Wstęp

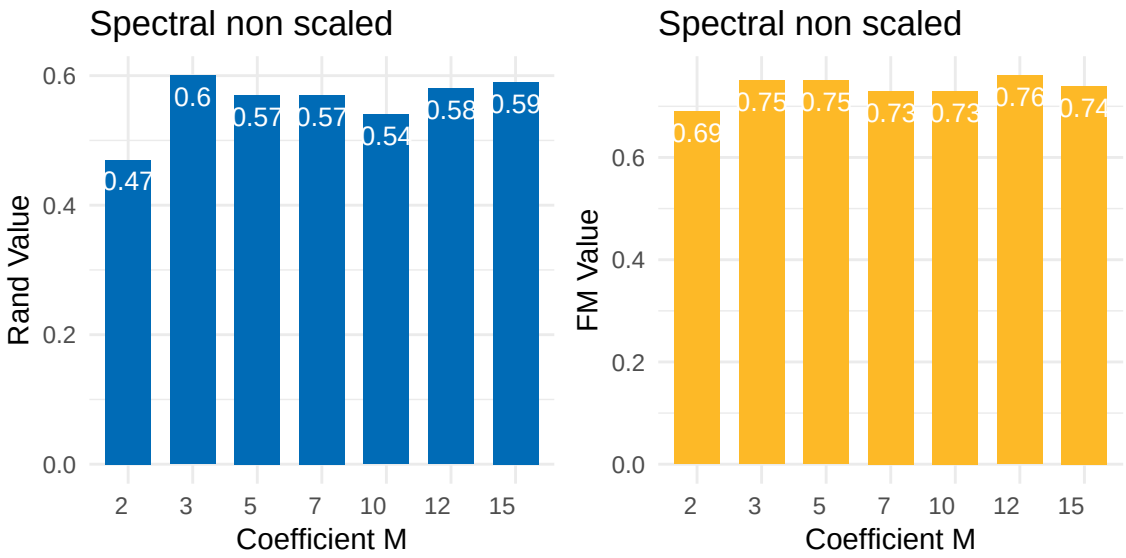
W analizie skupien ktora przeprowadziłem zdecydowałem sie na użycie zgodnie z poleceniem własnego algorytmu analizy skupień, wszystkich algorytmów z pakietu stats, algorytmu genie oraz znalezionego przeze mnie algorytmu cmeans z pakietu e1071 dostepnego na CRANie. O ostatnim z wymienionych algorytmow rozpisze sie troszeczke później gdy będę prezentował wyniki przez niego uzyskane, gdyż z pewnoscia zasada jego dzialania jest interesujaca.

Z racji na duża ilosc zbiorów(lacznie z dodanymi przeze mnie jest ich 46) uznalem iż znaczną część swojej analizy oprę o pokazywanie wyników średnich uzyskiwanych przez algorytmy. Zgodnie z wytyczynmi testy przeprowadziłem najpierw na zbiorach normalnych, a nastepnie na zbiorach ustandaryzowanych.
To po czym bedziemy decydowac czy algorytm jest dobry badz zly będą etykiety referencyjne, przygotowane przez ekspertów. Do porównania zgodności wyniow zwracanych przez algorytm i danychz góry etykiet będziemy stosować: indeks Fowlkesa-Mallowsa (dendextend::FM_index()) oraz skorygowany indeks Randa (mclust::adjustedRandIndex()). Każdy z powyższych indeksów zwraca wartość równą 1, jeśli dane podziały są równoważne. Im wartość jest mniejsza od 1 tym gorzej dany algorytm sie sprawuje i tym mniejszą skutecznościa się on charakteryzuje.

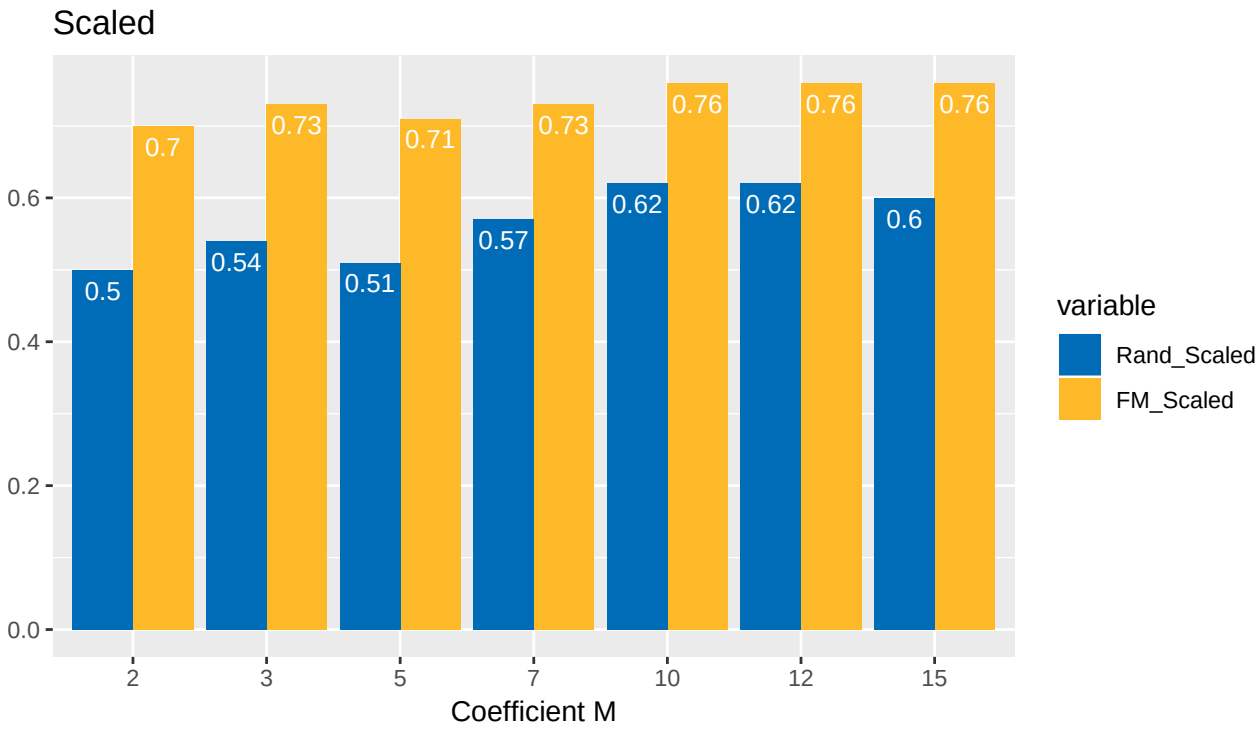
Algorytm spektralny

Prezentacje rozpoczne od pokazania dzialania algorytmu zaimplementowanego przeze mnie w zaleznosci od różnych wartosci parametru M. Parametr ten określa sposób generowania macierzy sąsiedztwa dla punktów analizowanego zbioru. Ponieważ działanie tego algorytmu opiera się na wyłonieniu k losowo wybranych centroidów, za każdym razem, gdy algorytm jest wywoływany, można uzyskać inny rezultat. Ponadto proces clusteringu jest wrażliwy na początkowy wybór środków. Funkcja kmeans () ma parametr nstart, który zaczyna od wielu początkowych konfiguracji i wybiera tę najkorzystniejszą. Uznałem ze ustawię opcje nstart=25, aby zwiększyć szansę na otrzymanie dobrych wyników.

Oto wartosci odpowiednio wspolczynnikow indeksow Randa oraz FM.

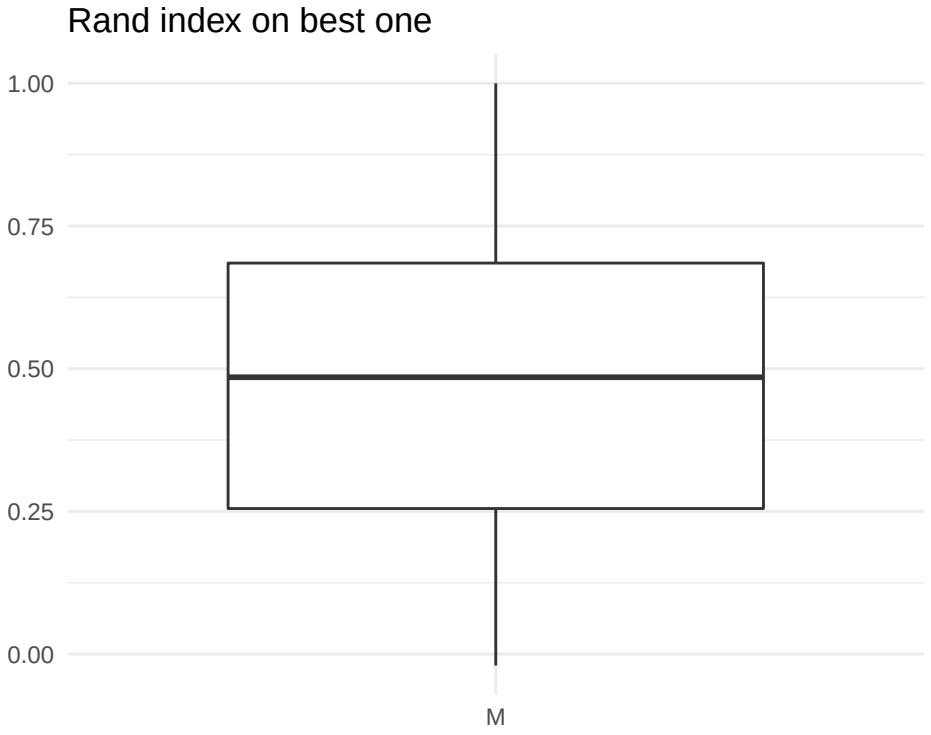


Jak widac na wykresach najlepiej algorytm prezentuje sie dla parametru M rownego 3 a takze dla M=12-15. Jednakze roznice pomiedzy posczegolnymi wybranymi przeze mnie wartosciami M nie sa zbyt duze i nie powoduja one znaczacej roznicy w wynikach dzialania. Sprawdzmy jak algorytm sprawuje sie dla danych ustandaryzowanych.



Standaryzacja w niewielki sposób wpłynęła na wyniki, pozwoliła jedynie podnieść wynik indeksu Randa 0.02, co jest znikomą poprawa.

Zdecydowałem się również sprawdzi jak będzie wyglądało rozłożenie wyników osiąganych przez algorytm dla poszczególnych zbiorów, dla najlepszego M. Chcę w ten sposób sprawdzić czy algorytm generuje stabilne wyniki, czyli czy można ufać jego wynikom. Dla większej czytelności użyje jedynie indeksu Randa, aby lepiej móć zobaczyć sprawowanie się algorytmu.



Widać ze wyniki są dość mocno rozbieżne dla różnych zbiorów co widać poprzez dość szerokie pudełko na wykresie pudełkowym. Widać tutaj, iż algorytm nie do końca dobrze radzi sobie z bardziej skomplikowanymi zbiorami.

Jeszcze jedna rzecza ktora chcialbym sprawdzic jest dokladnie ile wynikow wpada do jakiego przedzialu dla pokazanego powyzej algorytmu puszczonego dla jednego konkretnego M. Uwazam iz w ten sposob bedzie mozna dokladnie zobaczyc jak prezentuje sie rozklad danych, ktore algorytm daje.

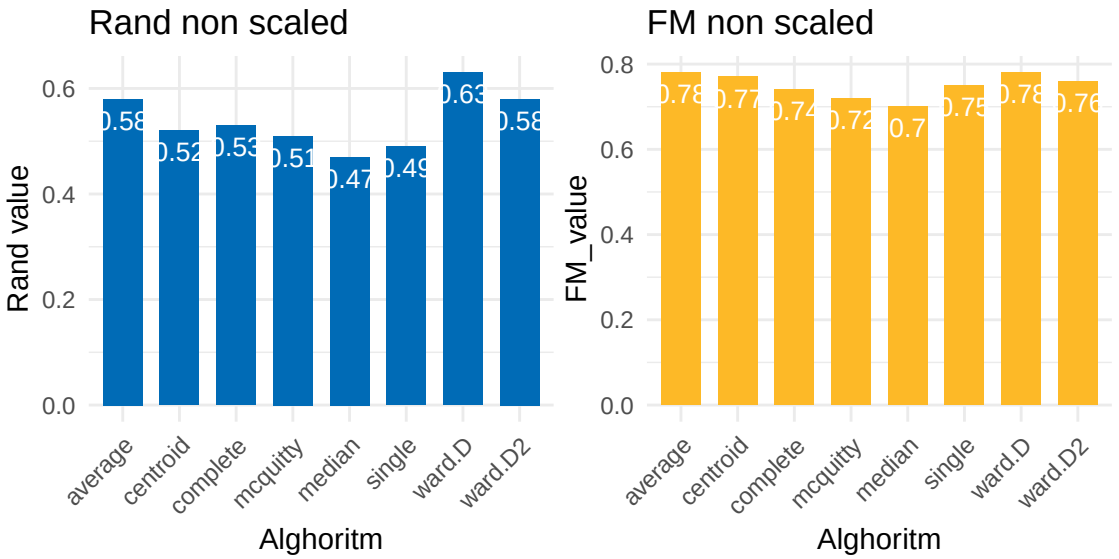
Przedzialy	Wartosci
0	1
(0,0.25]	11
(0.26,0.5]	14
(0.51,0.75]	11
(0.76,1]	9

Tak jak widać na wykresie pudełkowym nawjwiększa częsc danych wpadła do środkowych kubełkow. Jednocześnie widzimy teraz jednoznacznie ze do najlepszego kubełka wpadlo 9 obserwacji. Jest to wartosc ktora na pewno nalezy zapamietac do porownania wzgledem nastepnych algorytmow.

Nawiazaując do tego co napisałem w raporcie podsumowującym stworzone przeze mnie zbiory, widac ze algorytm osiaga nizsze średnie wyniki niz te ktore mial dla moich zbiorkow. Potwierdzilo sie wiec moje przypusczenie ze to "prostość" moich zbiorow miala wplyw na wynik algorytmu. Podsumowując ten algorytm można powiedziec ze dziala on w miare poprawnie, jednakże jego wyniki nie sa oszałamiajace. W następnej czesci raportu pozwolę sobie porównac jego wyniki z innymi algorytmami ktore testowałem.

Stats

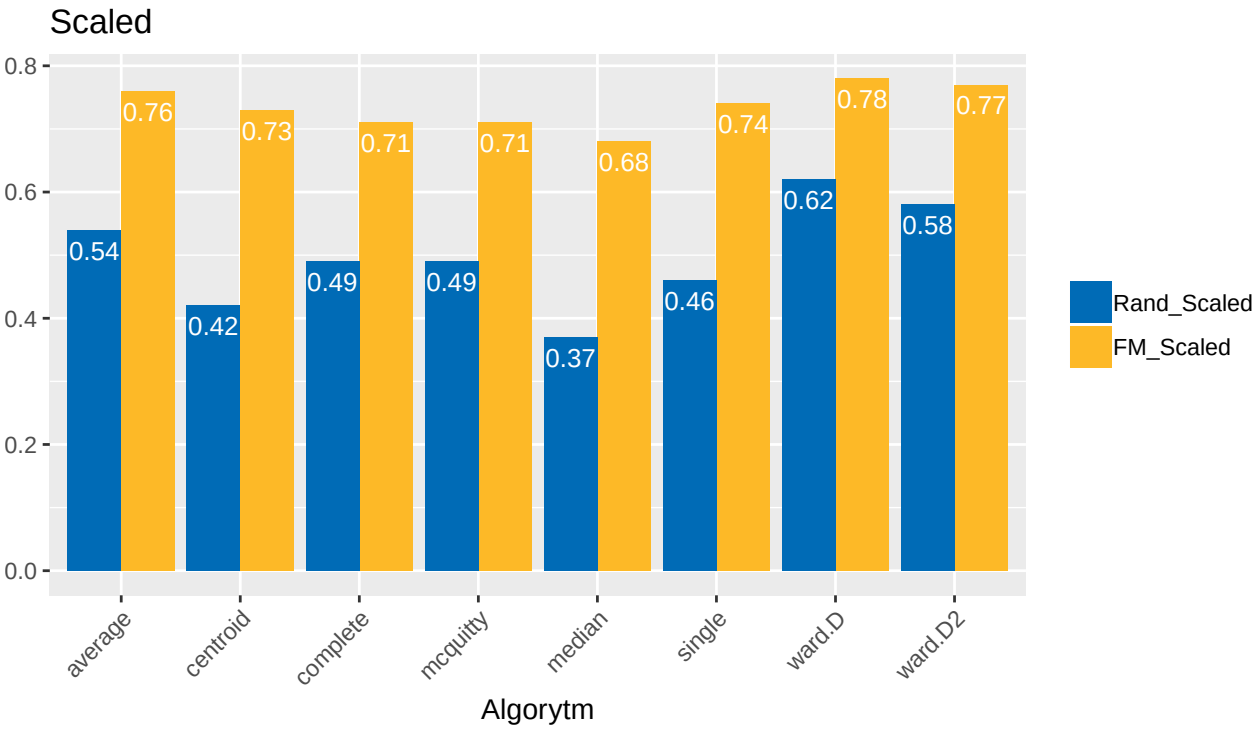
W tej czesci raportu zaprezentuje dzialanie algorytmow z pakietu stats. Zawarte jest w nim 8 algorytmow grupowania hierarchicznego. Przeanalizowalem dzialanie kazdego z nich co przedstawie teraz na wykresach.



W tym przypadku widac juz roznice w stosunku do do poprzedniego algorytmu, gdyz jest 'lider' ktory osiaga najlepsze wyniki. Liderem tym jest algorytm "ward.D".

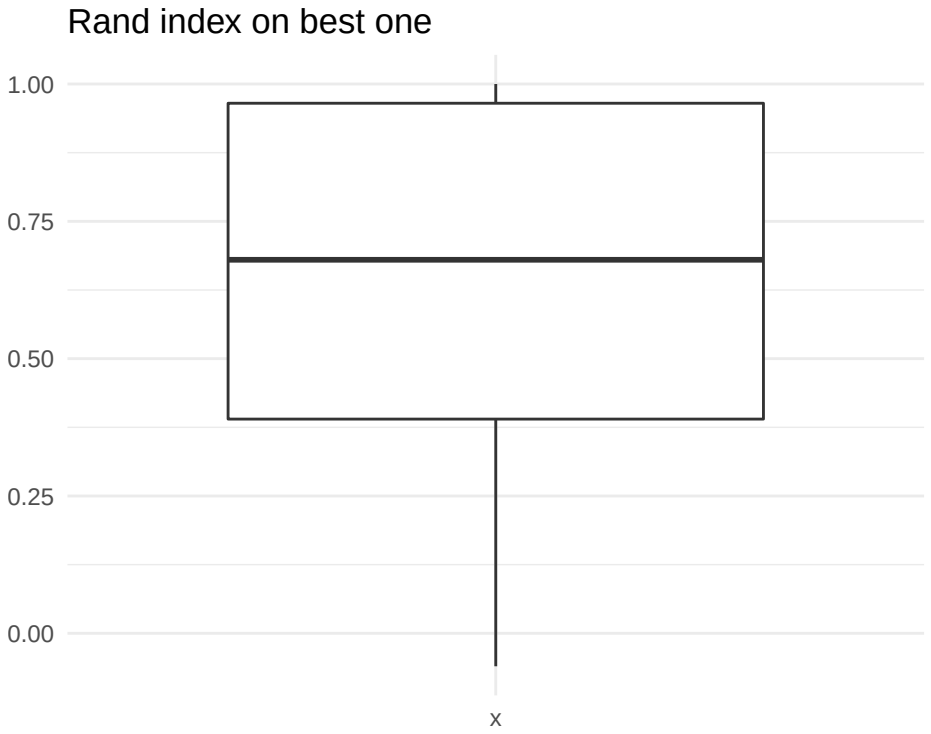
Rowniez wyniki indeksu FM zdaja sie to potwierdzac, jednakze inne algorytmy rowniez osiagaja dobre wyniki. Widac jednak ze algorytmy zawarte w pakiecie stats sprawuja sie troszke lepiej niz algorytm napisany przeze mnie. Z pewnoscia wynika to z jakosci napisanego kodu, jak i innego podejscia do problemu.

Sprawdzmy czy tym razem standaryzacja pozwoli poprawic wyniki.



Jak widac standaryzacja nie poprawiła wyników, a miejscami wręcz niepokojaco pogarsza wyniki.

Tak samo jak przy algorytmie spektralnym wykonam wykres pudełkowy, aby zobaczyć jak rozkładają sie wyniki najlepszego algorytmu z pakietu stats. Dzięki temu będziemy mogli wyciągnąć już ciekawe wnioski dotyczące zachowania oraz stabilności algorytmów.



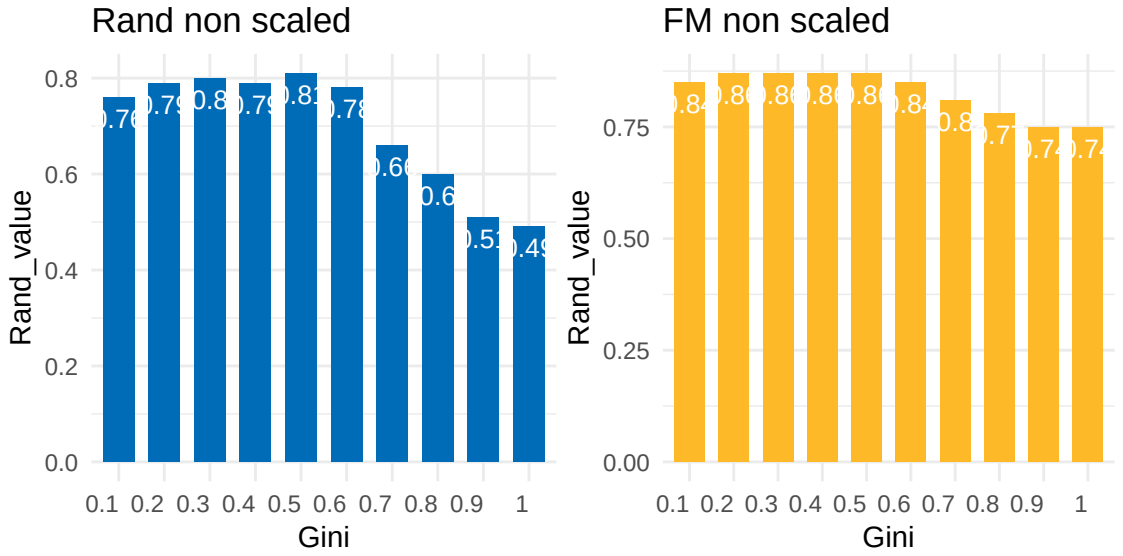
Widać tutaj jeszcze większy rozrzut wyników niz w przypadku algorytmu spektralnego. Pozwala to sądzić ze algorytm hclust jest jeszcze mniej odporny na różnego rodzaju odchyły przy róznych zbiorach, jednakże trzeba mu oddać ze prezentuje znacznie lepsze wyniki niż stworzony przeze mnie algorytm, jest po prostu skuteczniejszy.

Przedzialy	Wartosci
0	1
(0,0.25]	8
(0.26,0.5]	7
(0.51,0.75]	9
(0.76,1]	21

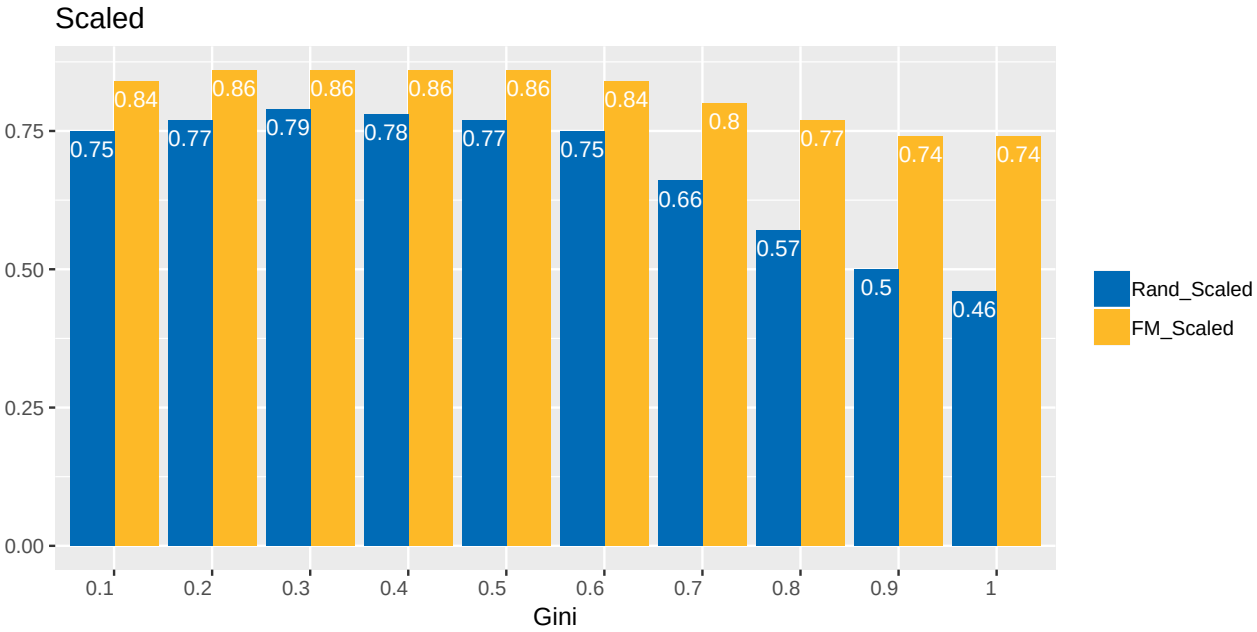
Widać znaczną poprawę względem poprzedniego algorytmu. Mamy znacznie więcej obserwacji w najwyższym kubełku, co oczywiście oznacza większą skuteczność działania.

Genie

Kolejnym algorytmem ktory przetestowalem jest hclust2 z pakietu Genie. Pozwala on na rozne dostosowanie parametru thresholdGini w zaleznosci od ktorego osiaga on rozbiezne wyniki. Zdecydowalem sie sprawdzic jak ten algorytm bedzie sprawowal w zaleznosci od tego parametru.

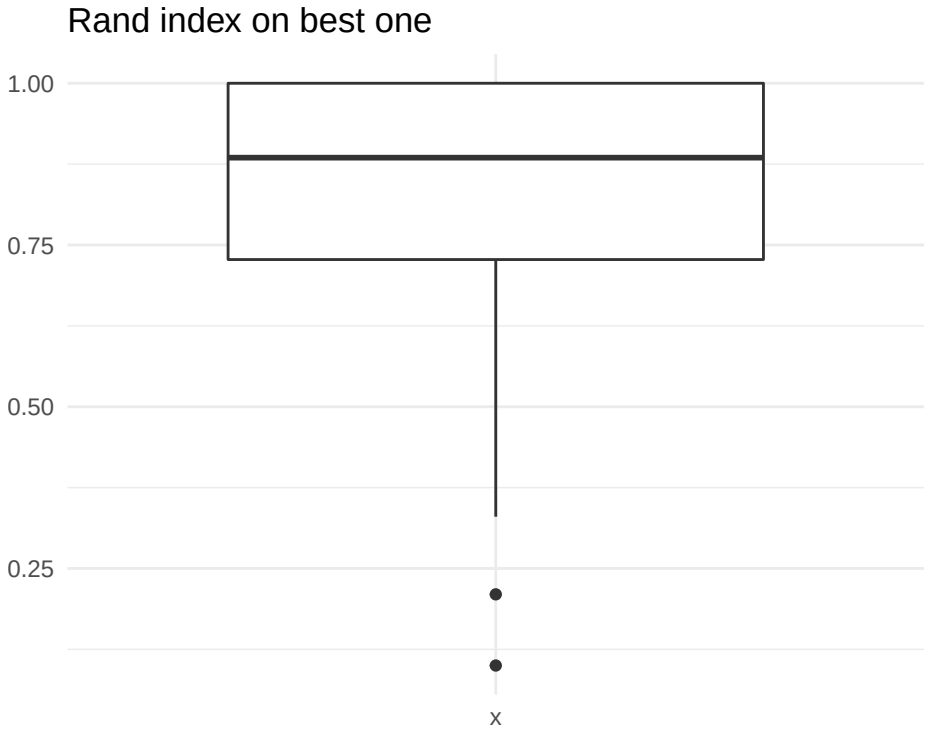


Udalo sie osiagnac takie same wyniki dla thresholdGini rownego 0.3 i 0.5. Widać również że ten algorytm radzi sobie bardzo dobrze z różnymi rodzajami zbiorow danych. Zgodnie z tym co jest napisane w opisie algorytmu na CRANie sprawuje sie on lepiej niz algorytm ward.D z biblioteki stats, ktory z kolei byl najlepszy ze wszystkich dostepnych w niej algorytmow. Roznica w wynikach jest dosc znaczaca bo wynosi prawie 0.1, co jest spora roznica w jakosci dzialania. Jednoczesnie dlugosc dzialania algorytmu hclust2 nie jest wieksza, wiec smialo mozna powiedziec ze jest on lepszy niz hclust z biblioteki stats. Sprawdzmy czy moze tym razem standaryzacja danych wejsciowych pomoze poprawic wyniki.



Tym razem standaryzacja nie przyniosla praktycznie zadnych zmian w wynikach, wiec mozna powoli dojsc do wniosku, na podstawie dzialan tych algorytmow ktore przedstawilem do tej pory, iz nie ma ona zbyt wielkiego wplywu na wyniki, badz nawet czasem powoduje gorsze sprawowanie sie algorytmu.

Poraz kolejny sprawdźmy rozrzut wyników stworzonych przez algorytm.



Tutaj widać już znaczną róznicę względem poprzednich algorytmów. Rozstrzał wyników jest dużo mniejszy, a i same wyniki są znacznie wyższe. Śmiało można stwierdzić ze ten algorytm prezentuje się zdecydowanie najlepiej. Generuje najlepsze wyniki i robi to w sposób najbardziej stabilny, co powoduje, że ciężko mieć do niego jakiekolwiek zastrzeżenia.

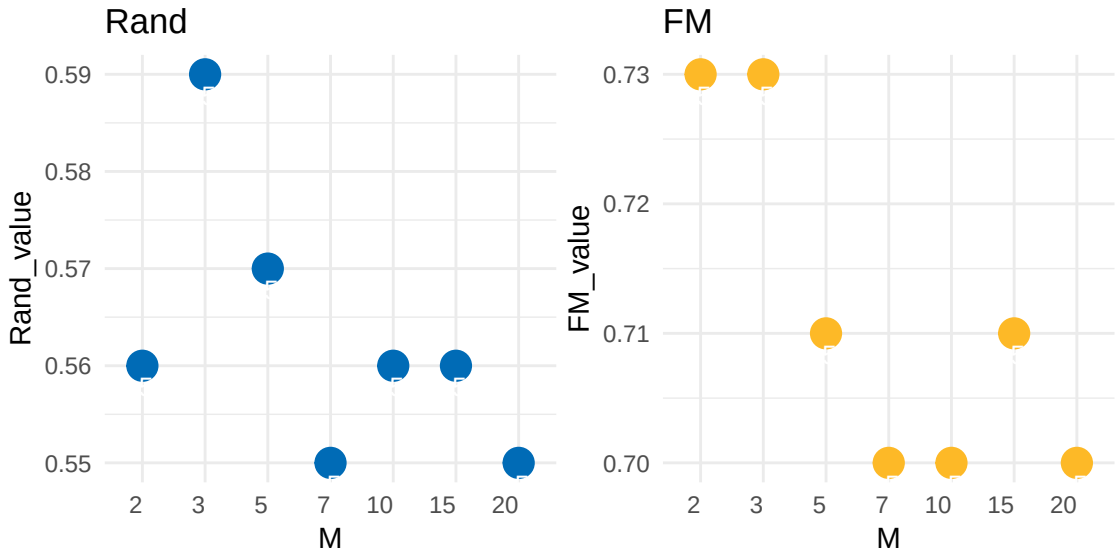
Sprawdzmy jeszcze dokladny rozklad wynikow w poszczegolnych kubelkach.

Przedzialy	Wartosci
(0,0.25]	2
(0.26,0.5]	5
(0.51,0.75]	5
(0.76,1]	34

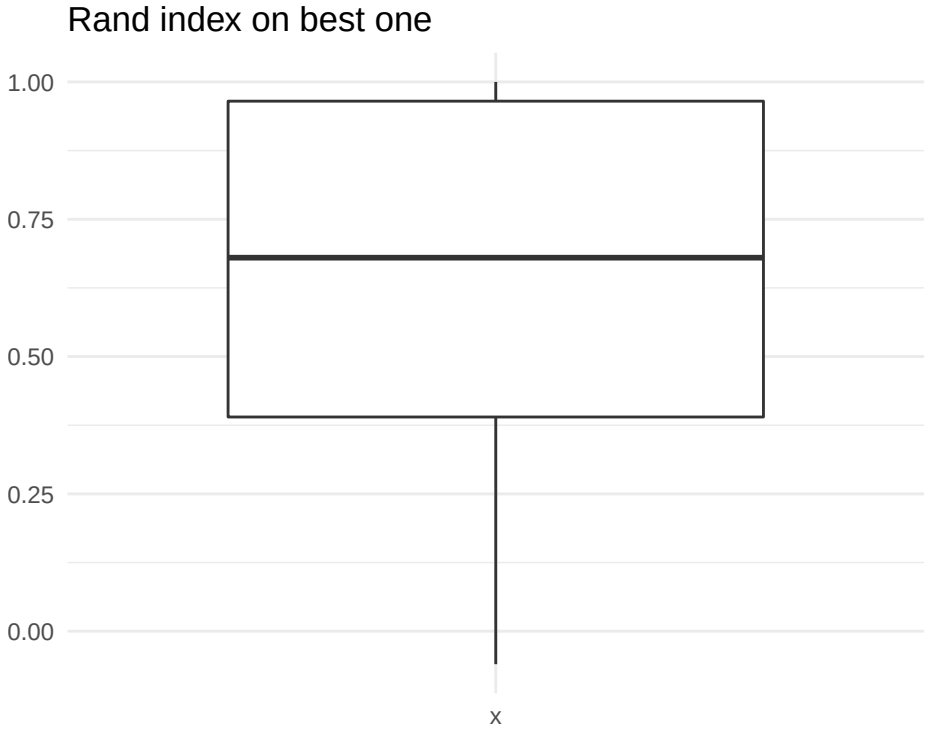
Algorytm uzyskał oszałamiające 34 na 46 wyników w najwyższych kubełku, co oznacza prawie 75% zbiorów udało mu sie sklasyfikować co najmniej bardzo dobrze. Jest to zdecydowanie najlepszy wynik spośród wszystkich z porownywanych przeze mnie algorytmów.

Cmeans

Jako swoj algorytm zdecydowalem sie uzyc algorytmu cmeans z pakietu e1071. Przeprowadza on tzw. fuzzy clustering, ktory charakteryzuje sie tym iz tutaj przykladowe punkty moga nalezec do kilku clusterow, co w normalnym procesie clusteringu nie ma miejsca. Algorytm ten zwraca rowniez niejako prawdopodobienstwo tego ze jakis punkt moze nalezec do ktoregos clustera, co rowniez wyroznia go od innych algorytmow. Cmenas udostepnia parametr M, który określa "zamglenie" zbioru. Sterując tym parametrem można zmieniać jego zachowanie, a co za tym idzie uzyskiwać inne wyniki. Z racji tego że poprzednie próby pokazały ze standaryzacja nie daje praktycznie żadnych pozytywnych efektóœ w działaniu algorytmów zdecydowałem, iż teraz nie będę przedstawiał wyników po standaryzacji danych. Zdecydowałem się również na troszeczkę inne przedstawienie danych osiąganych przez ten algorytm, gdyż myślę że dzięku temu lepiej widoczne będzie jego zachowanie.



Widać zaróœno po zachowaniu współczynniku FM jak i Randa, jak wraz ze wzrostem parametru zamglenia wyniki delikatnie spadają, jednakże nie są to znaczące wielkości. Algorytm dość małe wahania wyników w zależności od parametru zamglenia.
Sprawdzmy poraz koljeny jak przedstawia się rozkład wyników na poszczególnych zbiorkach.



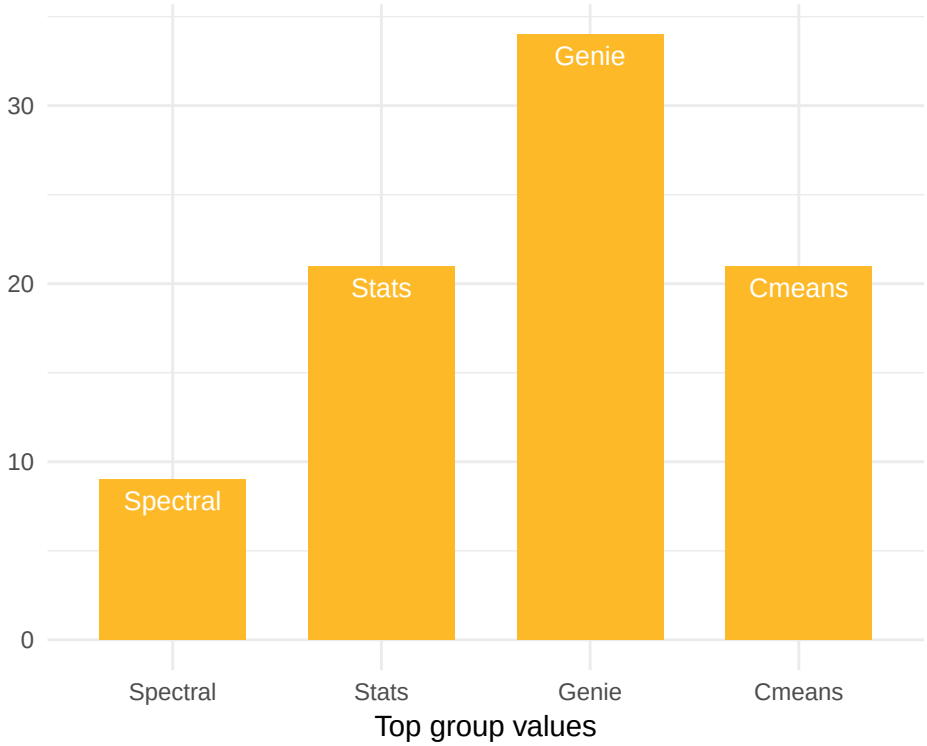
Algorytm ten prezentuje dość spory rozrzut, smiałbym stwierdzić że jeden z większych spośród testowanych przeze mnie. Jednocześnie jego średnia działania dość dobra. Sprawdźmy do jakich kubełków wpadają poszczególne zbiory, co myślę że daje jeden z bardziej miarodajnych obrazóœ jakości działania algorytmu./

Przedzialy	Wartosci
0	1
(0,0.25]	8
(0.26,0.5]	7
(0.51,0.75]	9
(0.76,1]	21

Cmeans spisał się gorzej od algorytmów z pakietu stats, ale jednak lepiej od mojego algorytmu spektralnego. Warto zauważyć ze cmeans działa bardzo szybko, co z pewnością można odnotować na jego plus. Jednakże dokładność jego wyliczeń pozostawia wiele do życzenia.

Wnioski

Sprawdźmy na jakiej ilości zbiorów ze wszystkich 46 testowanych przeze mnie każdy z algorytmów zdołal zdobyć wynik powyżej 0.75, co można będzie uznać za najlepszej jakości miernik sprawności działania tych wszytkich algorytmów.



Tak jak pisałem wcześniej genie zdecydowanie wygrywa swoją dokładnością oraz umiarkowaną predkością działania. Pozostałe algorytmy sprawują sie dobrze, jednakże można mieć co do nich poważne zastrzeżenia co do sporego rozrzutu przez nie generowanego. Podsumowując, widać iż problem clusteringu nie jest problemem trywialnym. Alogrytmy często mają róznego rodzaju problemy z odpowiednim pogrupowaniem danych. Jednakże, jak udało mi się pokazać w tej analizie istnieją takie algorytmy, które radzą sobie z tym problemem bardzo dobrze.