיוק

היפוטזת 0 / מודל 0

5. מודל 0 – זה למעשה לקחת את המודל הכי פשוט. למשל – תמיד להגיד 'לא' עבור שאלה האם אישה בהריון היא חולה סוכרת או לא חולה סוכרת – אם 75% לא חולות – הוא יענה תשובה נכונה ב75%. ככלל – אם אתה עושה פחות מהמודל 0 (או היפוטזת 0) אז כנראה עשית שיבוש בנתונים ובאיזון הרסת יותר מהמצב הכי "טיפש".
6. מה יכל לגרום לכך שאקבל דיוק פחות מהיפוטזת 0?
   א. בחרנו פרמטר לא טוב/ שגוי לבחון על פיו
   ב. חוסר איזון במידע של הtest – למשל שלימדתי את המודל על פי תמונות של 100 כלבים ו2 חתולים.
   ג. שכחנו להתייחס לפרמטר משמעותי שמשפיע.
7. מודל 0 בבעיית קלסיפיקציה זה מובן איך זה יכול להתקיים – שתמיד אומר כלב למשל. אבל בבעיית קלסיפיקציה איזו היפוטזת 0 יכולה להיות? למשל מה המודל הכי פשוט למציאת מחיר של בית בשכונה? במצב זה נחפש למשל את הממוצע/ החציון

False positive /negative

8. נחזור כעת למושגים שלמדנו – false positive (נקרא גם שגיאה מסוג ראשון -Type 1) והשני - false negative (נקרא גם שגיאה מסוג שני -Type 2).

   נניח שיש לי דאטה סט של 100 אנשים ואני בודק האם הם חולי סוכרת של חולי סוכרת. מה שצדקתי בו זה האלכסון – ה40 שאמרתי שחולים ובאמת חולים, וה30 שאמרתי שלא חולים ובאמת לא חולים.

0 - לא חולה	1 - באמת חולה	
10 - FP	40 - TP	1 - אמרתי שחולה P
30 - TN	20 - FN	0 - אמרתי שלא חולה N

9. Recall או TPR (או true positive rank) – מתוך כל אלו שאמרתי שחיובי – כמה מהם **צדקתי** לגביהם. $\frac{TP}{TP+FN}$ (למשל יש לי 60 שכן חולים, אני על 40 מתוכם אמרתי שחולים ועל 20 שלא, הrecall שלי הוא $\frac{40}{60}$)
10. Precision – מתוך כל אלו שאמרתי שחיובי – כמה **באמת חיובי**. במילים אחרות – כמה אני מצליח לזהות את מה שאני אמור. $\frac{TP}{TP+FP}$ (למשל יש לי 60 שכן חולים, אני על 40 מתוכם אמרתי שחולים אבל יש לי 10 שלא הצלחתי לזהות שהם חולים, הrecall שלי הוא $\frac{40}{50}$)
11. יש כלל שעכשיו נוכל להבין – low precision = high recall, low recall = high precision

    נציג את הטבלה הבאה כך שזה יהיה מובחן יותר – לדוגמא בהיפוטזת 0 שאומר תמיד שחולה :

0 - לא חולה	1 - באמת חולה	
80 - FP	20 - TP	1 - אמרתי שחולה P
0 - TN	0 - FN	0 - אמרתי שלא חולה N

במצב זה נקבל $recall=\frac{20}{20}=1$ ו- $precision=\frac{20}{100}=\frac{1}{5}$

F1 score

12. למדנו את המושג מודל טיפש, למשל בודק אם יש או אין פצצה ואומר תמיד "אין פצצה" – הדיוק שלו הוא גבוה מאוד כי באמת לרוב האנשים אין פצצה. משוואת F1 מגדירה את היחס בין ה Recall לprecision. חישוב הF1 עוזר לי במצב זה כדי לבדוק - אם המודל טיפש ועדיין הדיוק יצא מאוד גבוה – ה F1 ייתן לי 0 או מספר מאוד נמוך.

$$F1 = \frac{precision*recall}{precision+recall} = \frac{2TP}{2*TP+FN+FP}$$

```
# Train-test split, intentionally use shuffle=False
X = x.reshape(-1,1)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20, shuffle=

# Create two models: Polynomial and linear regression
degree = 2
polyreg = make_pipeline(PolynomialFeatures(degree), LinearRegression(fit_intercept
linreg = LinearRegression()

# Cross-validation
scoring = "neg_root_mean_squared_error"
polyscores = cross_validate(polyreg, X_train, y_train, scoring=scoring, return_est
linscores = cross_validate(linreg, X_train, y_train, scoring=scoring, return_estim

# Which one is better? Linear and polynomial
print("Linear regression score:", linscores["test_score"].mean())
print("Polynomial regression score:", polyscores["test_score"].mean())
print("Difference:", linscores["test_score"].mean() - polyscores["test_score"].mea
```

```
 0.93333333 1.         1.         1.        ]

In [8]:
# use average accuracy as an estimate of out-of-sample accuracy
print(scores.mean())

0.9666666666666668

In [9]:
# search for an optimal value of K for KNN
k_range = list(range(1, 31))
k_scores = []
for k in k_range:
    knn = KNeighborsClassifier(n_neighbors=k)
    scores = cross_val_score(knn, X, y, cv=10, scoring='accuracy')
    k_scores.append(scores.mean())
print(k_scores)

[0.96, 0.9533333333333334, 0.9666666666666666, 0.9666666666666666, 0.9666666666666668, 0.966666666666
6668, 0.9666666666666668, 0.9666666666666668, 0.9733333333333334, 0.9666666666666668, 0.9666666666666
668, 0.9733333333333334, 0.9800000000000001, 0.9733333333333334, 0.9733333333333334, 0.97333333333333
34, 0.9733333333333334, 0.9800000000000001, 0.9733333333333334, 0.9800000000000001, 0.966666666666666
```

```
# Retrain the model and evaluate
import sklearn
linreg = sklearn.base.clone(linreg)
linreg.fit(X_train, y_train)
print("Test set RMSE:", mean_squared_error(y_test, linreg.predict(X_test), squared=Fal
print("Mean validation RMSE:", -linscores["test_score"].mean())
```

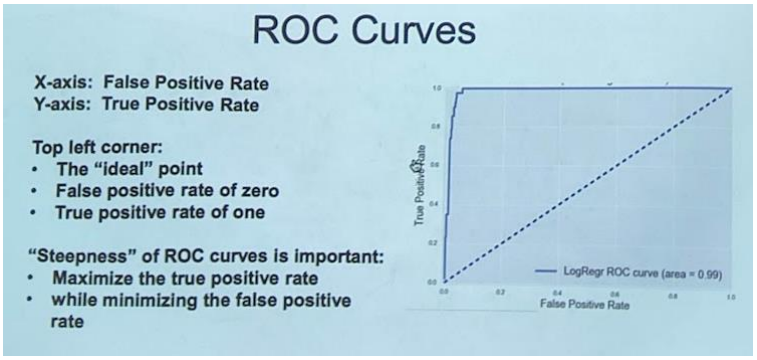
שיעור 11

Predict probability

1. כאשר אנו עושים מודל קלסיפיקציה למעשה אני רוצה לומר האם אובייקט קרוב לקלאס (/סוג) אחד או לקלאס אחד, באותה מידה ברגרסיה לינארית אני יכול לקבוע לו שאם הוא עבר תחום מסוים אז הוא מקלאס אחד ואם לא אז קלאס אחר.
2. נגענו לפני כן בעניין שאני לא רק רוצה לדעת לאיזה קלאס לדעת המודל האובייקט שבחנתי שייך אלא שאני רוצה גם לדעת עד כמה הוא בטוח בתשובה שלו. זה המשמעות של predict probability, כלומר הסתברות החיזוי. למשל אם בknn מתוך 5 שכנים יש 4 מסוג אחד ו1 מסוג אחר - הpredict probability תהיה 0.8.
3. הערה – נשים לב כי כאשר בknn אנו מחפשים עבור שכן אחד – ה predict probability יהיה 1 כלומר 100%. כי אין לו עוד התלבטויות. לכן חשוב לוודא שצורת החשיבה שלנו נכונה ולא משקפת מציאות לא מדויקת.

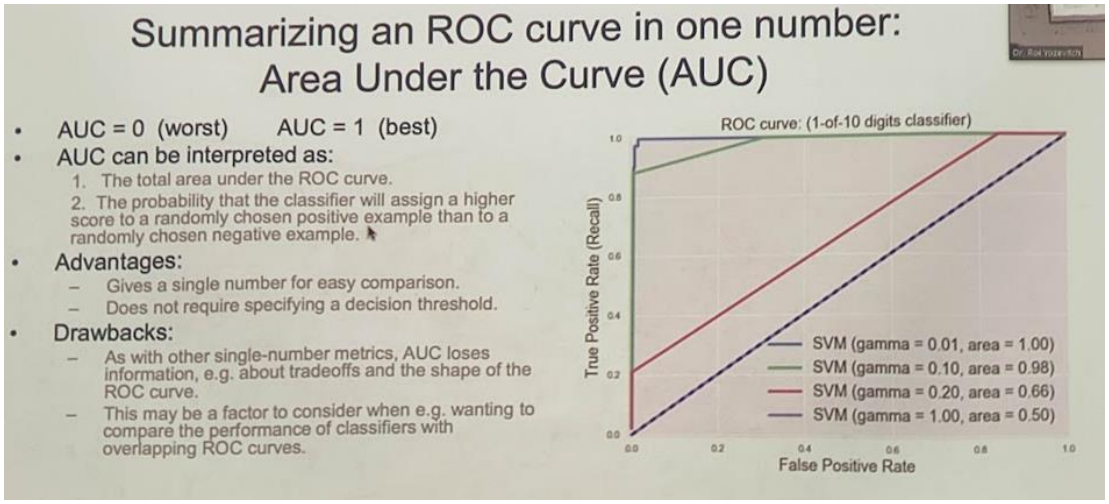
מודל ROC

1. גרף של false positive על true positive עבור כל תנאי סף מהברירה הכי מקילה – כלומר false positive נמוך, למשל אם אתה בטוח ב0.01 תחזיר positive, לבין התנאים הכי מקשים כלומר שהרף שממנו ניתן לומר positive – כלומר שבוודאות יהיה true positive גבוה, למשל אם אתה בטוח ב99.9 אז תחזיר חיובי.



2. נשים לב שהקו המקווקו הוא המודל 0 שלהם – שזה למעשה אם היינו עושים פשוט random כי אז אין נטייה לאף אחד מהצדדים.
3. הערה – הרף הזה שאותו נבחר כנקודת המפנה בין מה שאני מקבל לבין מה שלא – נקרא במונח המקצועי **threshold** .
4. לפי הבחירה של תנאי הסף - הtrash hold – אני בעצם מגדיר למודל roc כמה אני מוכן לטעות ולאיזה צד להטות את הטעויות שלי." לצד הFP או FN.

   ולכן קיימת עקומת ROC – שמעין דורשת ממני כדי "להיכנס" לחלק הרצוי - כמה ערכים אתה מוכן להקריב ולשייך לחלק שהן לא שייכים אליו.
5. בריבוע של ROC אנחנו רוצים שהעקומה תכניס כמה שיותר שטח של הריבוע– תתקרב ל"שפיץ" של 100% . השטח מייצג כמה הפרדתי בין ה TP לTN.



משתנים קטגוריים

6. למשל עבור הטבלה שמציגה לי למשל עמודה של מין – גברים ונשים – פחות נוח לי לעבוד עם משתנים המחולקים לקטגוריה לכן יש אפשרות לסווג אותם לערך מספרי, למשל שעבור נשים נרשום [0 , 1] ועבור גבר נרשום [1 , 0]
7. איך נעשה את זה בקוד?

	Survived	Pclass	Sex	Embarked
0	0	3	male	S
1	1	1	female	C
2	1	3	female	S
3	1	1	female	S
4	0	3	male	S

```
# dummy encoding of categorical features
from sklearn.preprocessing import OneHotEncoder
ohe = OneHotEncoder(sparse=False)
```

```
ohe.fit_transform(df[['Sex']])
```

```
array([[0., 1.],
       [1., 0.],
       [1., 0.],
       ...,
       [1., 0.],
       [0., 1.],
       [0., 1.]])
```

שיעור 12

לקראת הפרויקט

1. **דרך פעולה:**
   - להכיר את הדאטה ולעשות לו פרי-פרוססינג – עיבוד מקדים.
   - לעבור על העמודות ולראות ששמותיהן מבהירים בוודאות מה הן מייצגות.
   - לבדוק אם ישנם חוסרים כלשהם בדאטה ואיפה df.isnull().any() ואח"כ להחליט איך מתמודדים עם חוסרים בהתאמה לדאטה ולמה שהמודל שלי מחפש...

- Df.describe(include='all') אינפורמציה בסיסית על הדאטה – כמה נבדקים, כמה ערכים יש, מה הממוצע בכל עמודה, מה סטיית התקן בכל עמודה וכו' ובגלל שאנחנו בודקים עבור כל עמודה אז בעמודות שערכיהן מלל, ככל הנראה לא נקבל תשובה עבור "מקסימום" או "סטיית תקן"
- אפשר להשתמש בplt.boxplot(), plt.show() כדי לראות את הערכים החריגים ביותר כדי שההתפלגות שלי תהיה קרובה למציאות, ולהחליט אולי לוותר עליהם אם יש מספיק דאטה.
- השלב הבא הוא לראות התפלגויות של כל עמודה בנפרד.
אפשר להתחיל לסנן כל מיני תוצאות חריגות או לפחות לסמן. כדאי להבין האם הן חריגות אך מציאותיות או אולי חריגות עד כדי טעות בדאטה ואז כמובן שלא נרצה שהן ישפיעו לנו על המודל בצורה לא רצויה. (על הממוצע למשל...) אפשר להציג את ההתפלגויות בצורת פאי או בצורה אחרת שנוחה לעין.
כל הניתוח הראשוני מטרתו לבחון את הדאטה ולהתחיל לחשוב על דרכי הפעולה לעתיד.
רצוי לכתוב את ההערות שלנו- מדוע החלטנו להשאיר/ למחוק דאטה מסוימת וכו'.

- טקסט בדאטה: או שנמחק את העמודה אם היא לא קריטית למשל שם המכונית...
אבל למשל ב"סוג דלק" אם רצינו לחלק אותו לשלושה סוגים אבל כמובן שאין ביניהם יחס של טוב יותר או פחות. אז נוכל לפתור זאת בחלוקה לשלוש עמודות שבכל אחת מהן פשוט מסומן 0 אם זה לא הסוג שכתוב בראש העמודה ו1 אם כן – כי כמו שלמדנו – עדיף להימנע מלרשום את הסוגים למשל 1/2/3 כי אז האלגוריתם עלול להשוות ביניהם ולסווג כ "טוב יותר" את 3.
באותו אופן עדיף לעשות על זכר או נקבה וכו' אפשר לפצל לשתי עמודות.
למען פיצול עמודת סוג הדלק למשל: נשתמש ב pd.get_dumies (df ,columns=['full_Type'])

- נרמול הדאטה- לאחר החלוקה לקבוצות (train and test כמובן!) – נשתמש בscale.
לאחר שהדאטה נקי ומוכן-

שווה להשתמש בcorrelation matrix והקורלציה מראה לנו כמה אחוז מתוך השונות המשתנה "מסביר" "מנבא" לי. וככה יש לי איזושהי אינדיקציה מה המשקל של אותו משתנה במודל.
(באופן מופשט ולא מדויק- ככל שהאחוזים קרובים יותר לאחד אז בהסתמך על אותו משתנה נגיע לתוצאה מדויקת יחסית)

שווה להיזכר בהבדל והיחס בין train- test ל cross validation (שיעור 9 סעיף 6)

הדבר הראשון פיצול לחלק (תלוי בגודל הדאטה) ראשוני כך שישנה קבוצה של נתונים שלא משתתפת כלל באימון המודל בשום שלב! קריטי! וחייבים שיהיה חלק שלא נגענו בו בכלל! רק אחרי שפיצלנו חלק דאטה סטרילי נעבוד עם השאר ונעשה עליו test בסוף בסוף.

זה המהלך שמסיים את הפרויקט אי אפשר לבדוק שוב את המודל ולשפר את התוצאה כי אז כבר המודל מכיר את הדאטה ואחוז הדיוק שלו יהיה מוטה.

cross validation אפשר לעשות עם מודל אחד ולשנות פרמטרים. ואפשר לעשות עם כמה מודלים שונים וביניהם להשוות מי יותר טוב.

2. אתרים למשיכת הנתונים:
   - אתרים שניתן לשאוב מהם נתונים שיעזרו ספציפית לנושא שלנו – ע"י web scrolling
   - יבוא הנתונים מכתובת אתר API (דרך נוספת לשאיבת נתונים, חשוב לדעת אבל שהמון נתונים יהיו אולי לא קריאים)
   - Data.gov.il / Data.gov (– us)
   - הלשכה המרכזית לסטטיסטיקה.

3. הביצוע:
   - web crawling בעזרת beautifulsoup – משיכת דאטה מאתרים שכביכול לא מנגישים אותה. ישנו קורס על כך באתר קמפוס, של מכללת HIT בחולון. בנוסף מומלץ לראות סרטוני הדרכה ביוטיוב.
   - Import beautifulsoup

עבור כל כתובת ספציפית של נתונים מייבאים אותם לצורת טקסט , משהו שאני יכולה להדפיס / להכניס לטבלה .

Version control

4. בכל מחשב כאשר אנחנו מבצעים פעולה המחשב מעין 'זוכר' או 'מצלם' את הגרסאות כל כמה זמן כדי שיהיה ניתן לחזור או להשתמש בגרסה קודמת כאפיק מעודכן או חדש. למשל העובדה שיש לנו יכולת ללחוץ ctrl+z והמחשב יחזור לפעולה הקודמת מבוסס על version control שהמחשב צילם מה שהיה לפני כן. אופן שימוש נוסף הוא למשל כאשר בחברה עובדים על קוד מסויים, ויש עובד אחד שרוצה לנסות כיוון אחר לתכנות הוא יכול 'לייסד' נתיב חדש לזרימה של ה'צילומים' האלו שהמחשב עושה ואז כל עובד יכול להמשיך בנפרד על הקוד שלו, ואלי בשלב מסוים להתאחד חזרה לקוד משותף. זה הרעיון מאוחרי האתר git.
5. Git הוא בעצם מנהל גרסאות, כשכל גרסה כזו – שהיא מעין 'צילום תמונת מצב' נקרא commit, לcommit יש מזהה ספציפי (מעין id) שנקרא sha (דוגמה: e2adf8ae3e2e4) . כאשר יש לנו אוסף של גרסאות (ששמורות אחד אחרי השני כי נבנו ברצף כרונולוגי) הנקרא branch, שזה בעצם ענף, שניתן כמו שהסברנו – ליצור ענף חדש שיוצא מהענף המרכזי – במידה ורוצים לפתח את הקוד בכיוון חדש. הענף הראשי ממנו התחלנו נקרא master או main, שלאחר שמישהו יצא לענף חדש אנו נרצה בסוף לעשות merge, ולאחד את השינויים שלא קיימים בקוד הראשי. הקוד שנשמר בgithub שהוא ספריית האחסון על התשתית של git, אבל אם אנחנו מכירים מהמחשב שהכל שמור בתיקיות – אצלנו הקוד נשמרים במאגר- repository או בקיצור repo.

קריאה מומלצת

1. Hands on machine learning
2. machine learning engineering
3. גברת טועמת תה – בויקיפדיה
4. How to lay with statistics
5. האמת הלא נעימה על אינטיליגנציה
6. Islr – introduction statistical learning