k-Means

Door Joost van Bussum en Niels Risseeuw

Uitleg van de code:

Call tree

screePlot
….kMeans
……..findNearestCentroid
……….calculatePlotDistance
….calculateIntraclusterDistance

totaal-uitleg

Onze code begint met de functie **screePlot** aanroepen In deze functie zal uiteindelijk de intra-cluster distance teruggegeven worden. Eerst wordt de functie **kMeans** aangeroepen.

In de **kMeans** functie gebeurd het clusteren van alle data, er worden willekeurige punten uit de dataset gekozen als centroids. Vervolgens worden alle dichtstbijzijnde centroids gevonden van alle punten uit de dataset. Deze worden in een lijst toegevoegd: **newPlotDistributionList**. In deze lijst staan alle centroids met hun bijbehorende punten uit de dataset. Vervolgens wordt er een nieuwe locatie berekent voor de centroids en begint het riedeltje weer opnieuw. Dit gaat door totdat de centroids niet meer van positie veranderen en er dus definitieve clusters zijn gevonden.

Dit wordt gedaan door voor een range aan centroids voor elke cluster de intra-cluster distance en te appenden aan een lijst, **intraClusterDistancCalculationsOutcome,** bestaande uit de hoeveelheid centroids met de intra-cluster distance volgens de volgende formule(bij ons de functie **calculateIntraclusterDistance,** vanaf hier wordt ook de **calculatePlotDistance** functie aangeroepen om de afstand tussen het punt en de centroid te berekenen):

$$V = \sum_{j}^{k} \sum_{x_i \to c_j}^{n} (c_j - x_i)^2.$$

Dit wordt een gegeven aantal keer gedaan (in ons geval 5 keer) en vanuit die lijst wordt de centroid met de kleinse afstand toegevoegd aan een andere lijst namelijk: **intraClusterDistancOutcome.** Deze lijst zal onze screeplot vormen.

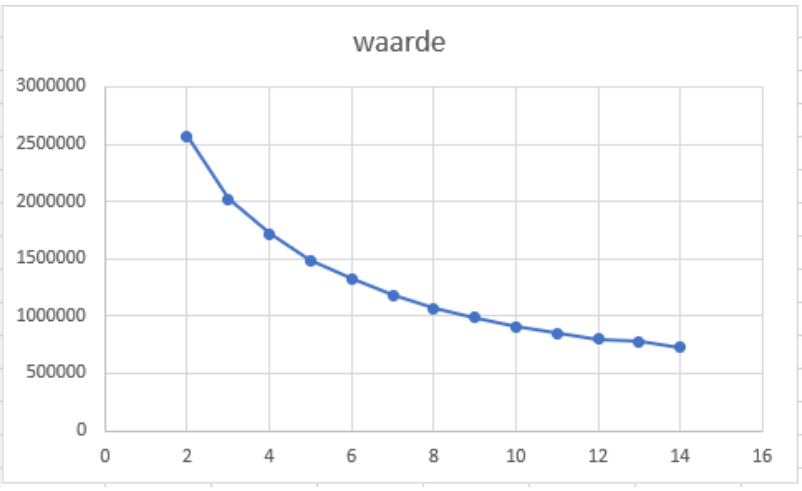
Resultaten

De resultaten van de **screePlot** functie zijn als volgt:

```
[Running] python -
u "c:¥Users¥Joost¥Documents¥C++ shizzle¥AppliedAI¥1.2¥opdracht1_2.py"
[[2, 2567473.0], [3, 2019506.0], [4, 1717421.0], [5, 1484931.0], [6, 1324
142.0], [7, 1181700.0], [8, 1064511.0], [9, 984952.0], [10, 904203.0], [1
1, 851899.0], [12, 799544.0], [13, 777828.0], [14, 726212.0]]
```

Geplot in een grafiek ziet dit er zo uit:

k	waarde	afgeleide
2	2567473	
3	2019506	-547967
4	1717421	-302085
5	1484931	-232490
6	1324142	-160789
7	1181700	-142442
8	1064511	-117189
9	984952	-79559
10	904203	-80749
11	851899	-52304
12	799544	-52355
13	777828	-21716
14	726212	-51616



Antwoord op de opdracht

Uit de data in het vorige kopje kunnen we concluderen dat de optimale k voor deze dataset 6 is. Dit omdat de steilheid van de afname hier een stuk minder wordt en meer plat gaat worden.