בס"ד

סיכום מבוא ללמידת מכונה – קורס מבוא למדעי הנתונים

כפיר גולדפרב

למידת מכונה מתחלקת לשני מקרים שונים:

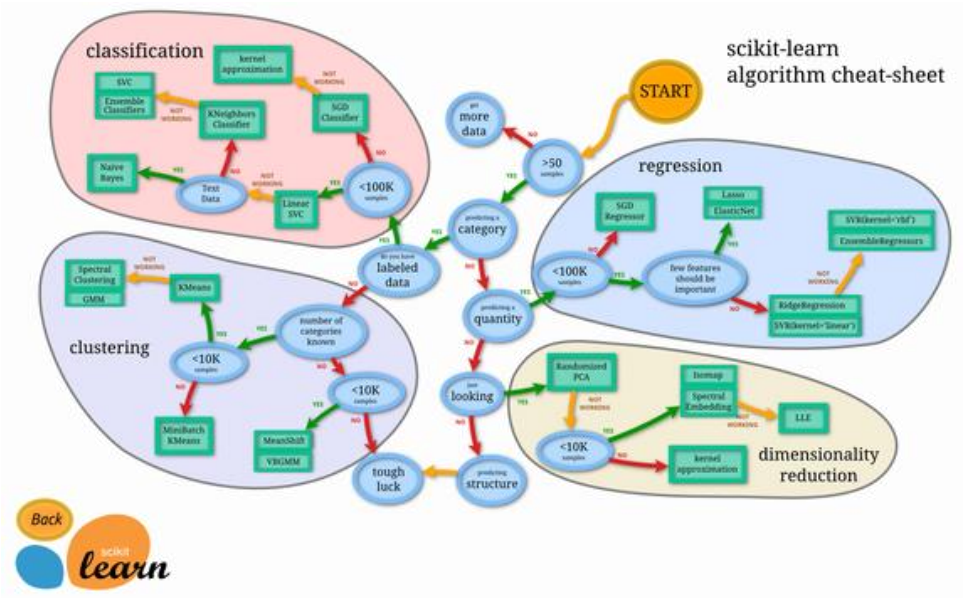
Supervised learning - המדען נתונים הוא זה שמחליט כמה נתונים לחלק ללמידת המכונה וכמה לבדיקת המכונה, המדען יחליט באיזה מודל להשתמש בסופו של דבר.

Unsupervised learning – המכונה עצמה תקבל את המידע ותנסה לבחור בעצמה את המודל לשימוש הטוב למידע בעזרת ממדים שתעשה.

- בקורס זה בהקשר של למידת במכונה מתעסקים בעיקר ב-Supervised learning .

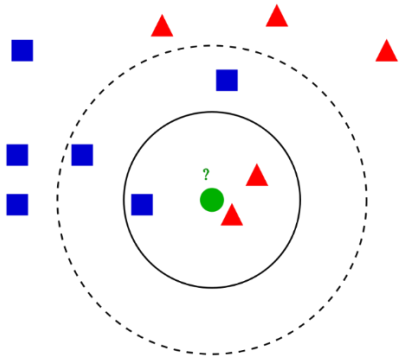
בלמדית מכונה יש מספר תחומים:

1. $\boldsymbol{classification}$ – בהינתן מידע שהוא נתון לסיווג $labeled - data$ אנחנו רוצים לסווג אותו, לאחת מכמה קטגוריות, ידועות מראש – היוכלת לחזות $class$ מ-$data$, מאמן את המכונה להבדיל בין המידע.
2. $\boldsymbol{Regression}$ – כאשר רוצים לסווג מחלקות לפי נתונים מספריים, למשל בה ליידי ביטוי מחירים (נדל"ן, בורסה וכו').
3. $\boldsymbol{Clustering}$ – לנסות לעשות סוג של $classification$ כאשר אין $label$ כלומר סידור לפי ההגיון.
4. $\boldsymbol{dimensional\ reduce}$ – כאשר ישנה כמות הפרמטרים עצומה שנותנת לי תוצאה גבוהה מאוד המערכת בודקת האם ניתן לצמצם בפרמטים כדי להגיע בכל זאת לתוצאות טובות ולא לפגוע בכשירות המכונה.

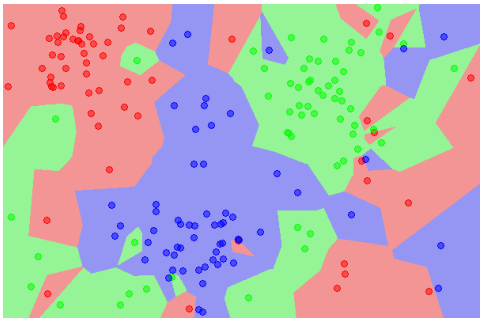


בקורס זה בעיקר נלמד בעיות **classification** בעזרת האלגוריתמים/סוגי מודלים הבאים:

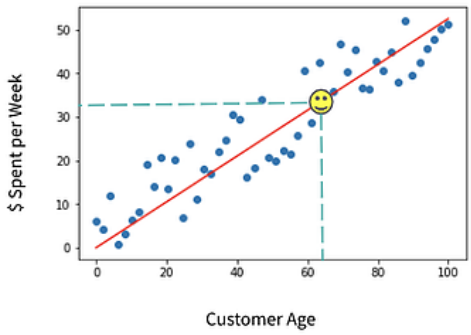
אלגוריתם $\boldsymbol{KNN}$ ($k\ nearest\ neightbors\ algorithm$) – המרה של הנתונים לנקודות בממערכת קרטזית – בהנתן נקודת חיזוי חדשה (predict) המודל יחשב מרחק בין k שכנים של הנקודה החדשה – הנקודה הקרובה ביותר תבחר להיות "דומה" לנקודה החדשה ולכן האלגוריתם ייצא מנקודת הנחה שמה שקרוב דומה.



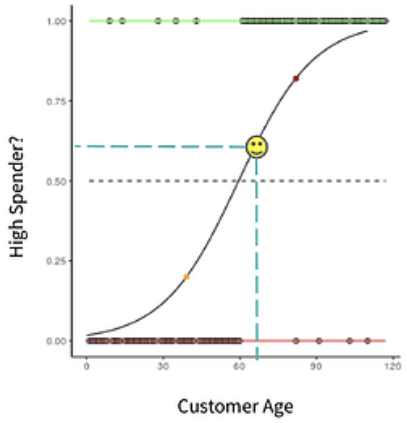
מרחק מנהטן מחלק את כל השטחים לתחומים של סוגים שונים :



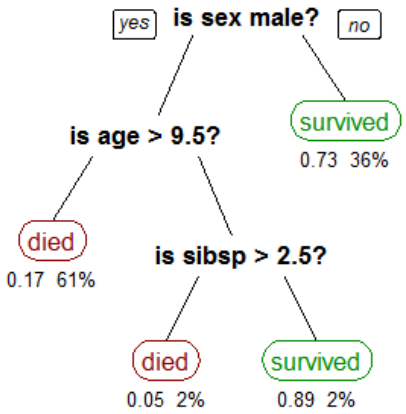
ריגרסיה לינארית – המרה של הנתונים לנקודות במערכת קרטזית (בדרך כלל דו-ממדי) – המודל ינסה לבנות קו ישר העובר בין כל ממוצאי הנקודות, קו זה יהיה תוצאה של פונקציה שהמודל בנה ובכך בהנתן נקודה חדשה x (predict) האלגוריתם יימצא את נקודת ה-y בעזרת משוואת קו הישר.



ריגרסיה לוגיסטית - המרה של הנתונים לנקודות במערכת קרטזית (בדרך כלל דו-ממדי) – המודל ינסה לבנות פונקציה פולינומית העוברת בין כל ממוצאי הנקודות, קו זה יהיה תוצאה של פונקציה שהמודל בנה ובכך בהנתן נקודה חדשה x (predict) האלגוריתם יימצא את נקודת ה-y בעזרת משוואת הפונקציה.



עץ החלטות – בהנתן המידע X המודל יבנה עץ החלטות לפי הנתונים האלה, המודל ינסה למצוא תנאים מסויימים שאיתם ניתן להפריד בין מקרים, ובהנתן נתון החדש הנתון יעבור בין כל צומת בעץ החלטה ולפי תנאי שיש בצומת כך ידע לאיזה צומת לעבור או לתת תוצאה מסויימת, לדוגמה:



סוגי למידת מכונה:

למידת מודל - model learning:

בנית מודל מסויים המקושר לכל הנתונים שיש לי, גם אם יש סטיות בנתונים המודל יבנה לפי רוב הנתונים, ברגע שיש לנו מודל יש לנו סונקציה המקשרת בין רוב הנתונים וכבר לא צריך דוגמאות כי יש לנו את הפונקציה.

היתרונות:

- לא צריך לשמור את הדוגמאות – חוסך זיכרון.
- לא לקוח הרבה זמן ריצה.

חסרון:

- המודל לפעמים טועה אך ברוב המקרים צודק.

למידה עצלנית / למידה מונחת דוגמאות - Memory / instance-based learning:

למשל knn זו למידה עצלנית, בגלל שעובדת רק לפי דוגמאות ולא לפי אג'נדה מסויימת, למדיה עצלנית מקבלת רק דומגאות מהזכרון וכל פעם שהיא עוברת על הדוגמא החדשה היא שומרת בזיכרון את הדוגמא ותוצואותיה ומשפרת את המודל

היתרון:

- דיוק גבוה.

חסרונות:

- זמן ריצה ארוך.
- דורש הרבה זיכרון.

סוגי מידע – Data Types:

- **מידע נומריאלי (מספרי) numerical data:**
  - מידע דיסקרטי (בדיד - מספרים שלמים) $how\ many? – discerte\ data$
  - מידע רציף (אינסופי – מספרים ממשיים) $how\ much?$ - $continuous\ data$
- **מידע קטגוריאלי categorial data:**
  - כל המידע שיש במידע קטגוריאלי הוא ערך לא בר השוואה וממוין רק לפי קטגוריות, גם אם יש בו ערך שהוא מספרי, לא ניתן להשוותו עם עם ערכים אחרי
- **מידע אורדינאלי ordinal data ערבוב בין מידע נומריאלי למידע קטגוריאלי:**
  - אפשר להשוות בין קטגוריות (למשל הסרט הנ"ל הוא טוב מהקודמו).

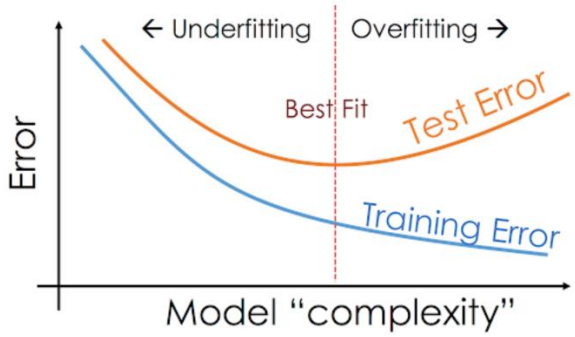
Overfitting and Underfitting:

כאשר נבנה מודל יש לנו ישנם מספר בעיות אפשריות:

underfitting - שהמודל פשוט מידי ואז המודל לא מספיק טוב בשביל הנתונים (datasets),

overfitting - שהמודל מורכב יותר מידי כך שמצד אחד יש פחות בעיות בלמידה על המודל כי הוא יותר מדוייק אך לאחר מכן בבדיקה על המודל יש מצבים שבהם המודל ישגה כיוון שהמודל מסובך מידי לנתונים החדשים,

נרצה כאשר אנחנו בונים מודל להגיע לנקודה האמצעית בין ה-overfitting לבין ה-underfitting, נקודה זאת נקראת best fit:



חילוק מידע ל-Training set and Testing set:

Training set – החלק איתו מאמנים את המודל.

Testing set – החלק איתו בודקים את המודל.

בדרך כלל נהוג לאמן את המכונה בעזרת 80% מהמידע הקיים ו-20% מהמידע כדי לבדוק אם המודל עובד כמו שצריך, אך האחוזים יכולים להשתנות בהתאם למדען הנתונים או אם יש מידע גדול מאודת יכול להיות שיספיק 70% ולפעמים גם פחות כדי לאמן את המכונה.

Sklearn:

הספריה הפופולארית ביותר של פייתון עבור למידת מכונה ולמידה עמוקה, ניתן לקרוא עוד ב-https://scikit-learn.org/stable.

לאחר שנבחר מודל להשתמש בו נרצה לאמן אותו ולבדוק אותו בעזרת מידע, על מנת לעשות זאת יש 2 פונקציות עקריות שצריך להכיר:

- *model.fit(x,y)* – משתמש ב-x כדי ללמוד ממנו וב-y כדי להתאמן עליו.
- *model.predict(z,w)* – לבדוק דוגמה חדשה (בדיקת *class*).

נשים לב ש-x זה ה-training set שלנו וה-y הוא ה-testing set שלנו.

אחרי שנאמן את המודל ניקח את ה-testing set שלנו ונחלק אותו לשני נתונים שונים, נתון אחד יכיל רק את המידע שאיתו המודל יכול לעשות predict ונתון שני יכיל רק את התוצאות שאמורות להתקבל אחרי predict (במקרה שלנו z ו-w) והפונקציה predict תוכל להגיד לנו כמה המודל הצליח לעשות predict בצורה טובה.

טבלת סיווג תוצאות אפשריות כאשר מודל עושה predict:

	תצפיות בפועל		
תצפיות חזויות		*בריא*	*חולה*
	בריא	True Positive	False Positive Type 1 Error
	חולה	False Negative Type 2 Error	True Negative

מודל דיוק – Model Accuracy:

נוסחאות חישוב ממדים שונים על מנת לדעת עד כמה המודל שלנו מדויק:

$$\boldsymbol{accuracy} = \frac{TP + TN}{TP + TN + FN + FP}$$

עוד מדד חשוב:

$$\boldsymbol{true\ positive\ rate \backslash TPR \backslash recall} = \frac{TP}{TP + FN}$$

עוד מדד חשוב:

$$\boldsymbol{precision} = \frac{TP}{TP + FP}$$

עוד מדד חשוב:

$$\boldsymbol{F\ score} = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$