# Lez6_Evaluation_metrics

November 1, 2023

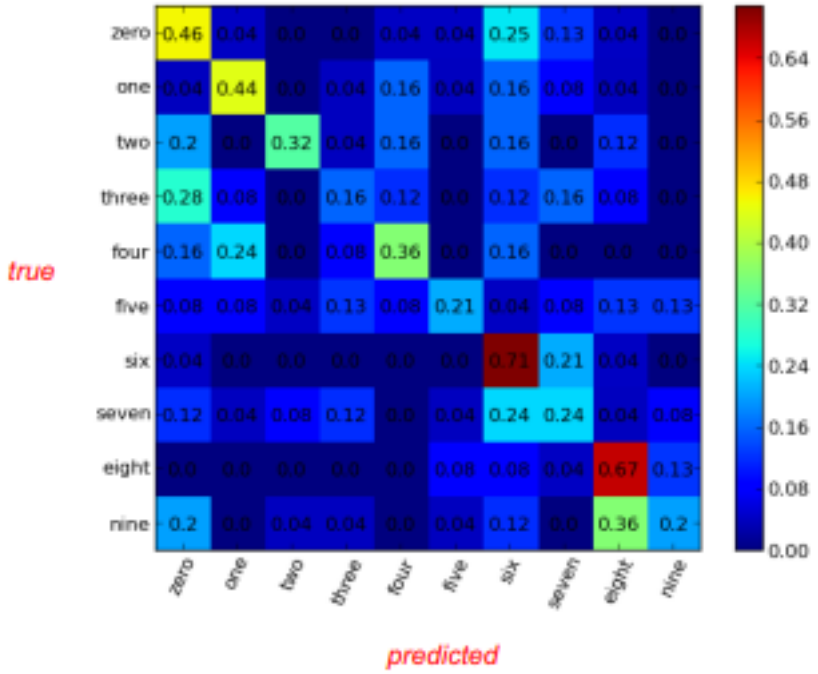
## 1 Lez6 Evaluation Metrics Lo Presti

### 1.1 Metriche di valutazione

Quando eseguiamo un algortimo come valutiamo il risultato della nostra soluzione? Di solito cerchiamo di minimizzare la funzione costo (questo è quello che facciamo per ottimizzare la nostra soluzione).

Differenziamo due casi: - *Classificazione*: si parla di *accuratezza* (rapporto tra quanti valori ho classificato correttamente rispetto a tutti i pattern che ho a disposizione), l'errore è il suo complemento. In formule: $accuracy = \frac{numero\ pattern\ classificati\ correttamente}{numero\ pattern\ totali} \cdot 100\%$ $error = 100\%$ - accuracy - *Problemi di Regressione*, si parla di *Root Mean Squadred Error*, legato alla funzione obiettivo usata in questi casi. $RMSE = \sqrt{\frac{1}{N}\sum_i (y^{(i)} - t^{(i)})^2}$

**Matrice di confusione** Essa viene nella classificazione per esaminare la distribuzione dell'errore.




Sulla diagonale, un classificatore perfetto ha tutti valori 1.0, sulle altre celle è 0.0 Come si legge? Prendiamo la prima riga, nella prima coordinata <*zero,zero*> leggiamo 0.46, vuol dire che nel 46%

dei casi lo zero viene riconosciuto correttamente come tale. Andando avanti, prendiamo <*sei,zero*>, dove abbiamo 0.25, questo vuol dire che nel 25% dei casi, lo zero è riconosciuto come sei.

### 1.2 Falsi Positivi e Falsi Negativi

Qui abbiamo a che fare con casi binari. *True positive (TP)* e *True Negative (TN)* sono gli elementi che il classificatore classifica correttamente. Identifichiamo:

- **True Positive (TP)**: Un modello positivo è stato correttamente classificato come positivo. *Esempio*: etichetto "Positivo" e lo faccio bene (True).
- **True Negative (TN)**: Un modello negativo è stato correttamente classificato come negativo. *Esempio*: etichetto "Negativo" e lo faccio bene (True).
- **False Positive (FP)**: Un modello negativo è stato erroneamente classificato come positivo. (Errore di Tipo I). *Esempio*: etichetto "Positivo" e lo faccio male (False). Quindi in realtà era "Negativo".
- **False Negative (FN)**: Un modello positivo è stato erroneamente classificato come negativo. (Errore di Tipo II). *Esempio*: etichetto "Negativo" e lo faccio male (False). Quindi in realtà era "Positivo".

L'accuratezza sono i valori che sono classificati correttamente in percentuale accurancy $= \frac{TP+TN}{P+N}$, l'*errore* è dato da $1 - accurancy$.
Spesso non bastano per valutare un modello, perciò si utilizzano anche precision e recall.

Esempio: $\begin{bmatrix} & Y & N \\ Y & 49 & 1 \\ N & 1 & 49 \end{bmatrix}$ $\begin{bmatrix} & Y & N \\ Y & 1 & 1 \\ N & 1 & 97 \end{bmatrix}$

Entrambi hanno accuratezza del 98%, non mi pesa quanto è grave l'aspetto che sto considerando.

### 1.3 Precision & Recall

L'accuratezza non tiene conto che gli errori che facciamo potrebbe essere grave. Mi vengono in aiuto altri due *indici*:

- **Precision**: la frazione delle predizioni positive che sono corrette P(is pos | predicted pos), la precisione è 1 è quando, predicendo qualcosa, non sbaglio. precision $= \frac{TP}{TP+FP}$, ma non mi dice nulla su quelli che ho classificato male. (Infatti nel denominatore, faccio solo predizioni *Positive*, sia che siano corrette (*True*), sia che siano scorrette (*False*) ). Questo dato da solo non basta, potrei avere un dataset molto semplice nella predizione, oppure potrei fare una sola predizione, aver ragione, ed automaticamente avere una precision massima! (é come un calciatore che tira e segna un solo rigore in carriera, risulta precisione 100%). Risponde alla domanda: *Ho predetto x volte "Positivo", quante volte avevo ragione su queste predizioni?* nb: Qui si parla di predizioni, non di come sia effettivamente il valore vero del dato.
- **Recall**: la frazione delle predizioni positive che sono correttamente identificate, non voglio dire a qualcuno che sta male che non ha niente. P(predicted pos | is pos) recall $= \frac{TP}{TP+FN}$ (quante volte un elemento della classe C è stato etichettato effettivamente come elemento della classe C). Da sola non basta, perchè se dicessi sempre "Positivo", allora $FN = 0$ (infatti non dico mai Negativo) e quando ho davanti un True Positive lo identifico per forza, allora recall è massima. Qui al numeratore abbiamo il numero totale di positivi veri, non delle

predizioni come sopra. Rispondiamo alla domanda: In totale ci sono $x$ positivi, quanti ne ho individuati?

Ci danno quindi informazioni sul tipo di errore che andiamo a fare (a seconda degli ambiti un tipo di errore è più grave di un altro).

Vorrei che fossero entrambi 1, presi singolarmente non bastano (o almeno il più alti possibile).

### 1.4 F1 SCORE

Poiché *Precision* e *Recall* sono entrambi importanti e colgono aspetti diversi spesso si vuole una metrica che combini con un unico valore entrambi. Per fare questo viene utilizzata la F1 score, che è una media armonica di precisione e recall
(media armonica = $\frac{\text{numero di valori che sto valutando}}{\text{reciproco dei valori}}$)

$F1 = \frac{2}{\frac{1}{Precision}+\frac{1}{Recall}} = 2\frac{Precision*Recall}{Precision-Recall}$

Nella media armonica il peso maggiore viene dato al valore più piccolo (si ha il reciproco).
Più i valori sono alti più F1 è alta. Tipicamente non si usa il minimo delle due, F1 ha anche il vantaggio di essere differenziabile.

## 2 Clustering - Unsupervised Learning

Parliamo di Learning non supervisionato, dove **non ho etichette**, bensì partiziono in gruppi di punti simili. Simili come? Noi assumeremo punti vicini come simili tra loro.

**Quando può esserci utile questa classificazione?** Utile per capire se i dati hanno struttura nascosta, per organizzarli. Pensiamo ad cluster che rappresenta l'interazione tra utenti, oppure persone con stesse gusti, o anche classificazione di galassie e stelle. In questi casi non abbiamo mai delle *etichette precise*.

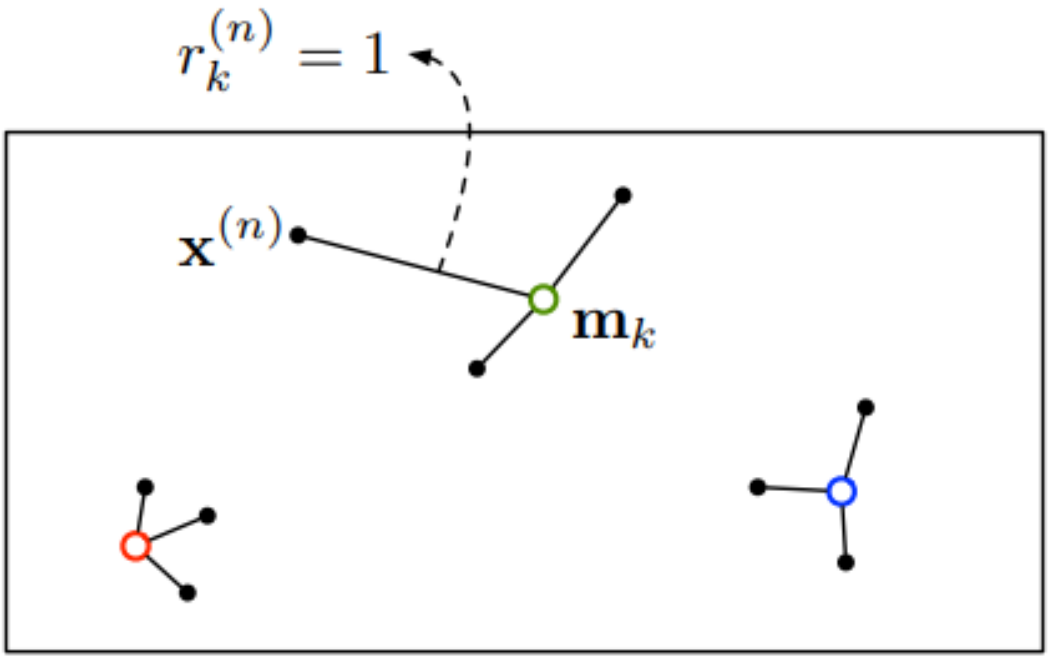
### 2.1 Problema di cluster

Ho un insieme di punti nello spazio euclideo che rappresentano le istanze, non ho i valori t ad esse associate, poichè abbiamo detto di non avere etichette.

Assumendo che ogni punto appartenga ad uno dei K cluster e che i punti dello stesso cluster siano simili (K iperparametro), il problema diventa *come identificare i cluster*. NB: Chi ci dice chi è $K$? Anche questo è un problema che affronteremo dopo!

#### 2.1.1 Algoritmo K means

Idea: bisogna trovare i centri dei cluster (ad ogni cluster corrisponde un punto $m_k$ che lo caratterizza) e poi dire quale punti di partenza $x^{(n)}$ associo a quale cluster.




Quindi l'obiettivo è:

Trovare il centro del cluster (non osservati, da trovare) $\{m_k\}_{k=1}^K$ e assegnare distanze (non osservate, da calcolare) $\{r^{(n)}\}_{n=1}^N$ per minimizzare la somma del quadrato delle distanze dei punti $\{x^{(n)}\}_{n=1}^N$ (osservati) rispetto ai loro centri assegnati.

Visto come problema di **ottimizzazione mista non lineare** questo può essere formulato come segue.

$$\min_{\{m_k\},\{r^{(n)}\}} J(\{m_k\},\{r^{(n)}\}) = \sum_{n=1}^{N}\sum_{k=1}^{K} r_k^{(n)} \|m_k - x^{(n)}\|^2$$

è un problema complesso, NP-hard. Le variabili sono "dove sono i vari centri" e la matrice $r^{(n)}$ ci dice per ogni punto a quale cluster lo assegno. ##### Esempio: $r^{(n)} = [0, ..., 1, ..., 0]^T$, Se in posizione $k$ abbiamo 1, allora il punto n-esimo è associato al cluster $k$, gli altri sono 0.

Il problema precedentemente esposto si risolve tramite una tecnica approssimata, la quale trasforma il problema in una coppia di problemi più semplice. Introduciamo il problema di ottimizzazione come *somma dei quadrati delle distanze tra un punto e il suo cluster assegnato*:

$$\min_{\{m_k\},\{r^{(n)}\}} \sum_{n=1}^{N}\sum_{k=1}^{K} r_k^{(n)} \|m_k - x^{(n)}\|^2$$

**Esempio** Prendiamo un certo sample $x^{(n)}$ assegnato al cluster $k = 3$, in termini di appartenza abbiamo:

$$r^{(n)} = [0, 0, 1, 0, ...]$$

La sommatoria interna, ovvero la distanza tra i punti di un cluster rispetto al loro cluster è:

$$\sum_{k=1}^{K} r_k^{(n)} \|m_k - x^{(n)}\|^2 = \|m_3 - x^{(n)}\|^2$$

### 2.1.2 Limitazione e Alternating Minimization

Ottimizzare **insieme** le attività di ricerca dei centri ed assegnazione dei punti a tali centri è complesso. Quello che facciamo è fissare una delle due variabili e risolvere rispetto all'altra, non otteniamo la soluzione esatta ma semplifichiamo il problema. Nel problema bisogna scegliere contemporaneamente dove mettere i *centroidi* e quali punti assegnare ai vari centroidi. Al variare dei centroidi cambia l'assegnamento, la difficoltà sta nel fatto di fare le cose congiuntamente. Introduciamo l'algoritmo di **Alternating Minimization**.

**Parte #1: Fissiamo i centri, troviamo le distanze** All'inizio, l'algoritmo sceglie punti *random* come centroidi, dati i vari punti questi vengono *assegnati* al cluster il cui centro è il più vicino.

$$r_k^{(n)} = \begin{cases} 1 & \text{if } k = \arg\min_j \|m_j - x^{(n)}\|^2 \\ 0 & \text{otherwise} \end{cases}$$

Idealmente si ottengono cluster che non sono necessariamente quelli giusti. Questa operazione è semplice ma costosa (non esponenziale, ma $O(n \cdot k)$, $n$ numero dei punti, $k$ numero di cluster).

**Parte #2: Fissiamo le distanze, troviamo i centri** Il secondo passo è quello in cui faccio l'operazione inversa, fisso gli assegnamenti $r$ e vado a minimizzare i centri ottimi (devo calcolare $\{m_k\}$). Per trovarli, calcolo la derivata della funzione delle distanze dei centroidi rispetto ai punti ad essi assegnati. Il risultato vieste posto a 0 comportando lo spostamento dei centroidi come conseguenza.

$$\frac{\partial}{\partial m_l} \sum_{n=1}^{N} \sum_{k=1}^{K} r_k^{(n)} \|m_k - x^{(n)}\|^2 = 0$$

$$\frac{\partial}{\partial m_l} \sum_{n=1}^{N} r_l^{(n)} \|m_l - x^{(n)}\|^2 = 0$$

$$2 \sum_{n=1}^{N} r_l^{(n)} (m_l - x^{(n)}) = 0 \Rightarrow m_l = \frac{\sum_n r_l^{(n)} x^{(n)}}{\sum_n r_l^{(n)}}$$

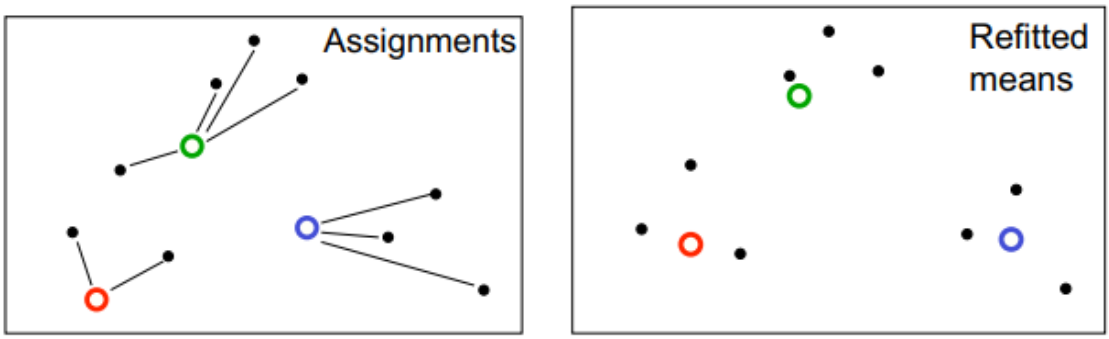
A questo punto itero ripartendo dai nuovi centri calcolati (finché non converge).

Ricapitolando, ciò che facciamo è:

1. Inizializzo i centri dei cluster (a caso) e assegno i punti al centroide più vicino (Assignment step):
   - $k^{(n)} = \arg\min_k \|m_k - x^{(n)}\|^2$
   - $r_k^{(n)} = \begin{cases} 1 & \text{if } k = \widehat{k}^{(n)} \\ 0 & \text{otherwise} \end{cases}$ for $k = 1, \dots, K$

2. ricalcolo la posizione dei cluster mettendoli al centro dei punti ad essi assegnati (*Refitting step*).

$$m_l = \frac{\sum_n r_l^{(n)} x^{(n)}}{\sum_n r_l^{(n)}}$$




Questo algoritmo è deterministico tranne che per la scelta randomica della posizione iniziale dei centroidi. Lo eseguo allora molte volte e mi scelgo la migliore.

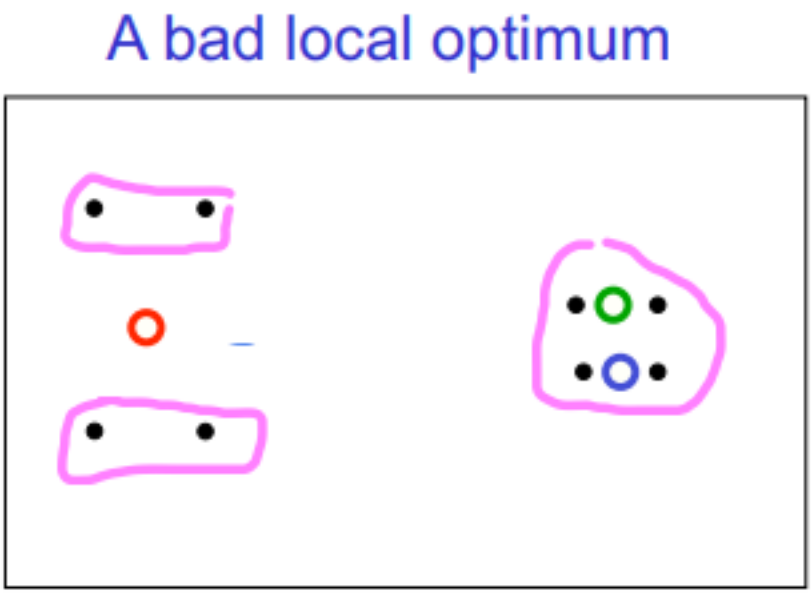
### 2.1.3 Note

Il centroide $\{m(k)\}$ coincide col **baricentro** perché andiamo a minimizzare il quadrato (dipende dalla funzione obiettivo scelta). Cambiando funzione obiettivo cambia totalmente l'algoritmo (posso scegliere una funzinoe obiettivo tale da volere che i punti siano a distanza massima dai centri dei cluster), l'importante è scegliere la funzione obiettivo che risolve il mio problema.

L'algoritmo converge sempre in un numero finito di passi.

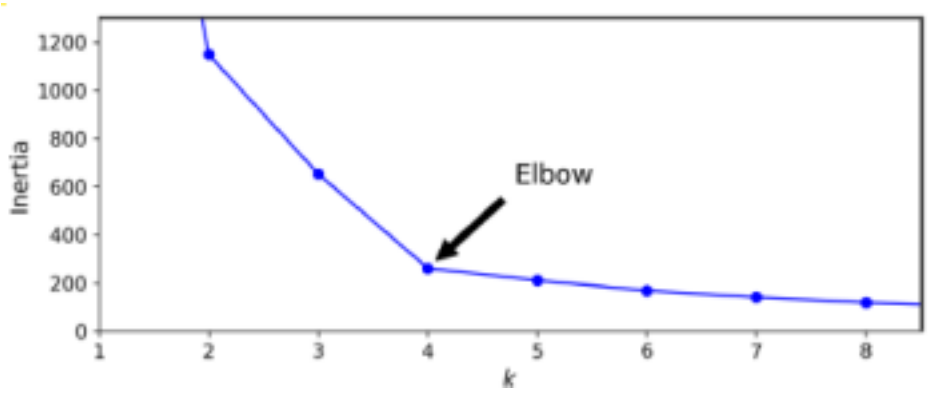
L'*algoritmo di k-means* riduce il costo della funzione obiettivo ad ogni passo, ad un certo punto non migliora più. Quando l'assegnamento dei punti non cambia l'algoritmo è arrivato a convergenza.

Essendo la funzione obiettivo non convessa, il minimo che trovo potrebbe essere un minimo locale (non globale). Nell'immagine a seguire, l'algoritmo trova i centri *rosso, verde, blu*, tuttavia una clusterizzazione *rosa* migliorebbe i risultati!




Dato che costa poco, lo eseguo tante volte e vedo in base alla funzione costo quale soluzione è migliore.

**Quale K scegliere?** Quando non sono convinto di quale sia il $k$ giusto, lo eseguo per valori diversi di $k$ e guardo l'errore, il metodo elbow method ci dice che ci fermiamo quando l'errore inizia a diminuire di poco (dove si forma il 'gomito').




Spesso le coordinate che uso per rappresentare i punti, fanno in modo che punti che non stanno vicini in un determinato spazio siano vicini in un altro spazio (con altre coordinate, o unità di misura. Passo da coordinate cartesiani a coordinate polari, oppure cambio unità di misura degli assi). Non lo vedremo quest'anno. Nella figura: a destra abbiamo due anelli concentrici, che potremmo mettere in un unico cluster. Cambiando coordinate (a sinistra), abbiamo evidenza dell'utilizzo di due cluster diversi!