למידת מכונה

רגרסיה ליניארית – Linear Regression

בהינתן סט של נקודות וה-label המתאים שלהם $\{x_1, x_2, \dots, x_n, t\}$, נרצה למצוא היפוטזה (לאמן מודל) אשר מוצא מינימום עבור פונקציית השגיאה MSE.

$$D = \{X_1, \dots, X_m\} \equiv \text{dataset with } m \text{ rows and } n \text{ features}$$
$$X_i = \{x_1, \dots, x_n\} \equiv \text{a row in the dataset}$$
$$T = \{t_1, \dots, t_m\} \equiv \text{each row's label}$$

$$y = h(X) = w_0 + x_1 w_1 + \cdots + x_n w_n \equiv \text{hypothesis/model}$$
$$Y = \{h(X_1), \dots, h(X_m)\} = \{y_1, \dots, y_m\} \equiv \text{hypothesis results/model guesses}$$
$$MSE(w_0, \dots, w_n) = \frac{1}{m}(T - Y)^2 = \frac{1}{m}\sum_{i=1}^{m}(t_i - y_i)^2 \equiv \text{loss function}$$
$$\downarrow$$
$$\text{find}\{W = (w_0, \dots w_n)\}: \min\big(MSE(W)\big)$$

פרקטיקה חשובה – נירמול ה-dataset

1. Normalization/Scaling:

נזיז את כל הערכים סביב $[-1,1]$ כך שהממוצע יהיה 0.

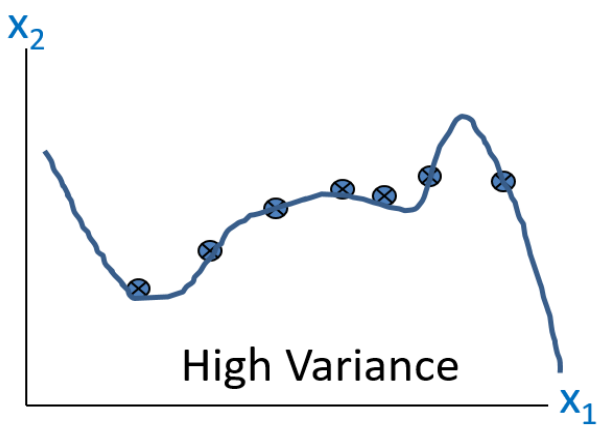
$$X = \frac{X - min(X)}{max(X) - min(X)} = \text{minmax scaling}$$
or
$$X = \frac{X - mean(X)}{max(X) - min(X)} = \text{mean normalization}$$

2. Standardization:

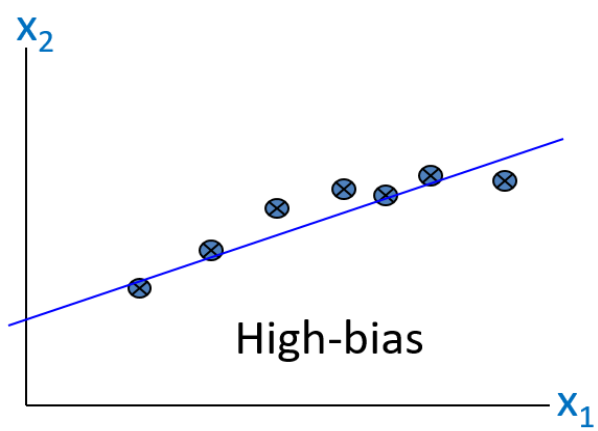
נזיז את כל הערכים סביב $[0,1]$ כך שסטיית התקן תהיה 1.

$$X = \frac{X - mean(X)}{sd(X)}$$

Overfitting & Underfitting



Overfitting (degree 5)



Underfitting (linear)

איך נמצא ווקטור משקולות W אשר נותן מינימום ל-MSE?

1. ישירות בעזרת אלגברה ליניארית, $O(n^3)$, בעזרת הנוסחא הבא:

$$W = (X^T X)^{-1} \cdot X^T \cdot T$$

לדוגמא:

X	T		W
$\begin{pmatrix} 1 & 70 & 3 \\ 1 & 80 & 3 \\ 1 & 85 & 4 \end{pmatrix}$	$\begin{pmatrix} 100 \\ 120 \\ 125 \end{pmatrix}$		$\left(\begin{pmatrix} 1 & 70 & 3 \\ 1 & 80 & 3 \\ 1 & 85 & 4 \end{pmatrix}^T \begin{pmatrix} 1 & 70 & 3 \\ 1 & 80 & 3 \\ 1 & 85 & 4 \end{pmatrix}\right)^{-1} \cdot \begin{pmatrix} 1 & 70 & 3 \\ 1 & 80 & 3 \\ 1 & 85 & 4 \end{pmatrix}^T \cdot \begin{pmatrix} 100 \\ 120 \\ 125 \end{pmatrix} = \begin{pmatrix} -25 \\ 2 \\ -5 \end{pmatrix}$

2. GD – Gradient Descent:

נבצע צעדים קטנים עד לקבלת מינימום מקומי, או עד שהשינוי ב-MSE יהיה קטן מספיק.
נעדכן את וקטור המשקולות בסוף כל epoc.

$$\lambda \equiv \text{Learning Rate, usually } 0.01$$

$$\Delta\vec{W} = \lambda \cdot \nabla loss_h(\vec{W}) = \forall_{i=1}^{n} \left\{ \lambda \frac{\partial loss_h(\vec{W})}{\partial w_i} \right\}$$

נחשב עבור כל שורה ב-dataset וקטור משקולות חדש.
קיימות 2 גרסאות עבור האלגוריתם הנ"ל,

- בגרסא בראשונה כל שורה ב-dataset נחשב ב-epoc ולכן נעדכן את וקטור המשקולות לאחר כל שורה.

$$\vec{W} = \vec{W} + \Delta\vec{W}$$

- בגרסא השנייה, נקרא גם SGD, ה-epoc מסתיים לאחר שסיימנו לעבור על k שורות מה-dataset, ולכן נבצע מיצוע עבור כל המשקולות החדשות שמצאנו בכל שורה ורק אז נעדכן את וקטור המשקולות.

$$\vec{W} = \vec{W} + \frac{\Delta\vec{W}}{k}$$

3. LWRLR – Locally Weighted Linear Regression:

בהינתן וקטור x, נחשב וקטורי משקלים $\beta_i \in [0,1]$ לדוגמאות האימון לפי מרחקיהם מ-x.
ככל שדוגמא יותר "דומה" ל-x כך היא תקבל β_i יותר קרוב ל-0.
ככל שדוגמא יותר "שונה" תקבל משקל יותר קרוב ל-1.

$$\tau \equiv \text{LWR constant} =$$
$$= \text{higher means further } x_i \text{ are taken into consideration, more flexibale} =$$
$$= \text{lower mean futher } x_i \text{ are } \textbf{not} \text{ taken into consideration, less flexibale}$$

$$\forall_{i=1\ldots m}: \left\{ \beta_i = e^{\frac{-\|x_i - x\|^2}{2\tau^2}} \right\}$$

$$WeightedMSE \equiv WMSE = \frac{1}{2m} \cdot \sum_{i=1}^{m} \beta_i (wx_i - t_i)^2$$

רגרסיה לוגיסטית – LR – Logistic Regression

- לא רגרסיה, אלא קלסיפיקציה.
- שימוש עם cross entropy.

משתמשים בפונקציית sigmoid עבור חיזוי הקלסיפיקציה, הפונקציה תחזיר ערכי הסתברות עבור קלסיפיקציה.
ערכים הגדולים מ-$\frac{1}{2}$ יסווגו כחיוביים.
ערכים הקטנים מ-$\frac{1}{2}$ יסווגו כשליליים.

$$z = w_0 + \sum_{i=1}^{n} w_i x_i = WX \equiv \text{distance } x \text{ from plane } w$$

$$y = g(z) = \frac{1}{1+e^{-z}} \equiv \text{sigmod function}$$

CE – Cross Entropy

נשתמש בפונקציית loss זו לחישוב עלות השגיאה פר דוגמא.

$$C(y,t) = -t \cdot ln(y) - (1-t) \cdot ln(1-y) =$$
$$= \begin{cases} -ln(y) & : \text{if } t = 1 \\ -ln(1-y) & : \text{if } t = 0 \end{cases}$$
$$loss_h(W) = \frac{1}{m} \cdot \sum_{i=1}^{m} C(y_i, t_i)$$

- ניתן להשתמש ב-CE כפונקציית ה-loss בשביל אלגוריתם SGD.

Multiclass Classification Using Binary Classifiers

One vs Rest

בגישה זו נבנה Classifier עבור כל קטגוריה (נבנה k מודלים).

- אימון: נאמן להפריד קטגוריה אחת מכל שאר הקטגוריות.
- חיזוי: בהינתן דוגמא, נבדוק את החיזויים של כל המודלים ונחזיר כתשובה את הקטגוריה בעלת החיזוי עם ההסתברות הגבוהה ביותר.

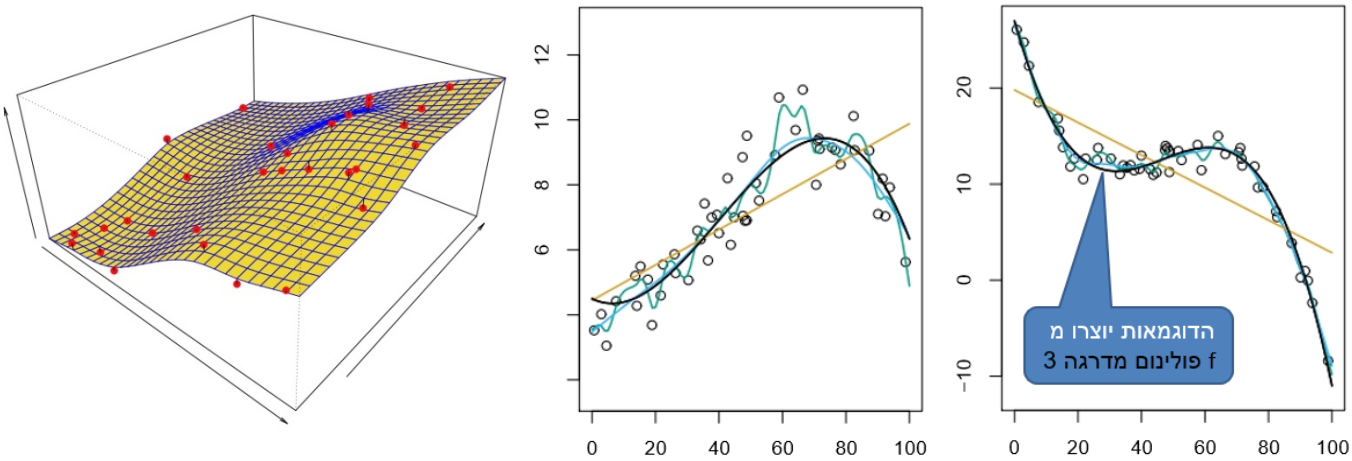
One vs One

גישה זו פחות פופולרית מהגישה הקודמת, בעיקר בגלל כובד החישוב שלה.
בגישה זו נאמן מודל עבור כל זוג קטגוריות ($k \text{ nCr } 2 = \frac{k(k-1)}{2}$ מודלים).

- אימון: נאמן להפריד כל זוג קטגוריות.
- חיזוי: Majority Vote – בהינתן דוגמא, נחזיר כתשובה את הקטגוריה בעלת מספר החיזויים הגדול ביותר.
- וויראציה לחיזוי: Weighted – ניתן משקל לכל מודל ע"פ השגיאה הממוצעת שעשה, נחשב את ההסתברות הממוצעת לכל קטגוריה ונבחר כתשובה את הקטגוריה בעלת ההסתברות המקסימלית.

רגרסיה בעזרת Thin Plate Spline

נשתמש בספליינים של לוח דק בין כל מספר דוגמאות אימון עבור מציאת היפוטזה כוללת גמישה במיוחד.



Cross Validation

באופן כללי,

$$CV_{loss} = \frac{1}{|V|}\sum_{i=1}^{|V|} loss_i$$
$$|D| = m$$

$K\ Fold$

1. נחלק את D ל-k קבוצות בגודל $\frac{m}{k}$ כל אחת.
2. כל פעם נבחר $k-1$ קבוצות בתור קבוצת אימון, והקבוצה האחרונה תהיה קבוצת הוולידציה.
3. נבצע k סבבים k מודלים שונים.

$Leave\ k\ Out$

1. נבחר קבוצת וולידציה רנדומית בגודל k.
2. נאמן מודל על כל שאר הדוגמאות, זוהי קבוצה בגודל $m-k$.
3. נבצע $(m\ \mathrm{nCr}\ k)$ סבבים עם קבוצות וולידציה שונות (כל פעם קבוצה שונה לחלוטין נבחרת!).

$Bootstrap$

1. נבחר באופן רנדומי (עם חזרות) m דוגמאות מ-D.
2. נקבל קבוצת אימון בעלת "כפילויות" ולכן ישנן דוגמאות אשר אינן נמצאות בקבוצת האימון.
   דוגמאות אלה יהפכו לקבוצת הוולידציה.
3. נבצע k סבבים עם קבוצות אימון וולידציה שונות.

פירוק ה-MSE

תזכורת, מה למידה מונחית מנסה לעשות?

- נתונים זוגות של קלט-פלט (קבוצת אימון)
- רוצים להסיק (ללמוד) פונקציה דטרמיניסטית f שממפה את הקלט לפלט.

נניח הנחה נאיבית שהתוחלת של ε (הרעש) מתאפסת ושהיא לא תלוייה בקלט (לא בהכרח מתקיים במציאות).

$$t = f(\vec{X}) + \varepsilon$$

טענה

$$\boldsymbol{MSE = (Bias)^2 + Variance + irreducable}$$

הוכחה

1. נפרק את MSE לשגיאה פריקה ולשגיאה לא-פריקה.

$$MSE = E_x, E_D\left[\left(t_x - h_D(x)\right)^2\right] = E\left[\left(f(x) + \varepsilon - h_D(x)\right)^2\right] =$$
$$= E\left[\left(f(x) - h_D(x)\right)^2 + 2\varepsilon\left(f(x) - h_D(x)\right) + (\varepsilon - 0)^2\right] =$$
$$= E\left[\left(f(x) - h_D(x)\right)^2\right] + 2 \cdot E[\varepsilon] \cdot E[f(x) - h_D(x)] + E[(\varepsilon - 0)^2] =$$
$$= E_x, E_D\left[\left(h_D(x) - f(x)\right)^2\right] + Var(\varepsilon) = \text{reducable}_{\text{err}} + \text{irreducable}_{\text{err}}$$

$E[\varepsilon] = 0$

הגדרת
ווריאנס/שונות

$$\boldsymbol{MSE = E[(h_D(x) - f(x))^2] \quad + \quad Var(\varepsilon)}$$

Reducable Error
תוחלת ריבועי ההפרשים בין f ל h

IrrReducable Error
השונות של ε

2. נפרק את השגיאה הפריקה ל-2 השגיאות המרכיבות אותה.

let: $\overline{h_D(x)} = E[h_D(x)]$

$$MSE_{reducable} = E\left[\left(h_D(x) - f(x)\right)^2\right] = E\left[\left(h_D(x) - \overline{h_D(x)} + \overline{h_D(x)} - f(x)\right)^2\right]$$
$$= E\left[\left(h_D(x) - \overline{h_D(x)}\right)^2 + 2\left(h_D(x) - \overline{h_D(x)}\right)\left(\overline{h_D(x)} - f(x)\right) + \left(\overline{h_D(x)} - f(x)\right)^2\right] =$$
$$= E\left[\left(h_D(x) - \overline{h_D(x)}\right)^2\right] + 2 \cdot \boldsymbol{E}\left[\boldsymbol{h_D(x) - \overline{h_D(x)}}\right] E\left[\overline{h_D(x)} - f(x)\right] + E\left[\left(\overline{h_D(x)} - f(x)\right)^2\right] =$$
$$= E_x, E_D\left[\left(h_D(x) - \overline{h_D(x)}\right)^2\right] + 0 + (E_x, E_D[h_D(x) - f(x)])^2 =$$
$$= Variance[h_D(x)] + (Bias[h_D(x), f(x)])^2$$

$\boldsymbol{E[\ldots] = 0}$

KPIs in Confusion Matrix

אחוז אבחנות שפירות שגויות מתוך "החולים בפועל" - מיזעור FN

Confusion Matrix		Predicted Value: חיזוי חיובי	Predicted Value: חיזוי שלילי	
Actual True Value	באמת חיובי 1	True Positive – TP (correctly predicted as positives)	False Negative – FN (incorrectly predicted as negatives) (Type II error)	Recall =Sensitivity =TPR predicted positives out of all actual positives: $\frac{TP}{TP + FN} = \frac{TP}{\text{positives}}$
	באמת שלילי 0	False Positive – FP (incorrectly predicted as mpositives) (Type I error)	True Negative – TN (correctly predicted as negatives)	Specificity =TNR predicted as negatives (out of all actual negatives): $\frac{TN}{FP + TN} = \frac{TN}{\text{Negatives}}$
		Precision % true malignants out of all predicted as malignant: $\frac{TP}{TP + FP} = \frac{TP}{\text{PredictedPositives}}$		Accuracy % total correct predictions: $\frac{TP + TN}{\text{All Examples}}$

"חולים באמת" מתוך החזויים כחולים - (מדידת FP)

אבחנות "בריאים" שגויות מקרב ה"בריאים באמת" - מדידת FP

sensitivity, recall, hit rate, or true positive rate (TPR)

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{P}} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} = 1 - \mathrm{FNR}$$

specificity, selectivity or true negative rate (TNR)

$$\mathrm{TNR} = \frac{\mathrm{TN}}{\mathrm{N}} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}} = 1 - \mathrm{FPR}$$

precision or positive predictive value (PPV)

$$\mathrm{PPV} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} = 1 - \mathrm{FDR}$$

negative predictive value (NPV)

$$\mathrm{NPV} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FN}} = 1 - \mathrm{FOR}$$

miss rate or false negative rate (FNR)

$$\mathrm{FNR} = \frac{\mathrm{FN}}{\mathrm{P}} = \frac{\mathrm{FN}}{\mathrm{FN} + \mathrm{TP}} = 1 - \mathrm{TPR}$$

false discovery rate (FDR)

$$\mathrm{FDR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TP}} = 1 - \mathrm{PPV}$$

false omission rate (FOR)

$$\mathrm{FOR} = \frac{\mathrm{FN}}{\mathrm{FN} + \mathrm{TN}} = 1 - \mathrm{NPV}$$

prevalence threshold (PT)

$$PT = \frac{\sqrt{TPR(-TNR + 1)} + TNR - 1}{(TPR + TNR - 1)}$$

threat score (TS) or critical success index (CSI)

$$\mathrm{TS} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN} + \mathrm{FP}}$$

accuracy (ACC)

$$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{P} + \mathrm{N}} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}$$

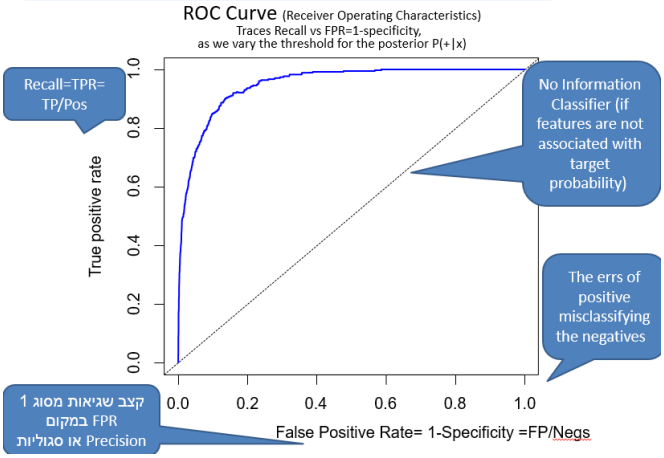
balanced accuracy (BA)

$$\mathrm{BA} = \frac{TPR + TNR}{2}$$

F1 score

is the harmonic mean of precision and sensitivity

$$\mathrm{F}_1 = 2 \cdot \frac{\mathrm{PPV} \cdot \mathrm{TPR}}{\mathrm{PPV} + \mathrm{TPR}} = \frac{2\mathrm{TP}}{2\mathrm{TP} + \mathrm{FP} + \mathrm{FN}}$$



Feature Selection

השיטה האקזוסטיבית

1. לכל תת קבוצה V_i של קבוצת הפיצ'רים V נבנה מודל M_{V_i} ונבדוק את שיערוך השגיאה (CV).
2. נבחר ב-V_i שנתנה את השגיאה הנמוכה ביותר.

שיטה לא פרקטית, $O\left(2^{|V|}\right)$

Forward Selection

תהליך איטרטיבי. נתחיל מתת קבוצת פיצ'רים ריקה. $F = \emptyset$

1. לכל פיצ'ר v_i שלא נמצא ב-F, בנה $F_i' = F + \{v_i\}$.
2. בנה מודל $M_{F_i'}$ ובדוק איזו תת קבוצה F_i' השיגה שגיאה מינימלית.

   הוסף את הפיצ'ר v_i המתאים ל-F.
3. חזור לשלב 1.

סיים את התהליך כאשר אין יותר פיצ'רים להוסיף או עד שהשגיאה לא קטנה יותר.

4. בסיום התהליך, הרץ CV על כל המודלים שנבנו בשלב 2 ובחר את המודל הטוב ביותר.

   מס' מקסימלי של אימונים: $\frac{n(n+1)}{2} \to O(n^2)$

Backward Selection

תהליך איטרטיבי. נתחיל מתת קבוצת פיצ'רים מלאה. $F = V$

5. לכל פיצ'ר v_i שנמצא ב-F, בנה $F_i' = F - \{v_i\}$.
6. בנה מודל $M_{F_i'}$ ובדוק איזו תת קבוצה F_i' השיגה שגיאה מינימלית.

   הורד את הפיצ'ר v_i המתאים מ-F.
7. חזור לשלב 1.

סיים את התהליך כאשר אין יותר פיצ'רים להוריד (כשהגענו לפיצ'ר יחיד) או עד שהשגיאה לא קטנה יותר.

8. בסיום התהליך, הרץ CV על כל המודלים שנבנו בשלב 2 ובחר את המודל הטוב ביותר.

   מס' מקסימלי של אימונים: $\frac{n(n+1)}{2} \to O(n^2)$

Hybrid Selection

שילוב של שתי השיטות. נתחיל מתת קבוצת פיצ'רים ריקה.

1. הוסף פיצ'ר שנותן תוצאה טובה ביותר (Forward)
2. הורד פיצ'ר שמזיק לתוצאה (Backward)

וכמובן שנשמור את המודל שבנינו בכל איטרציה ובסוף התהליך נחפש את המודל הטוב ביותר עם CV.

רגולריזציה – הענשת משקולות

נוסיף לפונקציית ה-$loss$ שלנו "עונש" למשקולות של האיברים בעלי חזקה גבוהה, נקבל פונקציית היפוטזה "חלקה" יותר.
נגדיר את γ בקבוע הרגולירזיצה, העונש למשקולת שערכה 1.

$$loss_{reg}(W) = loss(W) + \gamma \|W\|_n$$

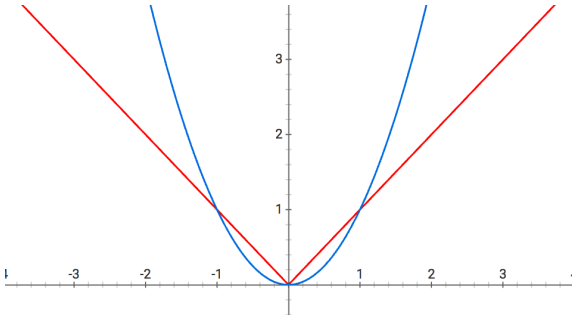
חשוב מאוד לנרמל הדאטא שלנו (חילוק בסטיית תקן) כדי שהמשקולות כולן יהיו באותו scale.

רגולריזציית Ridge – L2 – מרחק אוקלידי

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

$$w_i = w_i - \lambda \frac{\partial loss}{\partial w_i} - (\gamma \cdot w_i) =$$

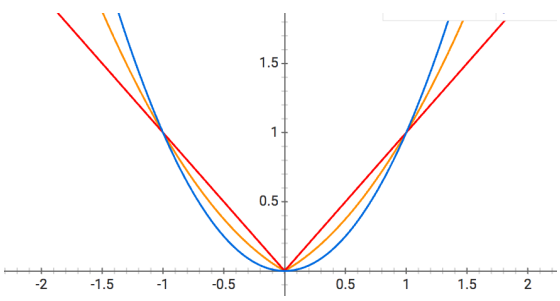
$$= w_i(1-\gamma) - \lambda \frac{\partial loss}{\partial w_i}$$



רגולריזציית Lasso – L1 – מרחק מנהטן

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

$$w_i = w_i - \lambda \frac{\partial loss}{\partial w_i} - (\gamma \cdot sign(w_i))$$



L1+L2 – Elastic Net

שילוב של Ridge ושל Lasso, מן ממוצע משוקלל.
במשקולות קטנות ה-Lasso ישפיע יותר.
במשקולות גדולות ה-Ridge ישפיע יותר.

Ensembles

Bagging

נאמן מודלים שונים על תתי קבוצות שונות של D.
נחשב את הפלטים על כל המודלים השונים שאימנו ונמצע את התוצאות (או שנסווג ע"פ הצבעת הרוב).
שיטות ש-Bagging יכול להסתמך עליהן:

1. Bootstrap
2. K-Fold

Boosting

נאמן **סדרה** של מודלים חלשים בשביל לקבל מודל חזק בסוף, כל מודל M_i מתמקד בדוגמאות שהמודל הקודם M_{i-1} טעה בהן.
נחשב את הפלטים על כל המודלים השונים שאימנו ונמצע את התוצאות (או שנסווג ע"פ הצבעת הרוב).

נשתמש באותו $Dataset$ אך נכפיל את הדוגמאות שבהן המודלים הקודמים טעו בהן.
נקבל סדרה של מודלים שכל אחד חזק "בתחום" אחר. הצבעת הרוב או מיצוע של התוצאות תפיק תוצאה סופית טובה יותר מתוצאה של כל מודל לחוד.

K-Nearest Neighbors – KNN

בהינתן אוסף דוגמאות אימון D ודוגמאת מבחן חדשה x,
נמצא את k הדוגמאות "הקרובות" ביותר ל-x, נקרא לקבוצה זו N_x.
נשערך לפי השכנים של x את ההסתברות של כל הקטגוריות הקיימות t_i, ונבחר בקטגוריה עם ההסתברות הגבוהה ביותר.
ולכן התחזית עבור הדוגמא x היא ע"פ הצבעת הרוב מתוך קבוצת N_x השכנים.
מידת "הקירבה" נמדדת ע"י פונקציית דימיון כלשהי, לדוגמא נורמה $L2$ או $L1$.

$$N_x = \{\text{k nearest samples } (x_i \in D) \text{ using } d(x, x_i)\}$$

קלסיפיקציה: $\mathbf{return}: \max\{\forall_{i \in N_x}: P_i(y = t_i \mid \{x, D\})\}$

רגרסיה: $\mathbf{return}: \text{mean}(N_x)$

פונקציות דימיון – משתנים נומריים

$$CosineSimilarity(A, B) = \frac{A \cdot B}{\|A\|_2 \cdot \|B\|_2} = \frac{\sum_{i=1}^{n}(a_i b_i)}{\sqrt{\sum_{i=1}^{n}(a_i)^2} \cdot \sqrt{\sum_{i=1}^{n}(b_i)^2}}$$

$$PearsonCorrelation(A, B) = 1 - CosineSimilatiry(A, B)$$

פונקציות דימיון – משתנים קטגוריאליים

$$ValueDofferenceMeasure \equiv VDM(val_A, val_B) = \sum_{i=1}^{|categories|} (|P(c_i|val_A) - P(c_i|val_B)|)^n$$

$$GravityLaw(A, B) = \frac{1}{d(A, B)^2}$$

$$GausSimilatiry(A, B) = e^{\frac{-d(A,B)^2}{2\tau^2}}$$

Weighted KNN

נשקלל כל שכן ע"פ מידת הדימיון שלו לדוגמאת ה-test.
ניתן להשתמש ב-$GausSimilatiry$.
τ משפיע כמו סטית התקן על רוחב צורת הגאוסיאן, כש-τ קטן נקודות רחוקות מהתוחלת מקבלות משקל קטן.

Bayesian Learning

מבוסס על תורת ההסתברות ובפרט על חוק Bayes:

$$P(A|B) = \frac{P(A,B)}{P(B)} = \frac{P(B|A) \cdot P(A)}{P(B)}$$

כלל מופעים Mutually Exclusive:

$$P(A \vee B) = P(A) + P(B) - P(A \wedge B)$$

$$[\text{let } \{\forall_{i=1}^{k} A_i\} \text{ be mutually exclusive}]$$
$$P(B) = \sum_{i=1}^{k} P(B, A_i) = \sum_{i=1}^{k} P(B|A_i)P(A_i)$$

חוק Bayes עבור מופעים mutually exclusive:

Posterior Probability of A_i given B

$$P(A_i|B) = \frac{P(B|A_i) \cdot P(A_i)}{\sum_{i=1}^{k}\left(P(B|A_i) \cdot P(A_i)\right)}$$

Prior Probability of A_i

Prior Probability of B

קלסיפיקציה

$$K = \{k_i\} \equiv \text{all possible categories of } x$$
$$P(k|x) = \frac{P(x|k) \cdot P(k)}{P(x)} = \frac{P(x|k) \cdot P(k)}{\sum_{i=1}^{|K|}\left(P(x|k_i) \cdot P(k_i)\right)}$$

Maximal A-Posteriori Estimation – MAP

נצטרך להשיג את כל הסתברויות ה-prior (הסתברות לקטגוריה בהינתן x <u>באוכלסיה כולה</u>).

$$y_{MAP} = argmax_k\{P(k|y=x)\} = argmax_k\left\{\frac{P(x|y=k) \cdot P(k)}{\sum_{i=1}^{|K|}\left(P(x|k_i) \cdot P(k_i)\right)}\right\} = \text{drop the מכנה} =$$
$$= argmax_k\{\left(P(x|y=k) \cdot P(k)\right)\}$$

מכנה משותף עבור כל קטגוריה k

בדרך כלל קשה מאוד להשיג את ההסתברויות הללו ולכן נסתפק ב-Maximal Likelihood.

Maximal Likelihood Estimation – ML

כאשר לא ניתן לשערך את ה-priors נניח הנחה שכל הקטגוריות שוות הסתברויות,
ולכן נוכל למקסם רק את $P(x|y=k)$.

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

Naïve Bayes Assumption

כאשר x הוא בעל מימד נמוך או שהמאפיינים הם בעלי מספר קטן של ערכים אפשריים ולעיתים יהיו מספיק נתונים ב-D כדי לשערך את $P(x|k)$ ע"י ספירה של הדוגמאות.

אך כאשר המימד הוא גדול או המאפיינים בעלי מספר גדול של ערכים אפשריים, לרוב לא יהיו מספיק דוגמאות ב-D והשערוך להסתברות לא יהיה מדויק. נזדקק להנחה נאיבית:

המאפיינים אינם תלויים זה בזה בהינתן קטגוריה k_i

ולכן הסתברות מותנית שווה לכפל ההסתברויות.

$$P(v_1, v_2, \dots, v_n|k) = \prod_{i=1}^{n} P(x = v_i|y = k)$$

$$\downarrow$$

$$y_{MAP} \approx argmax_k \left\{P(k) \cdot \prod_{i=1}^{n} P(x = v_i|y = k)\right\}$$

Naïve Bayes Classification – NBC

foreach k in $Categories$:

$P(k) \approx P(y = k) \approx \frac{n_k}{n}$

foreach v_i in x_j:

$P(v_i|k) \approx P(x_j = v_i|y = k) \approx \frac{n_{v_{i,k}}}{n_k}$

return $argmax_k \left\{P(k) \cdot \prod_{i=1}^{n} P(v_i|k)\right\}$

לעיתים נקבל הערכות להסברויות הקרובות ל-1 או ל-0 (כמובן שזה לא דבר ריאלי).

לשם כך "נתקן" את השיערוכים הללו בעזרת $m - estimation$.

M-Estimation

בשביל לתקן את השיערוכים כאשר אין מספיק דוגמאות מקטגוריה מסויימת, נוסיף דוגמאות וירטואליות מאותה קטגוריה.

$$P(x = v_i|y = k) \approx \frac{n_{v_{i,k}} + m \cdot P(x = v_i)}{n_k + m}$$

Laplace Smoothing

כאשר נתונים V ערכים ועבור חלקם לא ניתן לשערך את הסתברות ה-priors ע"י ספירה ישירה,

נניח התפלגות אחידה בין הערכים $v_1, v_2, \dots, v_n$ ונשערך:

$$P_{prior}(x = v_i) = \frac{1}{|V|}$$

אם נוסיף עוד $|V|$ דוגמאות וירטואליות במקבלות ערך $x = v_i$ בהסתברות $\frac{1}{|V|}$ נקבל:

$$P(x = v_i|y = k) \approx \frac{n_{v_{i,k}} + 1}{n_k + |V|}$$

עצי החלטה

נתונה קבוצה D של דוגמאות עם מאפיינים בינאריים ותווית מטרה $t = \{0 \text{ or } 1\}$.

אם הקבוצה היא הומוגנית (אחידה) מבחינת התווית, ניצור ממנה עלה.
אחרת, נבחר מאפיין "שיפצל הכי טוב" ל-2 קבוצות נפרדות. המאפיין שיבחר יהיה המאפיין שיפצל ל-2 קבוצות הכי הומגניות לפי t.

GrowTree(S):
if (t=0) for all <x,t> ϵ S, return (new leaf(0))
Else if (t=1) for all <x,t> ϵ S, return (new leaf(1))
Else
choose "best" feature x_j
S_0 =all <x,t> ϵ S with x_j=0
S_1 =all <x,t> ϵ S with x_j=1
return (new node (x_j<1,GrowTree(S_0), GrowTree(S_1))

נתונה קבוצה S של דוגמאות עם מאפיינים בינאריים ותווית מטרה בינארית y=0 או y=1

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

אחרת (S איננה הומגנית): נבחר מאפיין שיפצל "הכי טוב" ל 2 קבוצות נפרדות. כמה שיותר הומוגניות ב y

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

קריטריוני עצירה:

- שום פיצול אינו מפיק IG חיובי.
- כאשר אחוז ה-labels שבמיעוט (השגיאה) קטן מ-ε.
- עומק העץ גדול מ-$MaxDepth$ כלשהו.
- גודל העץ גדול מ-k כלשהו.
- ועוד..

Entropy

האנטרופיה מודדת את מידת ההומגניות של קבוצת אירועים.
האנטרופיה H של משתנה מקרי קטגורי V מוגדר ב:

$$P(v_i) \approx \frac{count(y = v_i)}{|D|}$$
$$H(v_i) = -P(v_i) \cdot \log_2 P(v_i)$$
$$H(V) = \sum_{v_i \in V} H(v_i)$$

Entropy משוקלל

$$H_{WMean}(V) = \sum_{v_i \in V} P(v_i) \cdot H(v_i)$$

Information Gain – IG

כעת נגדיר מושג חדש עבור עצי ההחלטה, Information Gain מגדיר כמה חוסר וודאות (אנטורפיה) הורדנו כאשר הוספנו צומת חדר לעץ ההחלטה.
בתהליך בניית העץ לפני הוספת צומת חדש נבדוק את ה-IG של כל המאפיינים שעדדין לא שומשו בענף שלו.
מבין כל המאפיינים, נבחר לפצל את המאפיין עם ה-IG הכי גדול.

$$children = \{c_1, c_2, \dots, c_k\}$$
$$IG(parent, children) = H(parent) \cdot H_{WMean}(children)$$

Random Forests + Bagging

בכל פעם שנבחר צומת לפיצול, נשתמש בהיפר-פרמטר m' המגביל את הבדיקה למספר מאפיינים הנבחרים רנדומית מתוך m המאפיינים הקיימים.
מקובל להשתמש ב-$m' = \sqrt{m}$.

מדד Gini

מדד נוסף המודד חוסר וודאות, בדומה ל-Entropy.

$$G(V) = \sum_{v_i \in V} P(v_i) \cdot \left(1 - P(v_i)\right)$$

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

Maximal Margin Classifier

מסווג בינארי ל-2 ערכים $\{+1, -1\}$, רוצים למצוא היפר מישור המפריד בין הדוגמאות, קבוצת האימון והוולידציה חייבות להיות ניתנות להפרדה לינארית **מלאה**, אחרת אין פתרון כלל.

נקודות מעל המישור יסווגו כ-$+1$: $w_0 + w_1 x_1 + \cdots + w_n x_n > 0$
נקודות מתחת למישור כ--1 : $w_0 + w_1 x_1 + \cdots + w_n x_n < 0$
וקטור דוגמא x יסווג כסימן של מרחקו מגבול ההחלטה.

בשביל למצוא את היפר המישור עם הגבולות המקסימליים נצטרך לפתור בעיית אופטימיזציה קוודרטית (בדומה לסימפלקס הליניארית).

$$\text{Maximize: } (M, w_0, w_1, \dots, w_n)$$
$$\text{Subject to: } \begin{cases} \forall (x_i, y_i) \in D: y_i\left(w_0 + w_1 x_{i,1} + \cdots + w_n x_{i,n}\right) \geq M \\ \sum_{j=1}^{n} \left(w_j\right)^2 = 1 \end{cases}$$

Soft Margin Classifier

נחפש קלסיפיקציה שאינה מכריחה את כל הדוגמאות שלנו להיות מעבר לצד יחיד של מישור ההפרדה, בעצם מאפשר outliers ורעש.

נוסיף משתי עודף ε_i לכל דוגמא, משתנה זה יאפשר לדוגמא להופיע בצד הלא נכון של הגבול.

יהיה לנו "תקציב" שאנחנו מוכנים לשלם עבור הנקודות החורגות, ככל שהחריגה גדולה יותר כך גם התשלום ε_i.

$$\varepsilon = \begin{cases} \varepsilon_i = 0 \rightarrow \text{correct side} \\ 1 \geq \varepsilon_i > 0 \rightarrow \begin{cases} \text{wrong side} \\ \text{inside the margin} \end{cases} \\ \varepsilon_i > 1 \rightarrow \text{wrong side} \end{cases}$$

בעיית האופטימיזציה החדשה:

$$\text{Maximize: } (M, w_0, w_1, \dots, w_n, \varepsilon_1, \dots, \varepsilon_m)$$

$$\text{Subject to: } \begin{cases} \forall (x_i, y_i) \in D: y_i(w_0 + w_1 x_{i,1} + \cdots + w_n x_{i,n}) \geq M(1 - \varepsilon_i) \\ \varepsilon_i \geq 0 \\ \sum_{i=1}^{m} \varepsilon_i < C \\ \sum_{j=1}^{n} (w_j)^2 = 1 \end{cases}$$

היפר-פרמטר "התקציב" C.
מגביל את סכום החריגות מ-M

C גדול: יותר "תקציב" לחריגות – מודל פחות גמיש – שקול ל-underfitting.

C קטן: פחות "תקציב" לחריגות – מודל יותר גמיש – שקול ל-overfitting.

$C = 0$: שקול ל-Max Margin.

Support Vector Machines – SVM

משתמש בטריק ה-Kernel, פסאודו-המרה של הווקטורים למימד אחר לצורך חישוב גבול החלטה טוב יותר עבור הבעיה הנתונה. פונקצית Kernel היא פונקציה שמכמתת דימיון בין 2 וקטורים.

הפונקציה צריכה לקיים תכונות מתמטיות מסויימות, אולם לצרכים פרקטיים כמעט כל פוקנציית דימיון תכיל תכונות אלו.

במקום לחפש וקטור W שימקסם את M ויקיים את שאר האילוצים,

נחפש וקטור של $V = \{\alpha_i\}$ שימקסם את M תחת אותם אילוצים.

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

$$w_j = \sum_{i=1}^{m} \alpha_i \cdot x_{i,j}$$

- אין צורך לעבור למרחב הגדול למען החישוב, מספיק לחשב את הדימיון בין כל זוגות הנקודות. במקרה הגרוע נקבל סיבוכיות של $O(m^2)$.
- המעבר למרחב הגדול נעשה באופן לא מפורש, יתרון גדול כי בהרבה אפליקציית המרחב הגדול הוא עצום .
- בזמן אימון צריך לחשב את הדימיון של כל הזוגות, אך בזמן חיזוי לנקודה x מספיק לחשב את הדימיון שלה רק לוקטורי התמך.

קרנלים פופולאריים:

$$\text{Linear: } K(x_1, x_2) = x_1 x_2$$

$$\text{Polynomial: } K_d(x_1, x_2) = (x_1 x_2)^d$$

$$\text{Gaussian: } K(x_1, x_2) = exp\left(\frac{-\|x_1 - x_2\|}{2\sigma}\right)$$

דרך נוספת להתסכל מיקסום ה-Margin

במקום למקסם את ה-margin נוכל לשמור אותו בגודל קבוע, $M = 1$.
אבל במקומו נמזער את נורמת המשקולות $\|w\|$.

נקבל בעיית אופטימיזציה חדשה:

$$\text{Minimize: } \|w\|$$

$$\text{subject to: } \forall (x_i, y_i) \in D: \begin{cases} y_i(w \cdot x_i) \geq 1 - \varepsilon_i \\ \varepsilon_i \geq 0 \\ \sum_{i=1}^{m} \varepsilon_i < C \end{cases}$$

ניתן להראות בעזרת לגרנגיאן כי ניתן להפוך את האילוצים לפונקציית loss ולהראות שבעיית האופטימיזציה הקוודרטית שקולה למזעור הפונקציה הבאה:

$$minimize_w \left\{ \sum_{i=1}^{m} \max[0, 1 - y_i h(x_i)] + \gamma \sum_{j=1}^{n} w_j^2 \right\}$$

Hinge loss - פונקצית loss שממזערת חריגות מ M=1.
- כש yh(x) גדולה מ1 - ה loss הוא 0.
- אם x בתוך ה margin, הloss בין 1-0.
- אם x בצד השני של גבול ההחלטה, loss>1

Ridge Regularization:
Penalty γ controls bias-variance in the same way as C

$for\ every\ \ (x_i, yi) \in D$

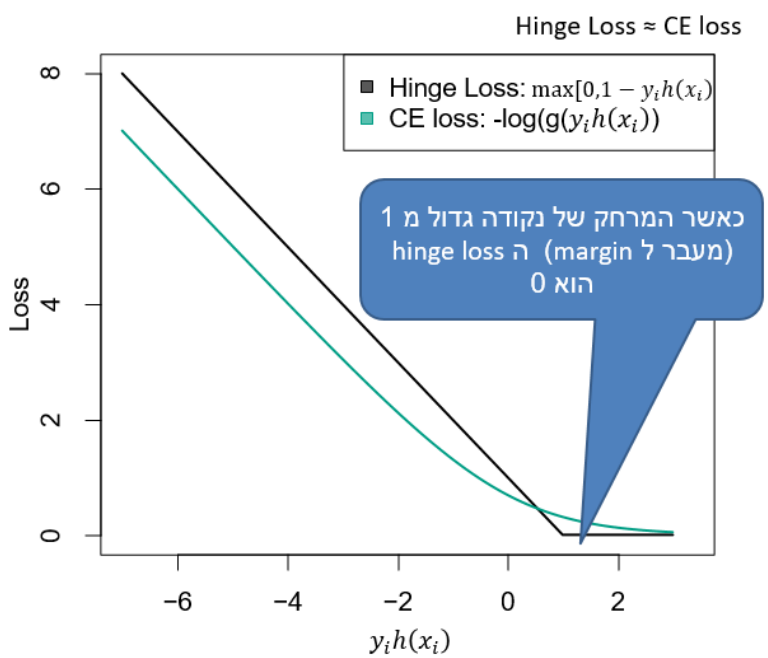
תחת האילוצים הרגילים:

$y_i\ (wx_i) \geq\ (\mathbf{1} - \boldsymbol{\varepsilon_i})$

$\varepsilon_i\ \geq 0,$

$\sum_{i=1}^{m} \varepsilon_i < C$

מסקנה:
SVM שקול לביצוע $GD + hinge$ עם רגורליזצית $L2$,
בהנחה ש-GD שימצא מינימום גלובלי.



Clustering

בהינתן קבוצה D של דוגמאות רוצים לחלק לקבוצות זרות שמכסות את D ומכילות דוגמאות "דומות" אחת לשניה ושונות מכל הדוגמאות של הקבוצות האחרות.

K-Means Clustering

נשתמש בפונקציית "אי-דימיון" בין 2 ווקטורים, למשל מרחק אאוקלידי בריבוע:

$$d(X,Y)^2 = \sum_{i=1}^{n} (x_i - y_i)^2$$

נצטרך גם מדד לשוני בין נקודות **בתוך** הקלסטר:

$$\text{WithinClusterVariation} \equiv WCV(C_k) = \frac{1}{|C_k|} \sum_{\forall (X,Y) \in C_k} d(X,Y)^2$$

$$TotalWCV = \sum_{k=1}^{|clusters|} WCV(C_k)$$

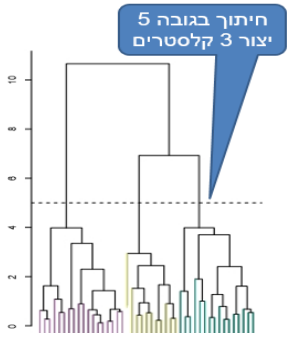
אלגוריתם הקליסטור מבוסס על האופטימיזציה:

$$\text{Minimize: } \{TotalWCV\}$$

1. מצרפים רנדומית כל דוגמא לאחד מ-K הקלסטרים.
2. חוזרים שוב ושוב עד שאין שינוי בקלסטרים:
   i. מחשבים את המרכז לכל קלסטר.
   ii. מצרפים מחדש את הנקודות על פי קירבתם למרכזי הקלסטרים (המרכז הוא ווקטור הממוצעי המאפיינים של הדוגמאות בקלסטר).

Bottom-Up Method

השיטה המקובלת, בשיטה זו נתחיל כשכל דוגמא ב-D היא קלסטא בפני עצמה (עלה בעץ).
בכל איטרציה נאחד את 2 הקלסטרים הכי דומים בעץ, נמשיך לאחד עד שנקבל קלסטר יחיד המכיל את D כולו.



נקבל דנדוגרמה.
בגובה 0 ישנם $|D|$ קלסטרים שכל אחד בגודל 1.
בכל שלב נקבל פחות ופחות קלסטרים עד שנגיע לקלסטר יחיד בעל $|D|$ איברים.

ציר ה-y משקף את דרגת הקרבה בין קלסטרים, ככל ששני קלסטרים יותר רחוקים (בציר y), כך הם פחות דומים.

מדידת אי-דימיון בין קלסטרים – Inter Class Dissimilarity

Linkage (betwwen clusters) types:

1. Maximal (complete) Dissimilarity:
   - Compute all pairwise dissimilarities d(x,y) between x∈C1 and y∈C2
   - Output the Maximal dissimilarity

   הכי פופולרים: Maximal and Average
   בד"כ מיצרים דנדוגרמות מאוזנות

2. Average
   - Compute all pairwise dissimilarities d(x,y) between x∈C1 and y∈C2
   - Output the average of the dissimilarities

3. Minimal (single)

   לא פופלרי : נטיה לייצר דנדוגרמות לא מאוזנות

   - Compute all pairwise dissimilarities d(x,y) between x∈C1 and y∈C2
   - Output the Minimal dissimilarity (e.g. the minimal road length from C1 &C2)

   בשימוש ב Genomics: מיצר בעיות בוויזואליזציה - גובה קלסטר
   ממוזג עלול להיות קטן מהקלסטרים שמרכיבים אותו

4. Cetroid
   - Output the dissimilarity d(center(c1),center(c2)) between the centers of the two clusters . Usually the center of a cluster is the average point.

Principle Component Analysis – PCA

טרנספורמציות ליניאריות ממימד n למימד קטן יותר m המנסות לשמר כמה שיותר אינפורמציה באמצעות קומבינציות ליניאריות של המאפיינים המקוריים.

עבור PCA בדרך כלל נרצה למרכז את המאפיינים סביב ל-0 (נירמול המאפיינים).

לעיתים נרצה גם לבצע Scaling (חילוק בסטיית התקן) כדי ליצור סטיית תקן 1.

אבל במידה וישנם 2 מאפיינים הנמדדים באותה סקאלה, יתכן מאוד שנרצה לבצע Scaling שווה לכל אחד כדי שההבדלים ביניהם כן יבואו לידי ביטוי.

m Principle Components

המרחקים האוקלידיים של הנקודות מהציר מתמזערים ובאותו רגע גם מתמקסמת השונות על הציר.

נתון מדגם D, נחפש טרנספורמציות ליניאריות שיעבירו את הקלט המקורי x ממימד n לוקטור חדש z ממימד קטן יותר m.

טרנספורמציות שנחפש הן: $\phi_1, \phi_2, \dots, \phi_m$ – וקטורים ממימד n.

z_i הוא ההטלה של דוגמא x_i על ציר ה-i:

$$z_i = \sum_{j=1}^{n} (\phi_{j,i} \cdot x_j)$$

Z הוא וקטור ההטלות של X על מערכת צירים חדשה בעלת m צירים אורתוגונליים.

המטרה:

למצוא m קטן ככל האפשר המספק m טרנספורמציות ϕ_i שישמרו אינפורמציה "חשובה" עבור הוקטור המקורי.

כל אחת מהטרנספורמציות ϕ_i נקרא Principle Component מס' i.

ה-Principle Component הראשון הוא הכי משמעותי, ה-PC השני הוא שני בחשיבותו וכן הלאה.

אלו הן בעצם בעיות אופטימיזציה מקסימליות, נראה עבור ה-PC הראשון:

$$z_{i,1} = \sum_{j=1}^{n} (\phi_{j,1} \cdot x_{i,j})$$

$$\text{Maximize:} \left\{ \frac{1}{m} \sum_{i=1}^{m} (z_{i,1})^2 \right\}$$

מיקסום הווריאנס

$$\text{subject to:} \left\{ \left(\sum_{j=1}^{n} (\phi_{j,1})^2 \right) = 1 \right\}$$

נכתב ע"י נעם לוי

לפי המצגות של דר גדי פנקס.

לכל טענה/בקשה/מענה ניתן לפנות ל:

https://tinyurl.com/ml-mashov