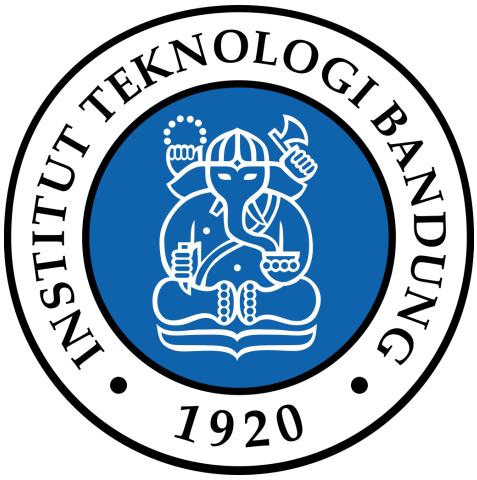
TUGAS BESAR 2

IF3270 - PEMBELAJARAN MESIN

IMPLEMENTASI *CLUSTERING*



Dibuat oleh :

Irfan Haris W.	13517041
Marsa Thoriq Ahmada	13517071
Mgs. M. Riandi Ramadhan	13517080
Fajar Muslim	13517149

TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2020

A. PENJELASAN IMPLEMENTASI

1. K-Means Clustering

Berikut ini adalah langkah dari implementasi algoritma K-Means Clustering.

a. Menginisiasi centroid sesuai dengan banyaknya *cluster*.

Memanfaatkan **rangeFeature** agar tahu rentang nilai dari satu fitur setelah itu digunakan untuk fungsi **setRange** untuk digabungkan rentang nilai tersebut untuk semua fitur. Dengan informasi yang didapat dari fungsi sebelumnya digunakan untuk menginisiasi centroid secara random dengan fungsi **initCentroid .**

b. Mengecek jarak dari setiap centroid yang ada.

Setiap *instance* dihitung jaraknya satu persatu dengan centroid menggunakan fungsi **checkDistCentroid.** Untuk perhitungan jarak yang digunakan menggunakan *euclidean distance* yang dihitung pada fungsi **distFromCentroid .**

c. Melakukan *clustering* dengan informasi jarak suatu *instance* dengan centroid yang ada.

Untuk memasukan suatu *instance* ke *cluster* tertentu perlu dilakukan pengecekan *cluster* mana yang terdekat dengan *instance* tersebut dengan menggunakan fungsi **checkCluster.** Setiap hasil *clustering instance* dari fungsi **checkCluster** digabungkan menggunakan fungsi **clustering** agar didapat hasil *clustering* dari semua *instance* pada *dataset*.

d. Jika hasil *cluster* berbeda dengan sebelumnya maka pindahkan centroid ke centroid yang baru lalu ulangi langkah 2.

Dengan fungsi **moveCentroid,** centroid yang lama akan dipindahkan ke centroid yang baru berdasarkan *cluster* yang baru dengan mencari posisi rata-rata dari *cluster* baru tersebut.

e. Jika hasil *cluster* sama dengan sebelumnya maka algoritma selesai.

2. Agglomerative Clustering

Berikut ini adalah langkah implementasi algoritma Agglomerative Clustering.

a. Menginisiasi matriks jarak antar *cluster*.

Matriks jarak antar *cluster* atau *proximity matrix* menyatakan jarak antar *cluster* yang dinyatakan dalam bentuk *euclidean distance*. Awal mulanya matriks 2 dimensi ini diisi dengan nilai 0 sesuai dengan banyak *instance* karena pada awal pembentukan *cluster*, setiap *instance* dinyatakan sebagai *cluster* yang berbeda satu sama lain. Kemudian untuk setiap jarak *cluster* dengan *cluster* lainnya, dihitung *euclidean distance* berdasarkan data yang diberikan. Inisiasi matriks jarak ini menggunakan fungsi **init_proximity_matrix** dan jarak antar *cluster* dihitung menggunakan fungsi **euclidean_dist**.

b. Menginisiasi daftar *cluster*.

Daftar *cluster* diinisiasi dalam bentuk *array of string*. Setiap string menyatakan satu *cluster*. Pada inisiasi awal, *string* ini merupakan representasi angka baris dari data yang diberikan yang telah diubah dalam bentuk *string*. Inisiasi daftar *cluster* menggunakan fungsi **init_cluster**.

c. Mencari pasangan *cluster* dengan jarak terdekat.

Jika kondisi terminasi belum terpenuhi yaitu banyak *cluster* akhir yang diekspektasikan tercapai, maka dilakukan pencarian pasangan *cluster* dengan jarak terdekat. Pasangan *cluster* dengan jarak terdekat ditemukan setelah melakukan pencarian di *proximity matrix* (matriks jarak antar *cluster*). Pasangan *cluster* dengan jarak terdekat akan di-*merge* pada fungsi yang akan dipanggil selanjutnya. Pencarian ini dilakukan dengan menggunakan fungsi **find_shortest_dist** dan akan mengembalikan pasangan cluster dengan jarak terdekat.

d. Memperbarui daftar *cluster*.

Jika telah ditemukan pasangan *cluster* dengan jarak terdekat, maka dilakukan pembaruan daftar *cluster*. Kedua cluster tersebut akan di-*merge* menjadi satu cluster dan direpresentasikan dalam bentuk *string* yang dipisahkan oleh karakter ‘,’. Pembaruan daftar *cluster* ini menggunakan fungsi **update_clusters**.

e. Memperbarui matriks jarak antar *cluster*

Setelah memperbarui daftar *cluster*, dilakukan pembaruan pada *proximity matrix* (matriks jarak antar *cluster*). Kedua *cluster* yang telah ditemukan sebelumnya akan digabung nilai jaraknya dan mempengaruhi jarak *cluster* yang telah di-*merge* dengan *cluster* lainnya. Pembaruan jarak ini dilakukan dengan menggunakan fungsi **update_proximity_matrix**. Penentuan jarak yang diperbarui disesuaikan dengan pendekatan yang diterapkan. Berikut ini adalah 4 pendekatan yang dapat digunakan pada *Agglomerative Clustering*.

- *Single linkage*

  Pendekatan ini menyatakan bahwa ketidaksamaan antara dua *cluster* merupakan ketidaksamaan minimum antara anggota dari dua *cluster* sehingga pembaruan jarak yang diambil merupakan nilai jarak terkecil dalam anggota *cluster* yang di-*merge*.

- *Complete linkage*

  Pendekatan ini menyatakan bahwa ketidaksamaan antara dua *cluster* merupakan ketidaksamaan maksimum antara anggota dari dua *cluster* sehingga pembaruan jarak yang diambil merupakan nilai jarak terbesar dalam anggota *cluster* yang di-*merge*.

- *Average linkage*

Pendekatan ini menyatakan bahwa ketidaksamaan antara dua *cluster* merupakan jarak rata-rata dari setiap anggota *cluster* dengan *cluster* lainnya sehingga pembaruan jarak yang diambil merupakan nilai rata-rata jarak dalam anggota *cluster* yang di-*merge*.

- *Average group linkage*

  Pendekatan ini menyatakan bahwa ketidaksamaan antara dua *cluster* merupakan jarak rata-rata antara semua anggota dalam setiap *cluster* yang dibandingkan sehingga pembaruan jarak yang diambil ini perlu menggunakan fungsi tambahan yaitu **cluster_means_distance**.

3. Evaluasi

Untuk implementasi evaluasi yang dilakukan menggunakan library sklearn.metrics dengan mengimport fowlkes_mallows_score dan silhouette_score untuk penjelasannya sbb.

a. Fowlkes Mallows

Fowlkes mallows melakukan evaluasi terhadap hasil algoritma klasterisasi kita terhadap hasil target data iris yang sebenarnya(fakta) menggunakan rumus berikut :

$$B_k = \frac{TP}{\sqrt{(TP+FP)(TP+FN)}}$$

TP = jumlah data True Positive

FP = jumlah data False Positive

FN = jumlah data False Negative

Semakin besar nilai indeks fowlkes-mallows yang kita dapatkan maka semakin baik hasil kerja algoritma klasterisasi kita.

b. Silhouette Coefficient

Silhouette coefficient melakukan evaluasi terhadap hasil algoritma klusterisasi menggunakan rumus berikut :

$$\text{Silhouette Coefficient} = (x-y)/ \max(x,y)$$

Dimana x adalah mean distance ke semua instance di cluster yang sama dan y adalah mean distance ke semua instance di cluster tetangga yang terdekat.

Nilai silhouette coefficient berada dalam rentang -1 sampai 1 dimana semakin mendekati 1 maka semakin baik hasil kerja algoritma klasterisasi kita.

4. Visualisasi

Dalam membuat visualisasi kami menggunakan library matplotlib dan seaborn. Visualisasi yang kami buat dalam bentuk 2D, dengan sumbu x dan sumbu y merupakan fitur dari sebuah dataset. Sebagai contoh, pada dataset iris terdapat 4 fitur dan 1 buah target. Untuk setiap pasangan fitur yang, divisualisasikan sebuah grafik dengan sumbu x dan y merupakan fitur yang bersangkutan dan target direpresentasikan dengan warna yang berbeda untuk setiap instance di grafik tersebut.

B. HASIL EKSEKUSI & ANALISIS

1. K-Means Clustering

a. Evaluasi

Evaluasi

Fowlkes_Mallows

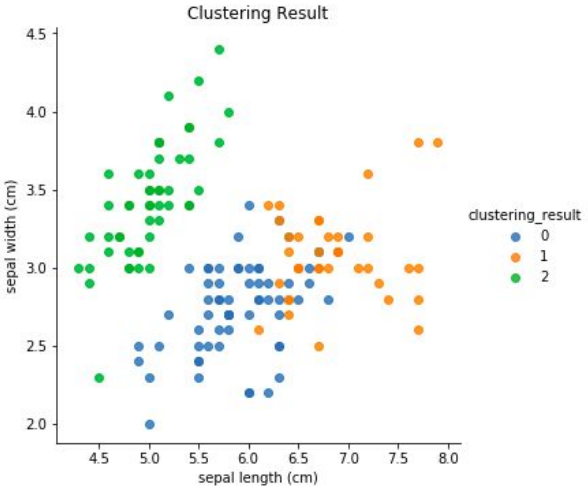
```
In [26]: fowlkes_mallows_score(ans, iris_y)
Out[26]: 0.8112427991975698
```

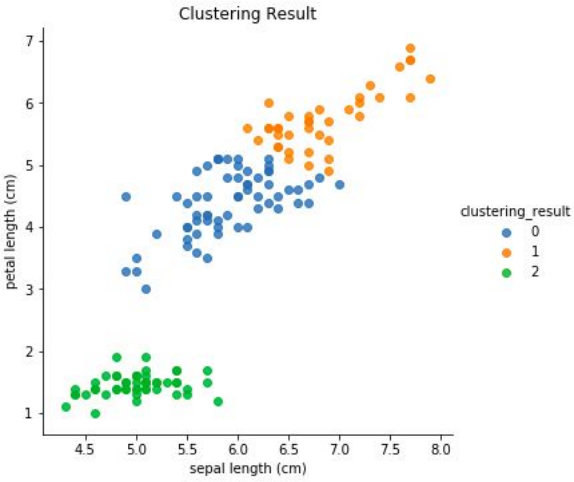
Silhouette Coefficient

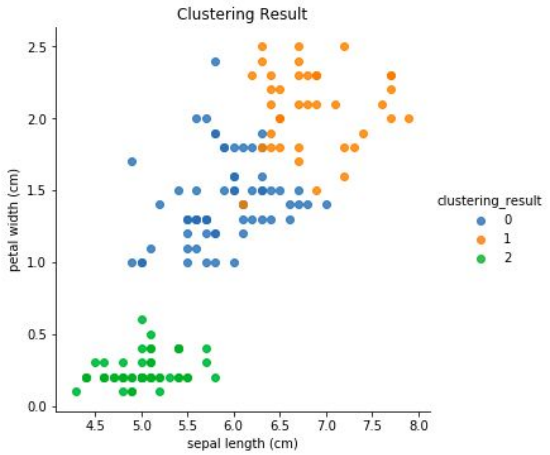
```
In [27]: silhouette_score(iris_data, ans)
Out[27]: 0.5511916046195915
```

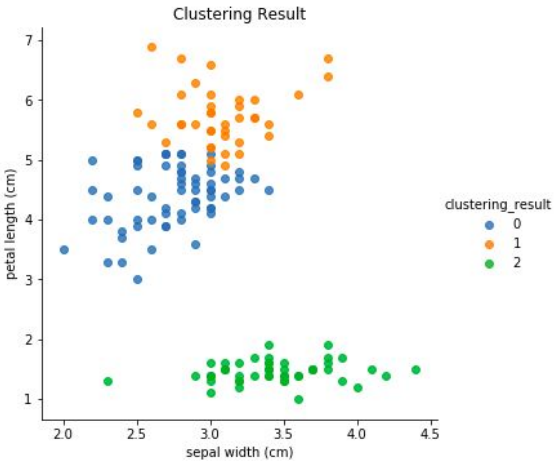
Berdasarkan hasil evaluasi dengan pendekatan Fowlkes-Mallows, dapat diketahui bahwa hasil pengujian memiliki tingkat kemiripan yang tinggi dengan label yang diberikan sebelumnya karena mendekati 1. Berdasarkan hasil evaluasi dengan pendekatan Silhouette Coefficient, dapat diketahui bahwa pengujian ini relasi antara objek dalam klaster cukup dekat dan jarak antar klaster cukup jauh karena mendekati 1.

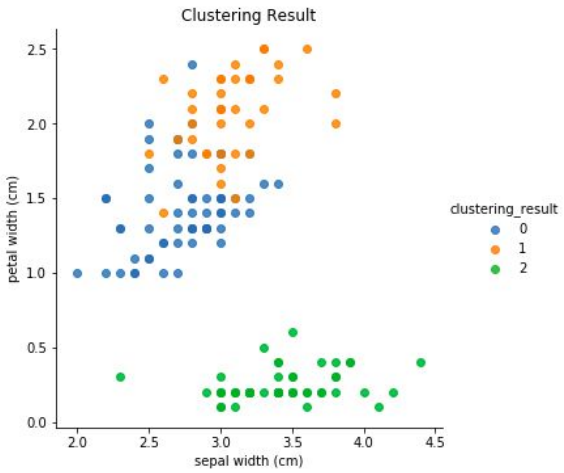
b. Visualisasi

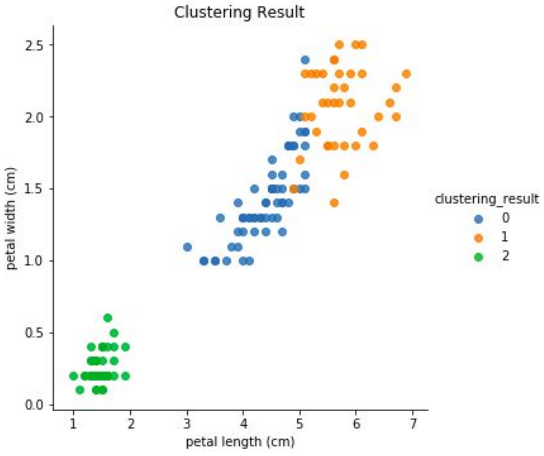












Berdasarkan hasil visualisasi di atas terhadap berbagai kombinasi fitur, dapat diketahui bahwa algoritma K-Means yang telah diimplementasikan cukup baik untuk melakukan klasterisasi. Hal ini juga telah dibuktikan dengan evaluasi pendekatan Silhouette Coefficient sebesar 0.55 dan pendekatan Fowlkes-Mallows sebesar 0.81.

2. Agglomerative Clustering

- **Average**

a. Evaluasi

Fowlkes_Mallows

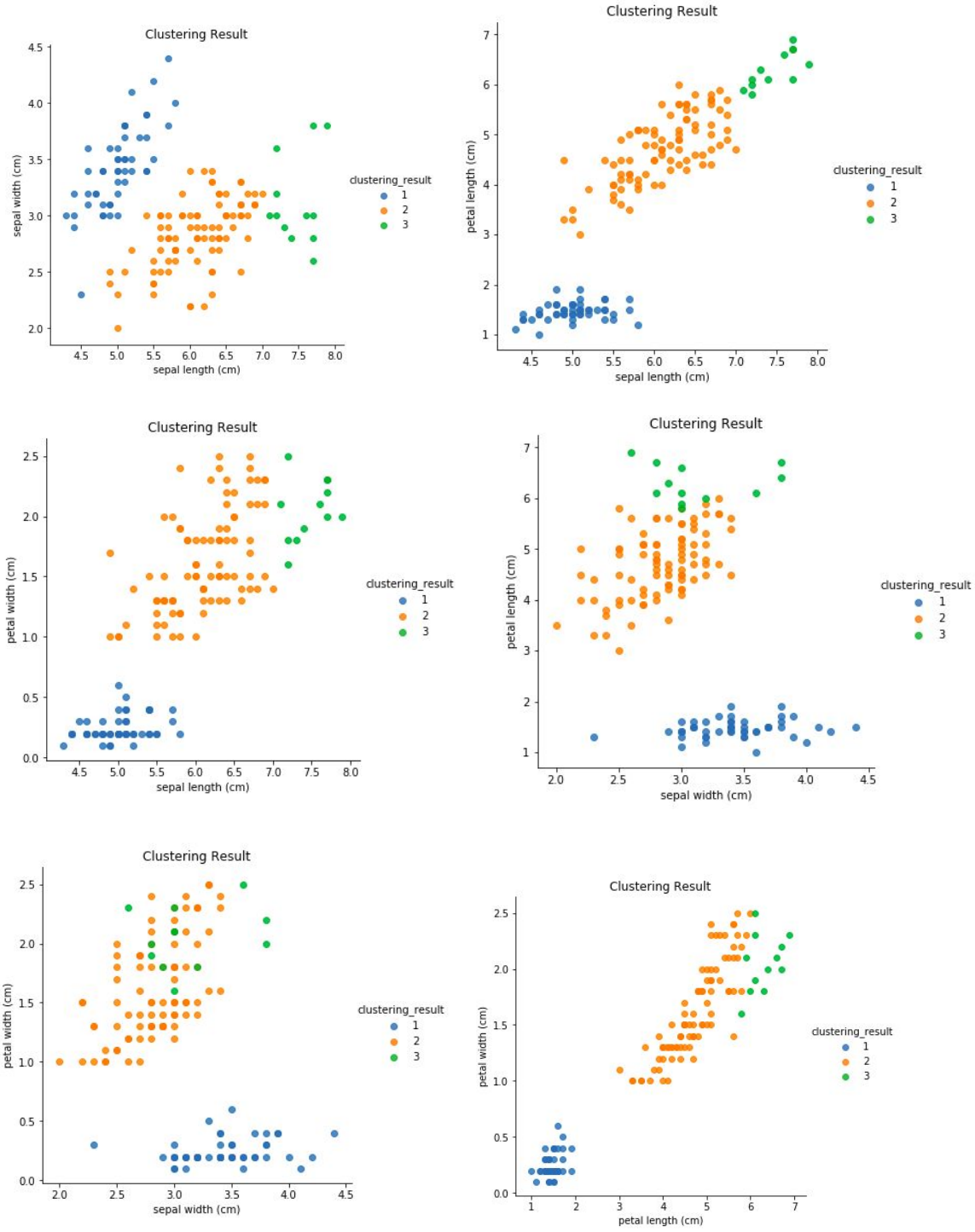
```
In [66]: fowlkes_mallows_score(arr_agglo_result_average, iris_y)
Out[66]: 0.7421637537684369
```

Silhouette Coefficient

```
In [70]: silhouette_score(iris_data, arr_agglo_result_average)
Out[70]: 0.5340213263943496
```

Berdasarkan hasil evaluasi dengan pendekatan Fowlkes-Mallows, dapat diketahui bahwa hasil pengujian memiliki tingkat kemiripan yang tinggi dengan label yang diberikan sebelumnya karena mendekati 1. Berdasarkan hasil evaluasi dengan pendekatan Silhouette Coefficient, dapat diketahui bahwa pengujian ini menghasilkan relasi antara objek dalam klaster cukup dekat dan jarak antar klaster cukup jauh karena mendekati 1. Dalam average linkage, setiap subset klaster dapat memiliki kohesi yang berbeda sehingga tidak dapat dilakukan pra-komputasi terhadap semua kemungkinan klaster.

b. Visualisasi



Berdasarkan hasil visualisasi di atas terhadap berbagai kombinasi fitur, dapat diketahui bahwa algoritma Agglomerative dengan Average Linkage yang telah diimplementasikan cukup baik untuk melakukan

klasterisasi. Hal ini juga telah dibuktikan dengan evaluasi pendekatan Silhouette Coefficient sebesar 0.53 dan pendekatan Fowlkes-Mallows sebesar 0.74.

- **Single**
  - **a. Evaluasi**

Fowlkes_Mallows

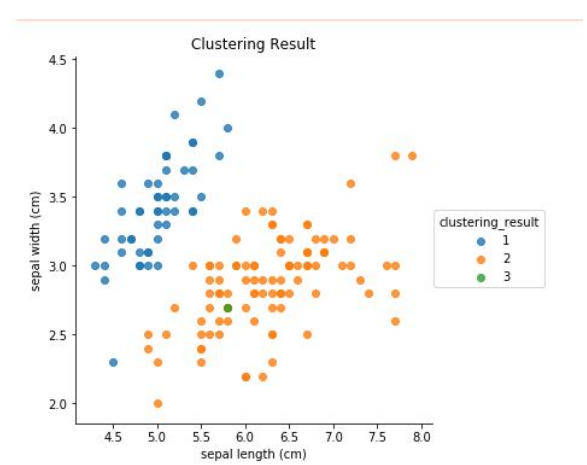
```
In [67]: fowlkes_mallows_score(arr_agglo_result_single, iris_y)
Out[67]: 0.7673442541005148
```

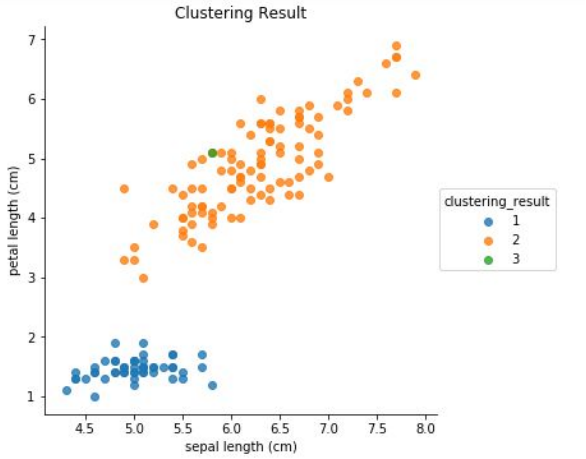
Silhouette Coefficient

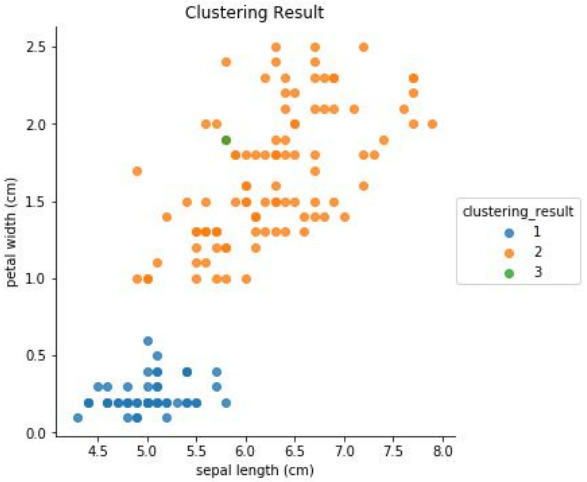
```
In [71]: silhouette_score(iris_data, arr_agglo_result_single)
Out[71]: 0.1327579936611085
```

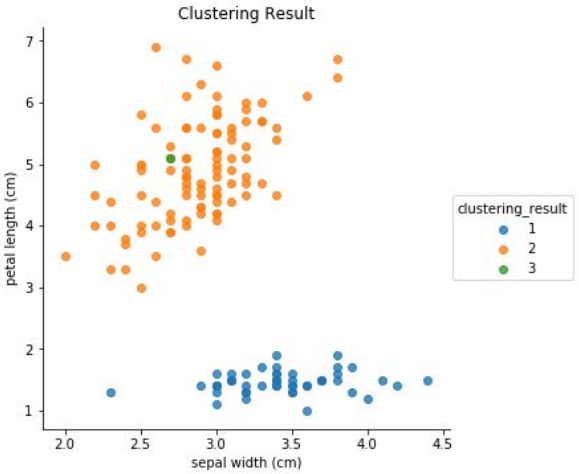
Berdasarkan hasil evaluasi dengan pendekatan Fowlkes-Mallows, dapat diketahui bahwa hasil pengujian memiliki tingkat kemiripan yang tinggi dengan label yang diberikan sebelumnya karena mendekati 1. Berdasarkan hasil evaluasi dengan pendekatan Silhouette Coefficient, dapat diketahui bahwa pengujian ini menghasilkan jarak antar klaster cukup dekat karena mendekati di antara -1 dan 1. Ini dikarenakan terdapat *chaining effect* sehingga penggunaan single linkage jarang menghasilkan solusi yang optimal.

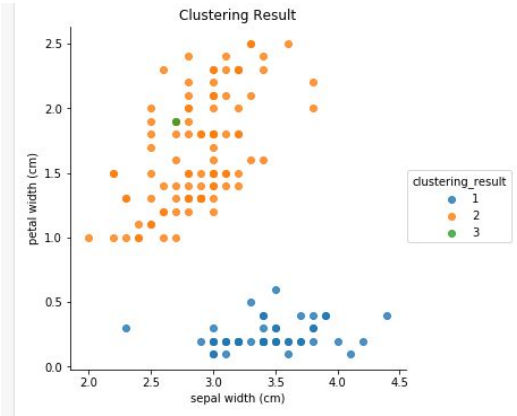
b. Visualisasi

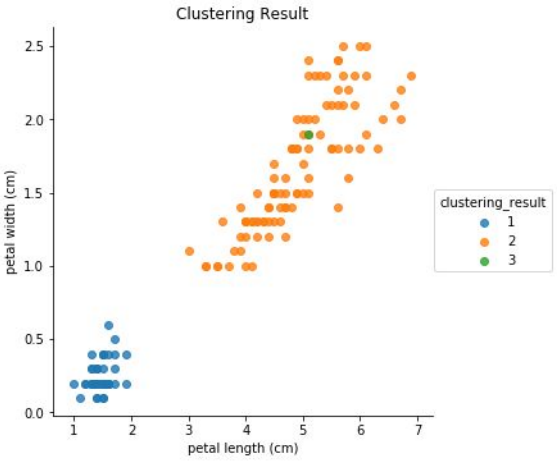












Berdasarkan hasil visualisasi di atas terhadap berbagai kombinasi fitur, dapat diketahui bahwa algoritma Agglomerative dengan Single

Linkage yang telah diimplementasikan tidak cukup baik untuk melakukan klasterisasi karena dapat dilihat terdapat 1 klaster yang berisi outlier dan terletak di dalam klaster lainnya yang lebih dominan. Hal ini juga telah dibuktikan dengan evaluasi pendekatan Silhouette Coefficient sebesar 0.13 dan pendekatan Fowlkes-Mallows sebesar 0.77.

- **Complete**
  a. **Evaluasi**

Fowlkes_Mallows

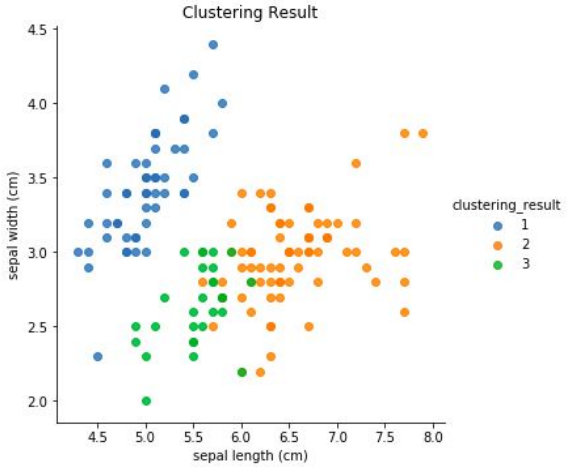
```
In [68]: fowlkes_mallows_score(arr_agglo_result_complete, iris_y)
Out[68]: 0.7686371028513819
```

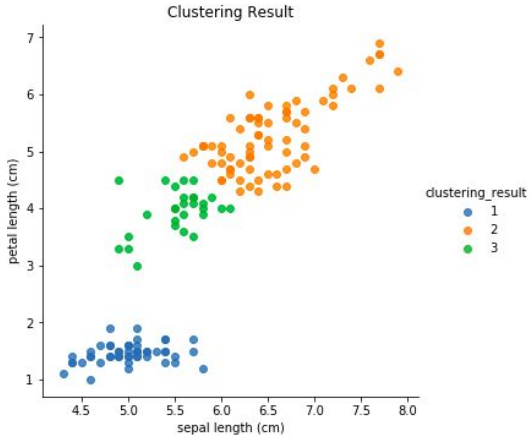
Silhouette Coefficient

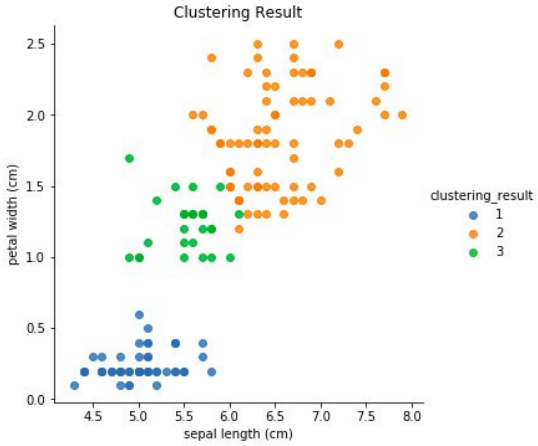
```
In [72]: silhouette_score(iris_data, arr_agglo_result_complete)
Out[72]: 0.5135953221192208
```

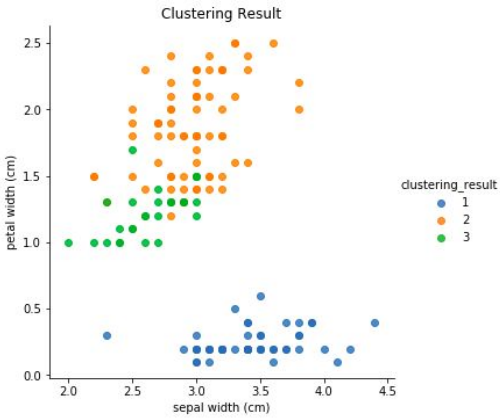
Berdasarkan hasil evaluasi dengan pendekatan Fowlkes-Mallows, dapat diketahui bahwa hasil pengujian memiliki tingkat kemiripan yang tinggi dengan label yang diberikan sebelumnya karena mendekati 1. Berdasarkan hasil evaluasi dengan pendekatan Silhouette Coefficient, dapat diketahui bahwa pengujian ini menghasilkan relasi antara objek dalam klaster cukup dekat dan jarak antar klaster cukup jauh karena mendekati 1. Dalam sudut pandang pragmatis, algoritma Agglomerative dengan complete linkage menghasilkan hierarki yang lebih baik dibandingkan single linkage karena dapat menghindari *chaining effect*.

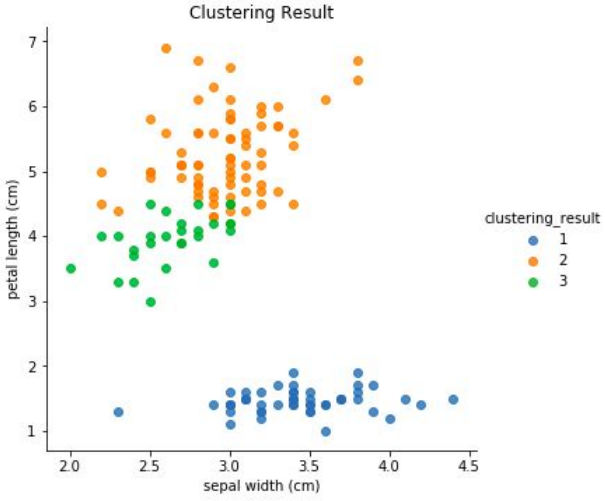
b. Visualisasi

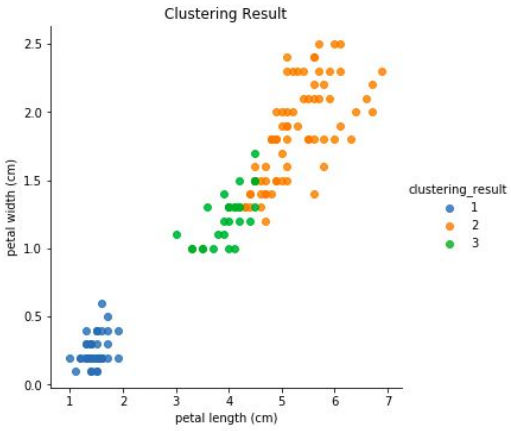












Berdasarkan hasil visualisasi di atas terhadap berbagai kombinasi fitur, dapat diketahui bahwa algoritma Agglomerative dengan Complete

Linkage yang telah diimplementasikan cukup baik untuk melakukan klasterisasi walaupun di beberapa kartesian kombinasi fitur terlihat bertabrakan antara klaster 2 dan 3. Hal ini juga telah dibuktikan dengan evaluasi pendekatan Silhouette Coefficient sebesar 0.51 dan pendekatan Fowlkes-Mallows sebesar 0.77.

- **Average Group**
  - **a. Evaluasi**

Fowlkes_Mallows

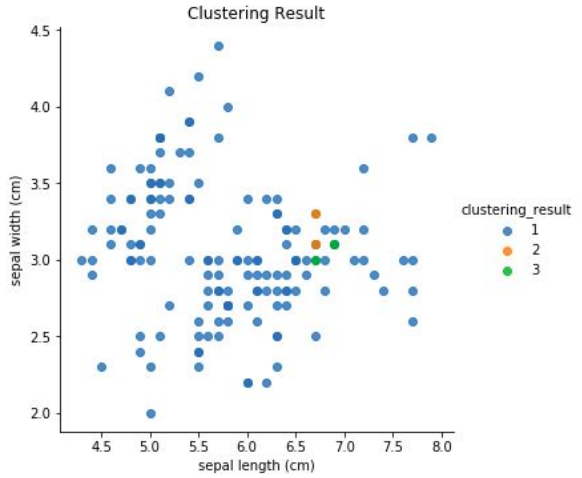
```
In [69]: fowlkes_mallows_score(arr_agglo_result_average_group, iris_y)
Out[69]: 0.5590321940207358
```

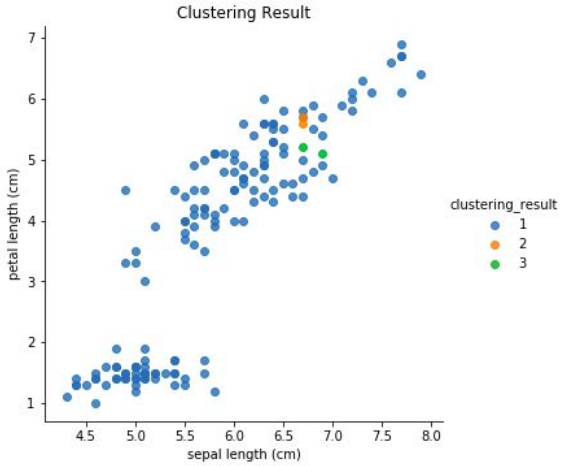
Silhouette Coefficient

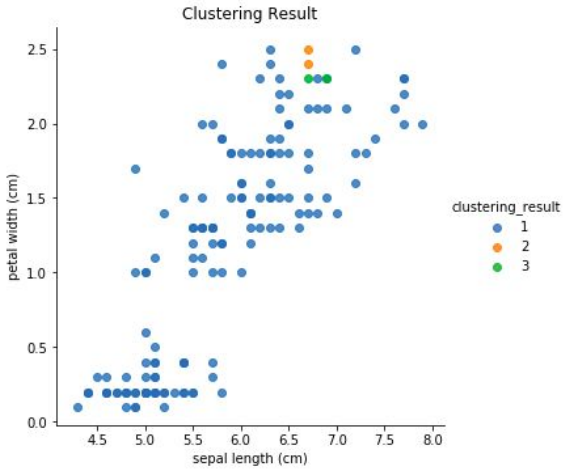
```
In [73]: silhouette_score(iris_data, arr_agglo_result_average_group)
Out[73]: -0.12306815802144243
```

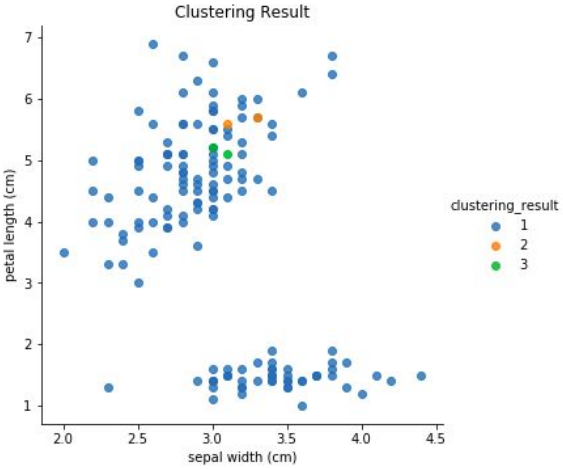
Berdasarkan hasil evaluasi dengan pendekatan Fowlkes-Mallows, dapat diketahui bahwa hasil pengujian memiliki tingkat kemiripan yang cukup tinggi dengan label yang diberikan sebelumnya karena mendekati 1. Berdasarkan hasil evaluasi dengan pendekatan Silhouette Coefficient, dapat diketahui bahwa pengujian ini menghasilkan jarak antar klaster cukup dekat karena mendekati -1. Ini menunjukkan bahwa tingkat kohesi dalam suatu subset dalam klaster cukup beragam atau tidak merata.

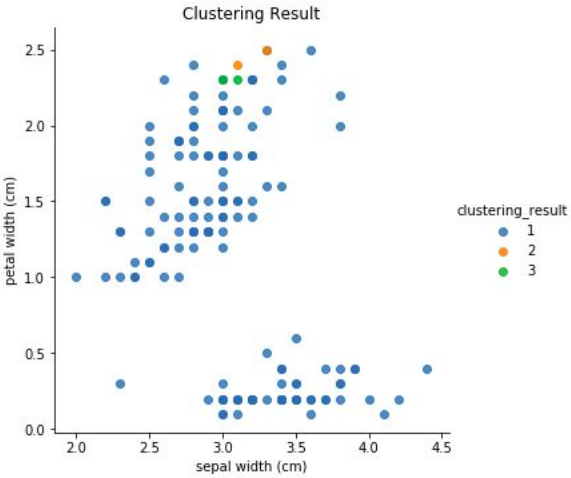
b. Visualisasi

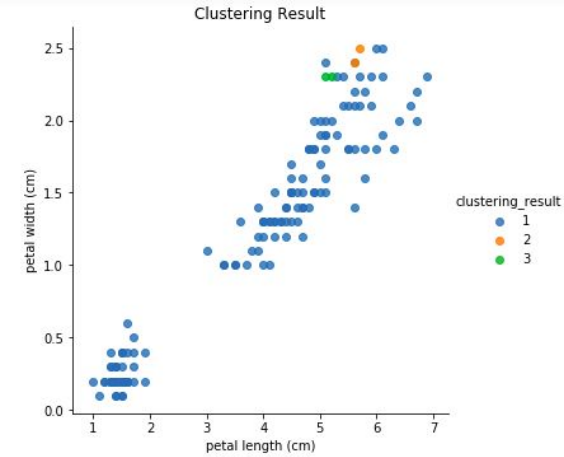












Berdasarkan hasil visualisasi di atas terhadap berbagai kombinasi fitur, dapat diketahui bahwa algoritma Agglomerative dengan Average-Group Linkage yang telah diimplementasikan tidak cukup baik untuk melakukan klasterisasi karena dapat dilihat terdapat 1 klaster dominan yang mengelilingi klaster-klaster lainnya. Hal ini juga telah dibuktikan dengan evaluasi pendekatan Silhouette Coefficient sebesar -0.12 dan pendekatan Fowlkes-Mallows sebesar 0.56.

C. ANALISIS PERBANDINGAN

Berikut ini adalah hasil perbandingan evaluasi dengan pendekatan Fowlkes-Mallows dan Silhouette Coefficient untuk setiap algoritma yang telah diimplementasikan.

No	Algoritma	Fowlkes Mallows	Silhouette Coefficient
1	K-Means	0.8112427991975698	0.5511916046195915
2	Agglomerative Single	0.7673442541005148	0.1327579936611082
3	Agglomerative Complete	0.7686371028513819	0.5135953221192208
4	Agglomerative Average	0.7421637537684369	0.5340213263943496
5	Agglomerative Average Group	0.5590321940207358	-0.123068158021441

Dari tabel analisis perbandingan diatas didapatkan algoritma yang mendapatkan nilai terbaik adalah algoritma K-Means berdasarkan evaluasi Fowlkes Mallows dan Silhouette Coefficient. Untuk bagian Agglomerative yang mendapatkan hasil paling baik berdasarkan evaluasi Fowlkes Mallows adalah Agglomerative menggunakan complete linkage sedangkan berdasarkan silhouette coefficient yang paling bagus adalah agglomerative average linkage.

Pada algoritma K-Means bisa mendapatkan hasil yang terbaik dikarenakan menggunakan partition based clustering jadi mendapatkan Silhouette Coefficient yang relatif lebih baik dibandingkan agglomerative yang menggunakan hierarchical based.

Dalam algoritma agglomerative complete linkage dapat diketahui relasi antara objek dalam klaster cukup dekat dan jarak antar klaster cukup jauh karena mendekati 1. Dalam sudut pandang pragmatis, algoritma Agglomerative dengan complete linkage menghasilkan hierarki yang lebih baik dibandingkan single linkage karena dapat menghindari *chaining effect*.

D. PEMBAGIAN TUGAS

Berikut ini adalah pembagian tugas yang diterapkan dalam menyelesaikan tugas besar ini.

NIM	Nama	Tugas	Kontribusi
13517041	Irfan Haris W.	Evaluasi Metrik, Update Matrix Agglomerative (Average dan Average Group Linkage)	25%
13517071	Marsa Thoriq Ahmada	K-Means Algorithm	25%
13517080	Mgs. M. Riandi Ramadhan	Agglomerative Algorithm, Update Matrix Agglomerative (Single dan Complete Linkage)	25%
13517149	Fajar Muslim	Visualisasi Clustering (Cartesian dan Dendogram)	25%