**סילואט - Silhouette**

מדד סילואט (Silhouette) הוא מדד ב[תורת האשכולות](https://he.wikipedia.org/wiki/%D7%A0%D7%99%D7%AA%D7%95%D7%97_%D7%90%D7%A9%D7%9B%D7%95%D7%9C%D7%95%D7%AA" \o "ניתוח אשכולות) המאפשר מדידה עד כמה חלוקת נתונים ממאגר מידע לקבוצות בעלות מאפיינים דומים שביצענו נכונה ומתאימה.  
במקרה זה בניית מדד לטיב האשכול שבוצע.

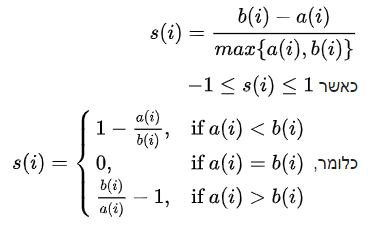
**סילואט** היא שיטה שתוארה לראשונה על ידי פיטר ג'יי ריסיו בשנת 1986[[1]](https://he.wikipedia.org/wiki/%D7%A1%D7%99%D7%9C%D7%95%D7%90%D7%98_(%D7%90%D7%A9%D7%9B%D7%95%D7%9C%D7%95%D7%AA)#cite_note-1) לטובת פירוש ואימות אחידות הנתונים באשכולות נתונים במאגרי מידע. השיטה מייצגת בצורה גרפית תמציתית עד כמה דומה נתון מסוים באשכול נתונים בהשוואה לכלל הנתונים באותו אשכול. ערך הסילואט, הנע בין (1-) ל-1, מציין עד כמה דומה נתון מסוים לנתונים האחרים באשכול אליו הוא משויך (לכידות) ועד כמה הוא שונה מהנתונים באשכולות הסמוכים (הפרדה).  
ערכי סילואט גבוהים מעידים על בנייה טובה של המודל, בעוד שערכים שליליים מעידים על בניה לקויה או  
על חלוקה שגויה לאשכולות.

**הגדרה:**  
בהינתן שקיים מאגר מידע שחולק ל-K אשכולות בכל שיטה שהיא, מוגדרים לכל נתון – i במאגר זה שני פרמטרים:

1. a(i)- פרמטר המגדיר עד כמה מאפייני הנתון דומים למאפייני הנתונים האחרים באותו אשכול.  
 a(i) הוא חישוב ממוצע המרחקים בין הנתון לשאר הנתונים באשכול ונקרא מדד **הלכידות** (בין הנתון   
 לחבריו באותו אשכול.  
 ככל שערך a(i)  {\displaystyle a(i)}a(i)sgקטן יותר כך הנתון תואם יותר לאשכול בו הוא נמצא.

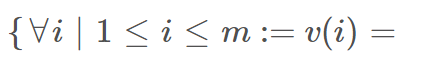
2. b(i) הוא חישוב ממוצע המרחקים בין הנתון לנתונים באשכול ה"שכן" כלומר האשכול בעל ממוצע   
 המרחקים המינימלי מהנתון, ונקרא מדד **ההפרדה** בין הנתון לנתונים באשכולות אחרים.  
 ככל שערך {\displaystyle b(i)}b(i) גדול יותר כך הנתון שונה יותר משאר האשכולות.

s(i) – הוא הערך אשר מצביע עד כמה הנתון הנבדק מתאים לאשכול בו הוא נמצא. ערך זה הוא  
 **ערך סילואט** ומחושב באופן הבא:



**משמעות ערך סילואט היא:**  
1.כאשר s(i) קרוב ל-1 ניתן לומר כי הנתון מתאים לאשכול בו הוא נמצא. ערך שכזה מתקבל כאשר ערך הלכידות  
 קטן בצורה משמעותית מערך ההפרדה.  
2.כאשר s(i) קרוב ל-0 ניתן לומר כי הנתון נמצא קרוב מאוד לגבול בין שני אשכולות שכנים.  
3.כאשר s(i) קרוב ל- (-1) ניתן לומר כי הנתון נמצא באשכול שלא מתאים

**How do the Silhouette algorithm work?**

Let there be a data set ‘data’ with m data points , each point is from the n’th dimension.  
Let us define for each data point i, a function V(i) which returns the pre given cluster affiliation as so:  
  
 C:\Users\tmp\AppData\Local\Microsoft\Windows\INetCache\Content.Word\m111.png



Let us define dk:



Let us define a function on dk:

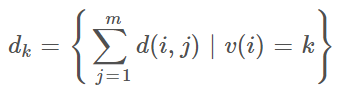


1. For each data point i:



1.1For each centroid k :

At this point we will calculate the average distance from point i to all K clusters.  
Meaning for each cluster 1<=k<=K, we will calculate the sum of distances from point i to all points from cluster number k. Now the received sum will be divided by the number of point cluster k own. The result is mean distance from point i to cluster k.



Now dk represents all the distances from point I to cluster k.

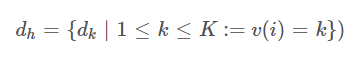
* 1. Now we will find the ‘Neighboring’ cluster, which means the closest cluster to point i.

It is essentially the cluster with the minimal average distance from point i.  
as long as that cluster isn’t i’s affiliation.  
Let us call that cluster dw.

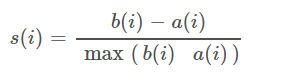
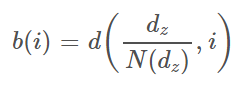
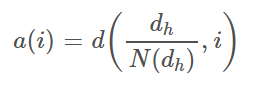


…

1.3 Let us define dh as the distance from point I to its own cluster.  
 This distance is calculated similarly as described above.



1.4 Now we have sufficient data to calculate the silhouette’s main parameters a(i),b(i),s(i).  
 Let us remember the function N(dk) which represents the number of points affiliated to   
 cluster k.



1.5 Now we have the silhouette value for point number i.  
 if the value is close to 1, the point is well grounded in its affiliated cluster.  
 If the value is close to 0, the point is close to a border between two neighboring clusters.  
 If the value is close to -1, the point is very far from it’s own affiliated cluster.

1.6 Repeat the same process for all data points.  
end

**K-MEDOIDS**

אלגוריתם K-medoids בדומה ל K-Means , גם הוא אלגוריתם אשכול. שני האלגוריתמים הנ"ל עוסקים בחלוקת אוסף n נקודות נתונים ל-k קבוצות זרות. בשונה מ-K-Means , נקודות המידע משמשות כמרכזי (Medoids) האשכולים. אופן חישוב המרחק עשוי להשתנות אך שמירה על עקביות נדרשת.  
אלגוריתם זה הינו רגיש יותר לרעש ולמחריגים.  
ניתן להגדירMedoid כאיבר המוכל באשכול נתון, שהמרחק הממוצע מאיבר זה לשאר איברי האשכול הוא מינימלי. משמע הוא האיבר הממורכז ביותר באשכול.

האלגוריתם:  
1.עבור אוסף data בעל n נקודות בחר k נקודות כמדואידים התחלתיים.  
 2.שייך כל נקודה בdata למדואיד הקרוב ביותר אליה לפי מרחק אוקלידי.  
3.כל עוד מספר האיטרציות קטן מהמקסימום שהוגדר ע"י המשמש וגם פונקציית המחיר גדולה מ-0:  
 3.1 לכל מדואיד m=1:k  
 3.1.1 מצא את הנקודה בעלת המרחק הממוצע המינימלי מכל שאר הנקודות באשכול הנוכחי.

3.1.2 אם הנקודה שנבחרה שונה מהמדואיד m סמן 1 במערך דגלים באינדקס המתאים לm.

3.2 אם מערך הדגלים הינו 0, כלומר אף נקודה לא נמצאה ממורכזת יותר מהמדואיד באותה נקודת זמן, סיים את הלולאה והחזר למשתמש את התוצאות. אחרת המשך.

1. For data set ‘data’ with n data points, randomly choose k data points as initial medoids  
2.Affiliate each data point to it’s closest medoids, using Euclidean distance.  
3.While iteration number is smaller than defined AND there is no medoids change:  
 3.1 For each medoid m=1:k  
 3.1.1 Find the point p with minimal distance from all other points in the same   
 cluster m.  
 3.1.2 If the point p is the same point as medoid m note there was a change in  
 medoid m.  
 3.2 If none of the medoids had been changed, break loop and return data to user.