Data yang digunakan:

1. **Liver Disease Patient Dataset 30K Train Data:**
   https://www.kaggle.com/datasets/abhi8923shriv/liver-disease-patient-dataset

2. **Distribusi Pembagian dan Pengeluaran Total per Kapita dan Gini Ratio 2005 - 2020:**
   https://jakarta.beta.bps.go.id/id/statistics-table/1/MjM3IzE%253D/distribusi-pembagian-total-pengeluaran-per-kapita-dan-gini-ratio--2005-2020.html

3. **US.News and World Report's College Data:**
   https://www.kaggle.com/datasets/flyingwombat/us-news-and-world-reports-college-data

1. PRINCIPLE COMPONENT ANALYSIS

- Menampilkan 15 data teratas:

1	College Name	Apps	Accept	Enroll	F.Undergrad	P.Undergrad	Books	Personal
2	Abilene Christian University	1660	1232	721	2885	537	450	2200
3	Adelphi University	2186	1924	512	2683	1227	750	1500
4	Adrian College	1428	1097	336	1036	99	400	1165
5	Agnes Scott College	417	349	137	510	63	450	875
6	Alaska Pacific University	193	146	55	249	869	800	1500
7	Albertson College	587	479	158	678	41	500	675
8	Albertus Magnus College	353	340	103	416	230	500	1500
9	Albion College	1899	1720	489	1594	32	450	850
10	Albright College	1038	839	227	973	306	300	500
11	Alderson-Broaddus College	582	498	172	799	78	660	1800
12	Alfred University	1732	1425	472	1830	110	500	600
13	Allegheny College	2652	1900	484	1707	44	400	600
14	Allentown Coll. of St. Francis de Sales	1179	780	290	1130	638	600	1000
15	Alma College	1267	1080	385	1306	28	400	400

Data menggunakan tujuh variable dengan skala yang berbeda-beda dan menghapus kolom "College Name" saat melakukan analisis. Total observasi adalah sebanyak 777 buah diwakili oleh nama kampus. Akan dilakukan reduksi dimensi menggunakan analisis komponen utama untuk mendapatkan komponen-komponen yang dapat mewakilkan variasi pada data dengan menggunakan kombinasi linear.

- Sebelum itu cek terlebih dahulu nilai null pada data

```
```{r Melihat banyak nilai null}
sum(is.na(data))
```
```

```
[1] 0
```

Setelah dicek dengan function *is.na* ternyata tidak ada nilai null. Maka lanjut ke analisis.

- Tahap pertama yang perlu dilakukan adalah standarisasi pada data untuk menyamakan skala sehingga data akan diubah sedemikian sehingga memiliki rata-rata (mean) 0 dan standar deviasi 1 menggunakan fungsi *scale()*. Hal ini juga dilakukan karena bisa mempengaruhi matriks varian kovarians nantinya.

```
```{r Standarisasi data}
data1 <- as.data.frame(scale(data))
data1_matrix <- round(as.matrix(data1),digits=3)
cat("\nData Hasil Standarisasi:\n")
head(data1_matrix, 5)
```
```

```
Data Hasil Standarisasi:
       Apps Accept Enroll F.Undergrad P.Undergrad  Books Personal
[1,] -0.347 -0.321 -0.063      -0.168      -0.209 -0.602    1.269
[2,] -0.211 -0.039 -0.288      -0.210       0.244  1.215    0.235
[3,] -0.407 -0.376 -0.478      -0.549      -0.497 -0.905   -0.259
[4,] -0.668 -0.681 -0.692      -0.658      -0.520 -0.602   -0.688
[5,] -0.726 -0.764 -0.780      -0.711       0.009  1.518    0.235
```

Ini adalah hasil dari standarsisasi data, menampilkan sebanyak 5 data teratas dan pembulatan 3 angka di belakang koma.

- Mencari matriks varian kovarians sebagai masukan untuk mendapatkan nilai eigen dan vector eigen.

```{r Matriks varian covarian}
covarian <- var(data1)
cat("\nCovarian:\n")
round(covarian,digits=3)
```

```
Covarian:
             Apps Accept Enroll F.Undergrad P.Undergrad Books Personal
Apps        1.000  0.943  0.847       0.814       0.398 0.133    0.179
Accept      0.943  1.000  0.912       0.874       0.441 0.114    0.201
Enroll      0.847  0.912  1.000       0.965       0.513 0.113    0.281
F.Undergrad 0.814  0.874  0.965       1.000       0.571 0.116    0.317
P.Undergrad 0.398  0.441  0.513       0.571       1.000 0.081    0.320
Books       0.133  0.114  0.113       0.116       0.081 1.000    0.179
Personal    0.179  0.201  0.281       0.317       0.320 0.179    1.000
```

Ini adalah matriks varian kovarians antar variabel. Diagonal pada matriks merupakan varian pada variabel tersebut. Non-diagonal merupakan nilai kovarians antar kedua variabel.

- Tahap kedua adalah mencari nilai eigen dari matriks kovarians yang didapatkan. Nilai eigen yang didapatkan akan digunakan untuk mencari vektor eigen.

```{r Mencari nilai eigen dan vektor eigen}
eigen <- eigen(covarian)
eigen_val = eigen$values
eigen_vec = eigen$vectors
cat("Nilai Eigen:\n")
round(eigen_val,digits=3)
cat("\nVektor Eigen:\n")
round(eigen_vec,digits=3)
```

```
Nilai Eigen:
[1] 4.117 1.106 0.890 0.605 0.207 0.048 0.028

Vektor Eigen:
       [,1]   [,2]   [,3]   [,4]   [,5]   [,6]   [,7]
[1,] -0.447 -0.184  0.200  0.149  0.610  0.553 -0.163
[2,] -0.465 -0.183  0.149  0.125  0.272 -0.724  0.338
[3,] -0.474 -0.105  0.033  0.072 -0.456 -0.159 -0.724
[4,] -0.472 -0.058 -0.029  0.009 -0.545  0.378  0.577
[5,] -0.308  0.211 -0.438 -0.793  0.191 -0.030 -0.041
[6,] -0.089  0.667  0.714 -0.191 -0.040 -0.012  0.003
[7,] -0.183  0.655 -0.485  0.540  0.105 -0.018 -0.006
```

Ini adalah nilai eigen dan vektor eigen yang didapatkan. Disini sudah terurut dari nilai eigen dari yang terbesar ke terkecil. Sehingga vektor eigen juga berurutan seperti nilai eigennya. Vektor eigen akan digunakan sebagai pengali untuk mendapatkan komponen utama.

- Tahap ketiga adalah mencari banyaknya komponen utama yang optimal berdasarkan proporsi total kumulatif. Terlebih dahulu menghitung nilai proporsi kumulatif pada tiap nilai eigen dengan cara membagi nilai masing-masing nilai eigen terhadap jumlah seluruh nilai eigen, lalu dikali 100%.

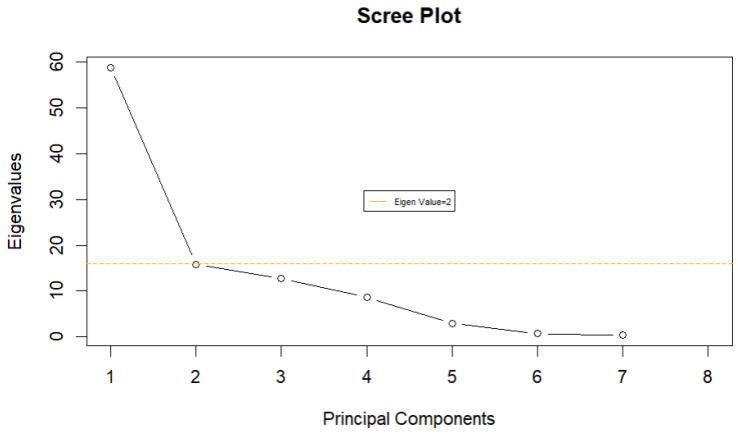
```{r Menghitung proporsi pada masing-masing nilai eigen}
sum = sum(eigen_val)
percentage_eigen_values <- (eigen_val / sum) * 100
cat("\nProporsi Nilai Eigen:\n")
round(percentage_eigen_values,digits=3)
```

```
Proporsi Nilai Eigen:
[1] 58.814 15.803 12.709  8.640  2.952  0.687  0.396
```

Ini adalah proporsi kumulatif masing-masing nilai eigen. Karena tadi nilai eigen sudah berurut dari terbesar sampai terkecil. Maka disini proporsi juga terurut. Nilai eigen pertama menjelaskan 58.8% varians pada data. Begitulah seterusnya. Disini saya memakai komponen utama yang menjelaskan proporsi kumulatif total sebesar 70%. Sehingga akan dipilih nilai eigen 1 dan nilai eigen 2 karena proporsi kumulatif total yang didapatkan adalah sebesar 74.6%.

- Cara lain mencari banyaknya komponen utama adalah dengan menggunakan scree plot.

```{r Menggunakan plot}
plot(percentage_eigen_values[1:8], type = "b", main = "Scree Plot", xlab = "Principal Components", ylab = "Eigenvalues")
abline(h=16, col="orange",lty=2)
legend("center",legend=c("Eigen Value=2"), col=c("orange"),lty=5,cex=0.5)
```



Ini adalah gambar scree plot. Dari sini terlihat bahwa grafik mulai melandai pada titik dimana komponen utama sebanyak 2. Sehingga dapat disimpulkan bahwa komponen utama yang optimal dipakai adalah sebanyak 2 buah.

```r
```{r Menampilkan kolom hasil PCA}
eigen_sel <- eigen_vec[, 1:2]
transform <- data1_matrix %*% eigen_sel
df_transform <- as.data.frame(transform)
round(df_transform,digits=3)
```
```

Description: df [777 × 2]

V1 <dbl>	V2 <dbl>
0.299	0.524
0.121	1.104
1.124	-0.652
1.594	-0.603
1.204	1.565
1.554	-0.620
1.400	0.235
0.891	-0.858
1.471	-1.620
1.134	1.115

1-10 of 777 rows

- Ini merupakan dua komponen utama hasil dari analisis komponen utama yang dapat menejelaskan 74.6% varians pada data.

2. FACTOR ANALYSIS

- Menampilkan 15 data teratas:

1	College Name	Apps	Accept	Enroll	F.Undergrad	P.Undergrad	Personal
2	Abilene Christian University	1660	1232	721	2885	537	2200
3	Adelphi University	2186	1924	512	2683	1227	1500
4	Adrian College	1428	1097	336	1036	99	1165
5	Agnes Scott College	417	349	137	510	63	875
6	Alaska Pacific University	193	146	55	249	869	1500
7	Albertson College	587	479	158	678	41	675
8	Albertus Magnus College	353	340	103	416	230	1500
9	Albion College	1899	1720	489	1594	32	850
10	Albright College	1038	839	227	973	306	500
11	Alderson-Broaddus College	582	498	172	799	78	1800
12	Alfred University	1732	1425	472	1830	110	600
13	Allegheny College	2652	1900	484	1707	44	600
14	Allentown Coll. of St. Francis de Sales	1179	780	290	1130	638	1000
15	Alma College	1267	1080	385	1306	28	400

Data masih sama dengan yang digunakan pada analisis PCA, namun disini hanya menggunakan enam variable dengan skala yang berbeda-beda dan menghapus kolom “College Name” saat melakukan analisis. Total observasi adalah sebanyak 777 buah diwakili oleh nama kampus. Akan dilakukan reduksi dimensi menggunakan analisis faktor untuk mendapatkan factor-faktor yang dapat mewakilkan variasi pada data.

- Sebelum itu cek terlebih dahulu nilai null pada data

```r
```{r Melihat banyak nilai null}
sum(is.na(dataku))
```
```

```
[1] 0
```

Setelah dicek dengan function *is.na* ternyata tidak ada nilai null. Maka lanjut ke analisis.

- Tahap pertama yang perlu dilakukan adalah standarisasi pada data untuk menyamakan skala sehingga data akan diubah sedemikian sehingga memiliki rata-rata (mean) 0 dan standar deviasi 1 menggunakan fungsi *scale()*.

```
```{r Standarisasi data}
datame <- scale(dataku)
cat("\nData hasil standarisasi:\n")
head(round(datame,digits=3))
```
```

```
Data hasil standarisasi:
       Apps Accept Enroll F.Undergrad P.Undergrad Personal
[1,] -0.347 -0.321 -0.063      -0.168      -0.209    1.269
[2,] -0.211 -0.039 -0.288      -0.210       0.244    0.235
[3,] -0.407 -0.376 -0.478      -0.549      -0.497   -0.259
[4,] -0.668 -0.681 -0.692      -0.658      -0.520   -0.688
[5,] -0.726 -0.764 -0.780      -0.711       0.009    0.235
[6,] -0.624 -0.628 -0.669      -0.623      -0.535   -0.983
```

Ini adalah hasil dari standarsisasi data, menampilkan sebanyak 5 data teratas dan pembulatan 3 angka di belakang koma.

- Tahap kedua adalah melakukan uji Bartlett Sphericity untuk melihat ada atau tidaknya korelasi antar variabel yang dicakup dalam penelitian. Lalu KMO untuk kecukupan sampel pada tiap variabel dengan tujuan menunjukkan seberapa cocok data tersebut dianalisis menggunakan analisis factor.

```
```{r Uji bartlett sphericity dan KMO}
bart_spher(datame)
KMO(datame)
```
```

```
        Bartlett's Test of Sphericity

Call: bart_spher(x = datame)

     X2 = 5609.801
     df = 15
p-value < 2.22e-16

Kaiser-Meyer-Olkin factor adequacy
Call: KMO(r = datame)
Overall MSA =  0.79
MSA for each item =
       Apps      Accept      Enroll F.Undergrad P.Undergrad    Personal
       0.80        0.78        0.77        0.78        0.89        0.89
```

Untuk uji Bartlett Sphericity:
H0 : Tidak terdapat korelasi antar variabel yang signifikan
H1 : Terdapat korelasi antar variabel yang signifikan
Ini adalah hasil dari uji Bartlett Sphericity, didapatkan p- value yang kurang dari 0.5 sehingga tolak H0, terdapat korelasi antar variabel yang signifikan. Lalu dari KMO terlihat bahwa nilai MSA tiap variabel dan nilai MSA total adalah lebih dari 0.5 sehingga jumlah sampel yang dicakup dalam data cukup untuk dilakukan pada analisis faktor.

- Tahap ketiga adalah mencari nilai eigen pada data sebagai salah satu acuan dalam menentukan banyaknya faktor. Berdasarkan aturan Kaiser, jumlah faktor ditentukan berdasarkan nilai eigen > 1.

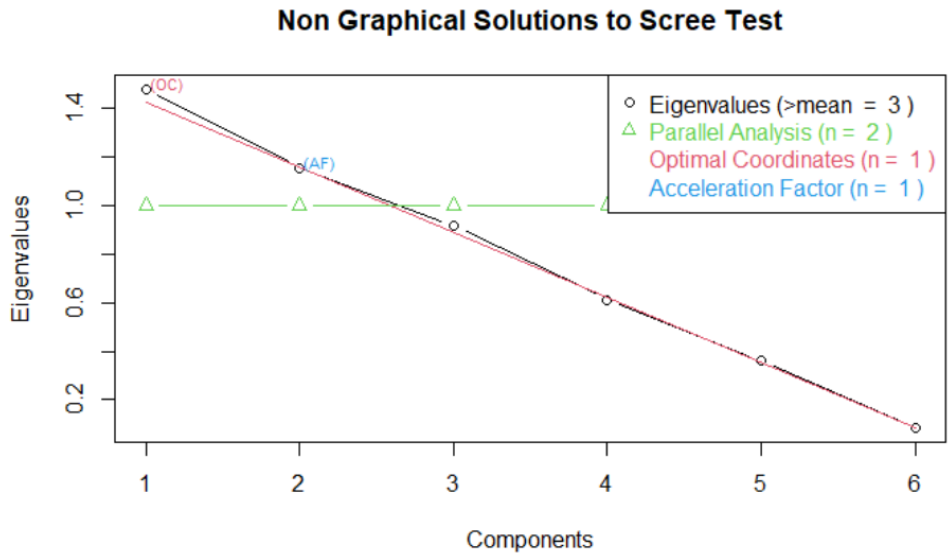
```{r Mencari nilai eigen dan vektor eigen}
R <- cov(datame)
eigen_data <- eigen(R)
cat("\nNilai Eigen:\n")
eigen_data $values
```

```
Nilai Eigen:
[1] 4.09202508 1.00634567 0.61789318 0.20783379 0.04819278 0.02770950
```

Ini adalah nilai eigen yang didapatkan. Terlihat bahwa nilai eigen > 1 adalah sebanyak 2. Sehingga bisa dikatakan bahwa jumlah faktor terbaik yang dapat dipilih adalah sebanyak 2 faktor.

- Cara lainnya adalah dengan menggunakan scree plot.

```{r Penentuan jumlah factor}
ap <- parallel(subject=20, var=6, rep=100, cent=0.05)
nfaktor <- nScree (eigen$values, ap$eigen$qevpea)
plotnScree(nfaktor)
```



Dari gambar scree plot dapat dilihat berdasarkan parallel analysis, jumlah faktor yang digunakan adalah sebanyak 2 faktor sama seperti penentuan berdasarkan nilai eigen.

- Tahap keempat adalah melakukan analisis faktor menggunakan metode principle component.

```{r Analisis factor menggunakan metode Estimasi Principal Component}
solution_pa <- fa(R, nfactors=2, rotate="varimax", fm="pa")
solution_pa
```

	PA1 <S3: AsIs>	PA2 <S3: AsIs>	h2 <dbl>	u2 <dbl>	com <dbl>
Apps	0.91	0.19	0.8747766	0.12522342	1.090078
Accept	0.96	0.24	0.9788581	0.02114191	1.124918
Enroll	0.84	0.47	0.9291572	0.07084275	1.569407
F.Undergrad	0.78	0.57	0.9427902	0.05720978	1.832516
P.Undergrad	0.30	0.59	0.4433660	0.55663396	1.476163
Personal	0.09	0.46	0.2233886	0.77661138	1.067916

Nilai Loading variabel X1 sebesar 0.91 menunjukkan bahwa besarnya korelasi variabel Apps dan faktor umum 1 sebesar 0.91.

Nilai Loading variabel X2 sebesar 0.96, menunjukkan bahwa besarnya korelasi variabel Accept dan faktor umum 1 sebesar 0.96.
Nilai Loading variabel X3 sebesar 0.84, menunjukkan bahwa besarnya korelasi
variabel Enroll dan faktor umum 1 sebesar 0. 84.
Nilai Loading variabel X4 sebesar 0.78, menunjukkan bahwa besarnya korelasi variabel F.Undergrad dan faktor umum 1 sebesar 0.78.
Nilai Loading variabel X5 sebesar 0.30, menunjukkan bahwa besarnya korelasi variabel P.Undergrad dan faktor umum 1 sebesar 0.30.
Nilai Loading variabel X6 sebesar 0.46, menunjukkan bahwa besarnya korelasi variabel Personal dan faktor umum 1 sebesar 0.09.
Nilai loading dikuadratkan menunjukkan total varians variabel tertentu yang dapat dijelaskan oleh faktor. Sama halnya untuk interpretasi faktor 2.

```
Factor Analysis using method =  pa
Call: fa(r = R, nfactors = 2, rotate = "varimax", fm = "pa")
Standardized loadings (pattern matrix) based upon correlation matrix

                        PA1  PA2
SS loadings           3.18 1.22
Proportion Var        0.53 0.20
Cumulative Var        0.53 0.73
Proportion Explained  0.72 0.28
Cumulative Proportion 0.72 1.00

Mean item complexity =  1.4
Test of the hypothesis that 2 factors are sufficient.

df null model =  15  with the objective function =  7.26
df of  the model are 4  and the objective function was  0.42

The root mean square of the residuals (RMSR) is  0.02
The df corrected root mean square of the residuals is  0.03

Fit based upon off diagonal values = 1
Measures of factor score adequacy
                                                 PA1  PA2
Correlation of (regression) scores with factors   0.97 0.84
Multiple R square of scores with factors          0.95 0.71
Minimum correlation of possible factor scores     0.90 0.43
```

- Sum square Loading 1 (ML1) Faktor 1 sebesar 3.18, menunjukkan total varians keseluruhan variabel asal (X) yang dapat dijelaskan oleh faktor 1. Sum square Loading 2 (ML2) Faktor 2 sebesar 1.22 menunjukkan total varians keseluruhan variabel asal (X) yang dapat dijelaskan oleh faktor 2. Proportion var ML1/Proporsi keragaman faktor 1 sebesar 0.53 menunjukkan proporsi total varians variabel asal yang dapat dijelaskan oleh faktor 1 sebesar 53%. Proportion var ML2 /Proporsi keragaman faktor 2 sebesar 0.20 menunjukkan proporsi total varians variabel asal yang dapat dijelaskan oleh faktor 2 sebesar 20%. Karena nilai eigen ML1 dan ML2 lebih dari 1 dan kumulatif proporsi total variansi yang dapat dijelaskan oleh kedua faktor sudah mencapai 73%.

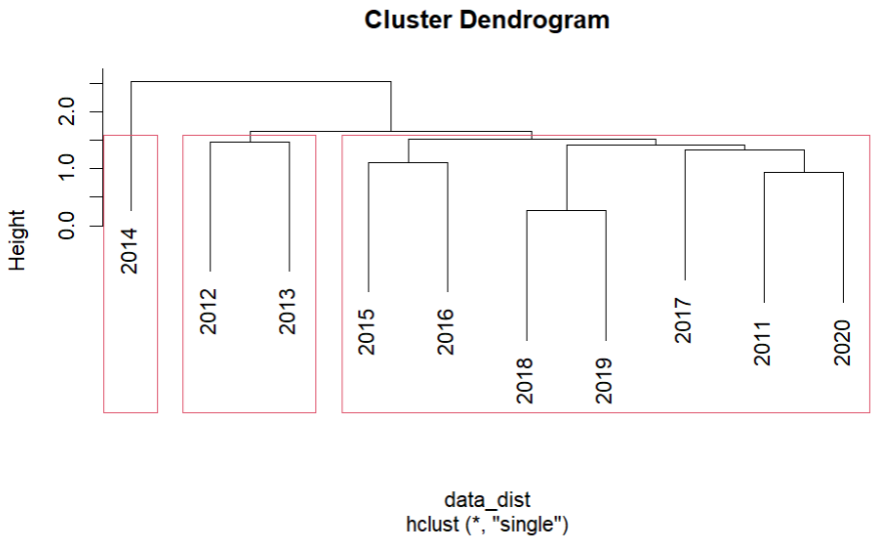
3. HIERARCICAL CLUSTERING

- Menampilkan data:

	Tahun / Year	Distribusi Pembagian Pengeluaran Per Kapita (40%) Rendah/Low	Distribusi Pembagian Pengeluaran Per Kapita (40%) Sedang/Middle	Distribusi Pembagian Pengeluaran Per Kapita (20%) Tinggi/High	Gini Ratio / Gini Index
1	2011	16,96	35,37	47,67	0,385
2	2012	15.71	35.51	48.77	0.421
3	2013	14.75	35.89	49.36	0.433
4	2014	15.54	34.17	50.29	0.431
5	2015	16.03	36.28	47.69	0.431
6	2016	16.09	36.28	47.69	0.411
7	2017	17.16	35.73	48.18	0.413
8	2018	17.3	36.03	46.81	0.394
9	2019	17.3	36.09	46.61	0.391
10	2020	17.25	35.11	47.65	0.399

Data menggunakan 4 variable dengan kolom “Gini Ratio/Gini Index” yang berbeda satuan. Total observasi adalah sebanyak 12 buah diwakili oleh tahun. Akan dilakukan pengelompokan hirarki untuk melihat beberapa kelompok yang berbeda berdasarkan kemiripan antar data.

- Tahap pertama adalah menjadikan kolom “Tahun/Year” sebagai label/kelas untuk visualisasi nanti. Kemudian melakukan standarisasi data pada kolom ke 2 sampai ke 5 untuk menyamakan skala sehingga data akan diubah sedemikian sehingga memiliki rata-rata (mean) 0 dan standar deviasi 1 menggunakan fungsi *scale()*.

```
```{r Standarisasi data}
dataku$`Tahun / Year` <- factor(dataku$`Tahun / Year`)
data <- as.data.frame(scale(dataku[,2:5]))
round(head(data, 5),digits=3)
```
```

Distribusi Pembagian Pengeluaran Per Kapita (40%) Rendah/Low <dbl>	Distribusi Pembagian Pengeluaran Per Kapita (40%) Sedang/Middle <dbl>
0.607	-0.426
-0.771	-0.210
-1.829	0.377
-0.958	-2.279
-0.418	0.979

Distribusi Pembagian Pengeluaran Per Kapita (20%) Tinggi/High <dbl>	Gini Ratio / Gini Index <dbl>
-0.357	-1.444
0.619	0.563
1.142	1.232
1.967	1.120
-0.339	1.120

Ini adalah hasil dari standarsisasi data, menampilkan sebanyak 5 data teratas dan pembulatan 3 angka di belakang koma.

- Tahap kedua adalah menghitung jarak antar variabel menggunakan perhitungan jarak Euclidean.

```
```{r Build jarak}
data_dist <- dist(data)
data_matrix <- as.matrix(data_dist)
round(data_matrix,digits=3)
```
```

```
       1     2     3     4     5     6     7     8     9    10
1  0.000 2.631 3.998 4.226 3.098 2.235 1.731 1.419 1.540 0.934
2  2.631 0.000 1.478 2.538 1.663 1.678 1.773 3.001 3.216 2.399
3  3.998 1.478 0.000 2.916 2.136 2.499 3.076 4.218 4.408 3.865
4  4.226 2.538 2.916 0.000 4.028 4.188 3.674 5.079 5.308 3.785
5  3.098 1.663 2.136 4.028 0.000 1.117 1.862 2.640 2.817 2.873
6  2.235 1.678 2.499 4.188 1.117 0.000 1.521 1.854 2.007 2.312
7  1.731 1.773 3.076 3.674 1.862 1.521 0.000 1.684 1.943 1.325
8  1.419 3.001 4.218 5.079 2.640 1.854 1.684 0.000 0.261 1.629
9  1.540 3.216 4.408 5.308 2.817 2.007 1.943 0.261 0.000 1.828
10 0.934 2.399 3.865 3.785 2.873 2.312 1.325 1.629 1.828 0.000
```

Ini adalah matriks hasil perhitungan antar variabel menggunakan perhitungan jarak euclidean. Untuk diagonal pada matriks bernilai 0 karena merupakan jarak ke variabel itu sendiri. Lalu untuk nilai matriks tersebut, semakin kecil nilainya maka antar dua data tersebut semakin mirip. Begitu pula sebaliknya.

- Tahap ketiga adalah membuat dendogram dengan metode aggromerative single. Aggromerative berarti data yang dikelompokkan mulai dari yang paling kecil atau dari bawah dan membentuk kelompok yang lebih besar atau semakin ke atas. Single linkage berarti mengukur jarak terpendek antar kelompok data.

```
```{r Plot 3 cluster}
single <- hclust(data_dist, method="single")
plot(single, dataku$`Tahun / Year`, labels=dataku$`Tahun / Year`)
rect.hclust(single,3)
```
```



Ini adalah gambar dendogram menggunakan metode aggromerative single linkage. Sumbu Y (Height) menunjukkan jarak atau perbedaan antara pengamatan atau kluster yang digabungkan. Sementara sumbu X menunjukkan pengamatan atau objek (Tahun). Meskipun dari sini dapat dikelompokkan menjadi 2, 3, 5, 6, 7, 8, 9, dan 10 kelompok menggunakan fungsi *rect.hclust*, disini saya menggunakan hanya 3 kelompok sebagai acuan untuk dilakukan analisis lebih lanjut.

- Tahap keempat adalah melihat tiap tahun beserta kelompoknya.

```
```{r Daftar nama & cluster}
anggota <- data.frame(id=dataku$`Tahun / Year`, cutree(single,k=3))
anggota
```
```

Description: df [10 × 2]

id <fctr>	cutree.single..k...3. <int>
2011	1
2012	2
2013	2
2014	3
2015	1
2016	1
2017	1
2018	1
2019	1
2020	1

1-10 of 10 rows

Dari sini terlihat bahwa kelompok 1 beranggotakan tahun 2011, 2015, 2016, 2017, 2018, 2019, dan 2020 yang berarti memiliki kemiripan tertentu, meskipun ada beberapa sub-kluster yang lebih

dekat satu sama lain. Lalu kelompok 2 beranggotakan tahun 2012 dan 2013 yang berarti sangat mirip satu sama lain. Terakhir kelompok 3 hanya beranggotakan tahun 2014 yang berbeda secara signifikan dari tahun-tahun lainnya.

- Tahap terakhir adalah melihat karakteristik yang membedakan tiap cluster. Disini saya membedakannya dari rata-rata tiap kelompok pada tiap variabel.

```{r Rata-rata nilai tiap cluster}
datas <- dataku[,2:5] %>%
  mutate(cluster = anggota[, 2])
summary <- datas %>%
  group_by(cluster) %>%
  summarise_all(mean)
summary
```

cluster	Distribusi Pembagian Pengeluaran Per Kapita (40%) Rendah/Low	Distribusi Pembagian Pengeluaran Per Kapita (40%) Sedang/Middle
1	16.87	35.84143
2	15.23	35.70000
3	15.54	34.17000

Distribusi Pembagian Pengeluaran Per Kapita (20%) Tinggi/High	Gini Ratio / Gini Index
47.47143	0.4034286
49.06500	0.4270000
50.29000	0.4310000

Dari gambar terlihat bahwa tiap kelompok dibedakan dari nilai rata-rata pada variabel yang berbeda-beda. Kelompok 1 memiliki rata-rata distribusi pembagian pengeluaran per kapita (40%) rendah sebesar 16.87, rata-rata distribusi pembagian pengeluaran per kapita (40%) sedang sebesar 35.84, rata-rata distribusi pembagian pengeluaran per kapita (20%) tinggi sebesar 47.47, dan rata-rata gini ratio sebesar 0.403.

Kelompok 2 memiliki rata-rata distribusi pembagian pengeluaran per kapita (40%) rendah sebesar 15.23, rata-rata distribusi pembagian pengeluaran per kapita (40%) sedang sebesar 35.70, rata-rata distribusi pembagian pengeluaran per kapita (20%) tinggi sebesar 49.06, dan rata-rata gini ratio sebesar 0.427.

Kelompok 3 memiliki rata-rata distribusi pembagian pengeluaran per kapita (40%) rendah sebesar 15.54, rata-rata distribusi pembagian pengeluaran per kapita (40%) sedang sebesar 34.17, rata-rata distribusi pembagian pengeluaran per kapita (20%) tinggi sebesar 50.29, dan rata-rata gini ratio sebesar 0.431.

Dari sini dapat dilihat bahwa untuk mengurutkan nilai dari tertinggi ke terendah ataupun sebaliknya berbeda-beda pada tiap variabelnya. Bisa jadi rata-rata distribusi pembagian pengeluaran per kapita (40%) rendah yang paling tinggi ada di kelompok 1 namun ternyata rata-rata gini ratio justru kelompok 1 yang paling kecil.

4. Non-Hierarcical Clustering (K-Means Clustering)

- Menampilkan 15 data teratas:

1	College Name	Apps	Accept	Enroll	F.Undergrad	P.Undergrad	Books	Personal
2	Abilene Christian University	1660	1232	721	2885	537	450	2200
3	Adelphi University	2186	1924	512	2683	1227	750	1500
4	Adrian College	1428	1097	336	1036	99	400	1165
5	Agnes Scott College	417	349	137	510	63	450	875
6	Alaska Pacific University	193	146	55	249	869	800	1500
7	Albertson College	587	479	158	678	41	500	675
8	Albertus Magnus College	353	340	103	416	230	500	1500
9	Albion College	1899	1720	489	1594	32	450	850
10	Albright College	1038	839	227	973	306	300	500
11	Alderson-Broaddus College	582	498	172	799	78	660	1800
12	Alfred University	1732	1425	472	1830	110	500	600
13	Allegheny College	2652	1900	484	1707	44	400	600
14	Allentown Coll. of St. Francis de Sales	1179	780	290	1130	638	600	1000
15	Alma College	1267	1080	385	1306	28	400	400

Data yang digunakan sama seperti saat melakukan analisis PCA yaitu menggunakan tujuh variable dengan skala yang berbeda-beda dan menghapus kolom “College Name” saat melakukan analisis. Total observasi adalah sebanyak 777 buah diwakili oleh nama kampus. Akan dilakukan pengelompokan menggunakan analisis cluster menggunakan non-hirarki clustering atau biasa disebut k-means clustering untuk melihat beberapa kelompok yang berbeda berdasarkan kemiripan antar data. Analisis ini berbeda dengan hirarki clustering dalam hal proses pembentukan kelompok atau cluster.

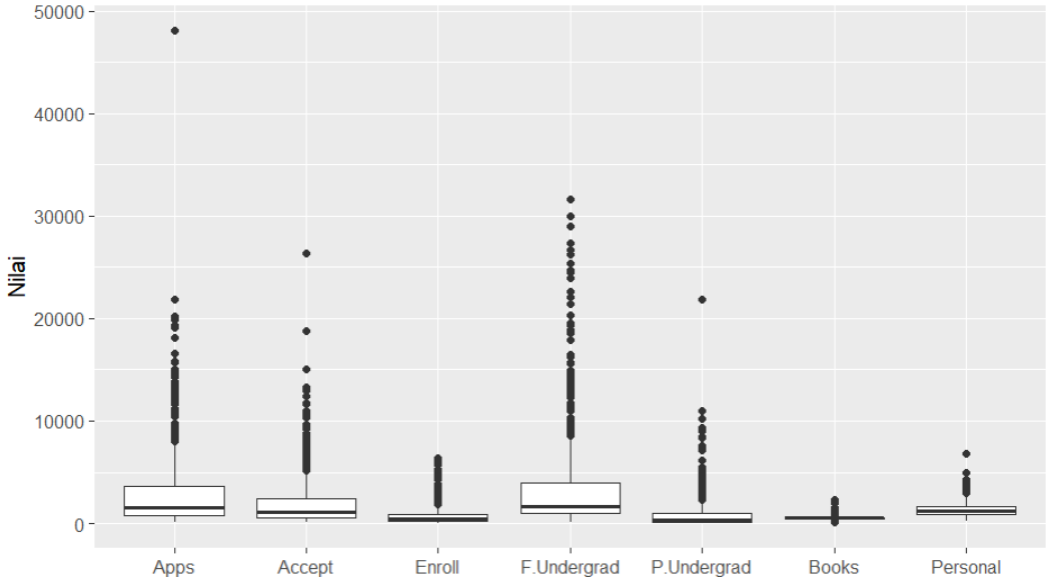
- Sebelum itu cek terlebih dahulu nilai null pada data

```
```{r Melihat banyak nilai null}
sum(is.na(dataku))
```
```

```
[1] 0
```

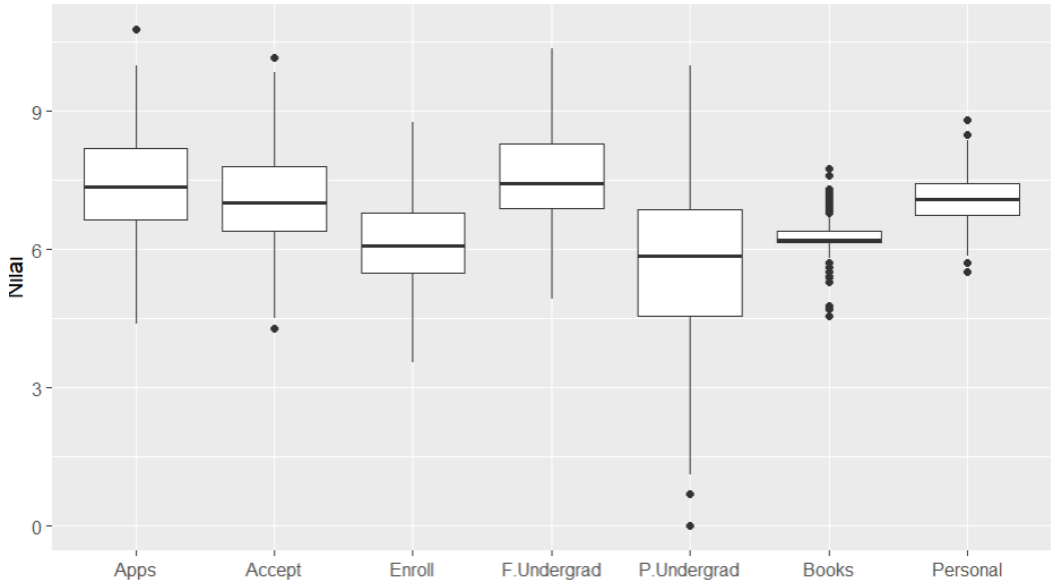
Setelah dicek dengan function *is.na* ternyata tidak ada nilai null. Maka lanjut ke analisis.

Lalu mengecek apakah ada outlier atau tidak dengan boxplot



Karena terlalu banyak outlier maka akan dilakukan transformasi log pada data.

```r Transformasi log pada data
data_trans <- log(dataku)
```



Setelah dilakukan transformasi, terlihat outlier berkurang secara signifikan walaupun masih ada beberapa. Namun akan dilanjutkan langsung ke analisis.

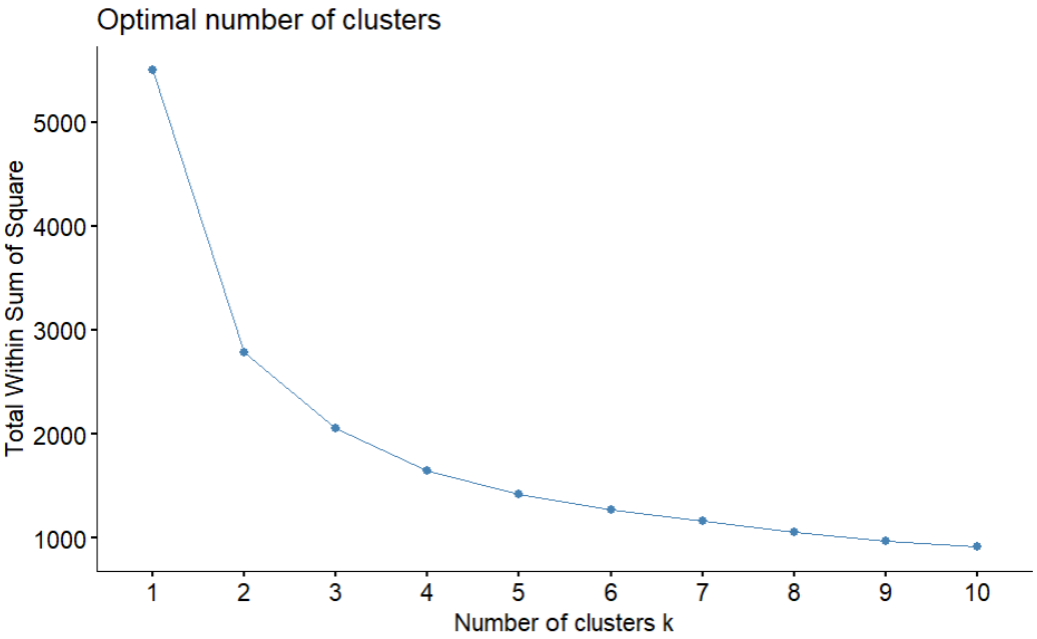
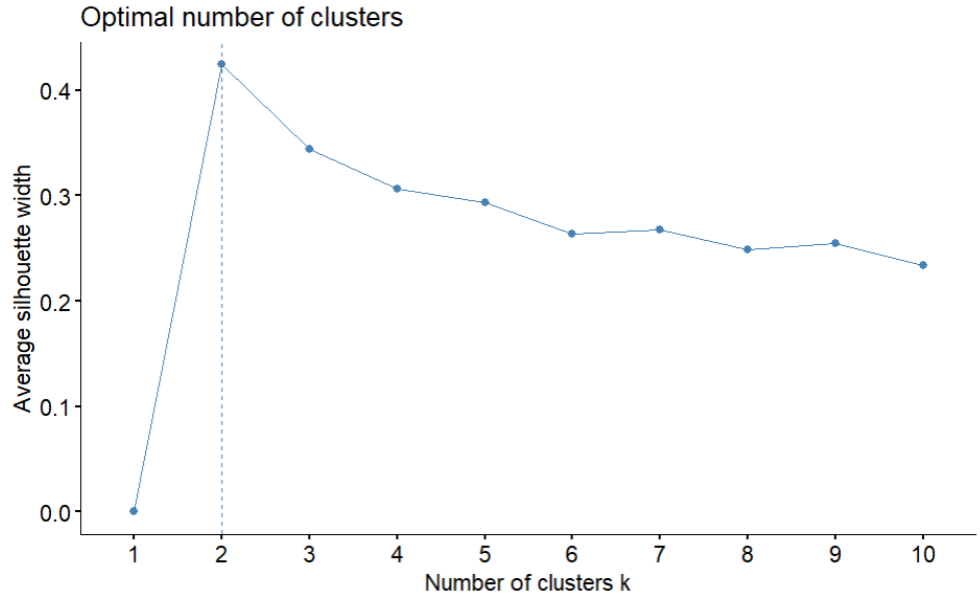
- Tahap pertama adalah menjadikan kolom "Tahun/Year" sebagai label/kelas untuk visualisasi nanti. Kemudian melakukan standarisasi data pada kolom ke 2 sampai ke 5 untuk menyamakan skala sehingga data akan diubah sedemikian sehingga memiliki rata-rata (mean) 0 dan standar deviasi 1 menggunakan fungsi *scale()*.

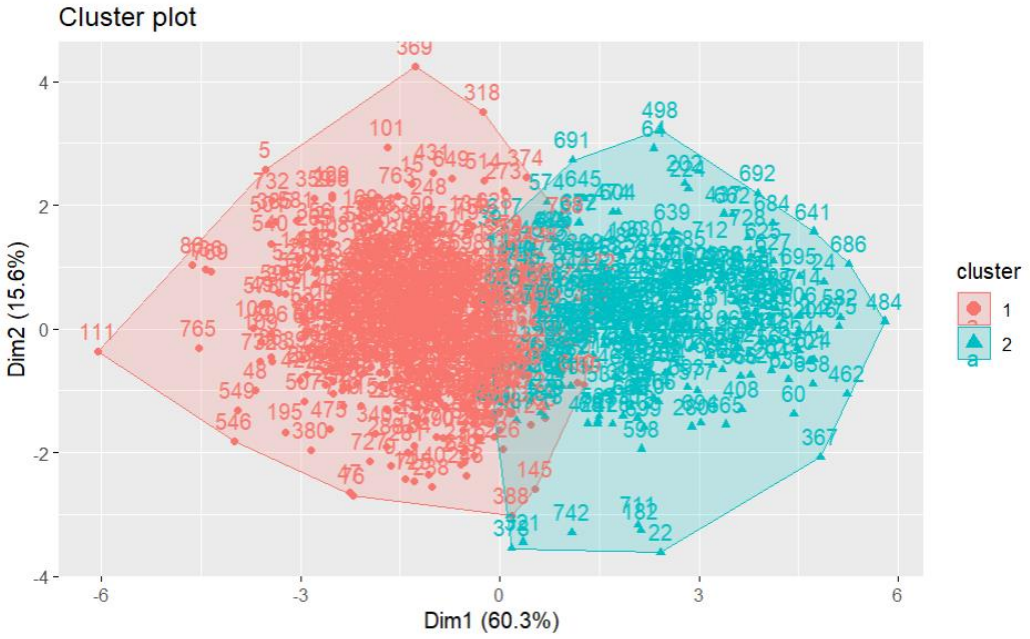
```r Standarisasi data
summary(dataku)
scale <- dataku %>%
  mutate(across(everything(), ~ scale(.) %>% as.vector))
round(scale,digits=3)
```

Apps	Accept	Enroll	F.Undergrad	P.Undergrad	Books	Personal
-0.347	-0.321	-0.063	-0.168	-0.209	-0.602	1.269
-0.211	-0.039	-0.288	-0.210	0.244	1.215	0.235
-0.407	-0.376	-0.478	-0.549	-0.497	-0.905	-0.259
-0.668	-0.681	-0.692	-0.658	-0.520	-0.602	-0.688
-0.726	-0.764	-0.780	-0.711	0.009	1.518	0.235
-0.624	-0.628	-0.669	-0.623	-0.535	-0.299	-0.983
-0.684	-0.685	-0.729	-0.677	-0.411	-0.299	0.235
-0.285	-0.122	-0.313	-0.434	-0.541	-0.602	-0.725
-0.507	-0.481	-0.595	-0.562	-0.361	-1.510	-1.242
-0.625	-0.620	-0.654	-0.598	-0.511	0.670	0.678

Ini adalah hasil dari standarsisasi data, menampilkan sebanyak 5 data teratas dan pembulatan 3 angka di belakang koma.

- Tahap kedua adalah mencari jumlah cluster optimal dengan menggunakan fungsi *fviz_nbclust*, digunakan untuk menampilkan grafik yang membantu menentukan jumlah cluster optimal berdasarkan Within Sum of Squares (WSS) dan metode Silhouette. WSS mengukur total variabilitas dalam cluster yang diukur sebagai jumlah kuadrat jarak setiap titik data ke pusat cluster terdekat. Tujuan dari grafik ini adalah untuk mencari titik di mana penurunan WSS mulai menurun secara signifikan (elbow point), yang menunjukkan jumlah cluster yang optimal. Metode Silhouette mengukur seberapa baik setiap titik data cocok dengan cluster mereka sendiri dibandingkan dengan cluster lainnya. Grafik ini membantu mengevaluasi kualitas cluster dengan mempertimbangkan seberapa baik setiap titik data sesuai dengan clusternya dan seberapa baik cluster tersebut terpisah satu sama lain.

```r
```{r Mencari cluster optimal}
fviz_nbclust(data_trans, kmeans, method="wss")
fviz_nbclust(data_trans, kmeans, method="silhouette")
final <- kmeans(data_trans, 2)
```
```



Sumbu x mewakili jumlah cluster (k), dan sumbu y mewakili WSS. Ketika k meningkat, WSS berkurang karena titik-titik data dibagi menjadi lebih banyak cluster, mengurangi varian dalam setiap cluster. Titik "elbow" adalah dimana laju penurunan WSS melambat secara signifikan. Ini menunjukkan bahwa menambah lebih banyak cluster di luar titik ini memberikan pengurangan varian yang semakin berkurang. Sehingga dalam grafik ini, titik elbow adalah pada saat penurunan nilai secara signifikan berhenti yaitu di k=2. Artinya akan digunakan cluster sebanyak 2 buah pada analisis ini.

- Dengan cara lain menentukan jumlah cluster optimal yaitu menggunakan metode Silhouette.



Sumbu x mewakili jumlah cluster (k), dan sumbu y mewakili lebar silhouette rata-rata. Lebar silhouette mengukur seberapa mirip suatu titik data dengan cluster-nya sendiri dibandingkan dengan cluster lain. Nilai berkisar dari -1 hingga 1, di mana nilai yang lebih tinggi menunjukkan clustering yang lebih baik. Saat k meningkat maka lebar silhouette rata-rata naik hingga mencapai puncak yang menunjukkan jumlah cluster optimal.
Setelah melewati puncak tersebut, lebar silhouette rata-rata cenderung menurun karena menambah lebih banyak cluster menyebabkan beberapa cluster mungkin menjadi terlalu kecil atau tidak bermakna. Sehingga dalam grafik ini, puncak ada pada k=2. Hasil ini sama seperti menggunakan metode elbow.

- Tahap ketiga adalah melakukan visualisasi menggunakan fungsi *fviz_cluster.* Dimensi terlebih dahulu direduksi menjadi hanya 2 dimensi untuk memudahkan dalam melakukan visualisasi.

```
```{r Mencari cluster optimal}
fviz_nbclust(data_trans, kmeans, method="wss")
fviz_nbclust(data_trans, kmeans, method="silhouette")
```
```



Sumbu x (Dim1) dan sumbu y (Dim2) mewakili dua dimensi utama yang menjelaskan variansi data terbesar. Dalam hal ini, Dim1 menjelaskan 60.3% dari total variansi, dan Dim2 menjelaskan 15.6% dari total variansi. Data dibagi menjadi dua cluster yatu cluster 1 (diwakili oleh warna merah dan simbol lingkaran) dan cluster 2 (diwakili oleh warna biru dan simbol segitiga). Data yang termasuk dalam cluster 1, sebagian besar terletak di sebelah kiri grafik dan memiliki rentang nilai yang lebih kecil di Dim1, tetapi lebih menyebar di Dim2. Data dalam cluster 2, sebagian besar terletak di sebelah kanan grafik, menunjukkan rentang nilai yang lebih luas di Dim1, tetapi rentang yang lebih terbatas di Dim2. Ada beberapa overlap antara cluster 1 dan cluster 2 di tengah-tengah grafik, menunjukkan beberapa data yang mungkin berada di batas antara dua cluster.

- Tahap keempat adalah menghitung jumlah data pada tiap cluster.

```{r Menghitung jumlah data dalam setiap clusterr}
cluster_counts <- table(final$cluster)
cluster_counts
```

```
  1   2
503 274
```

Terdapat 503 data pada cluster 1 dan 274 data pada cluster 2

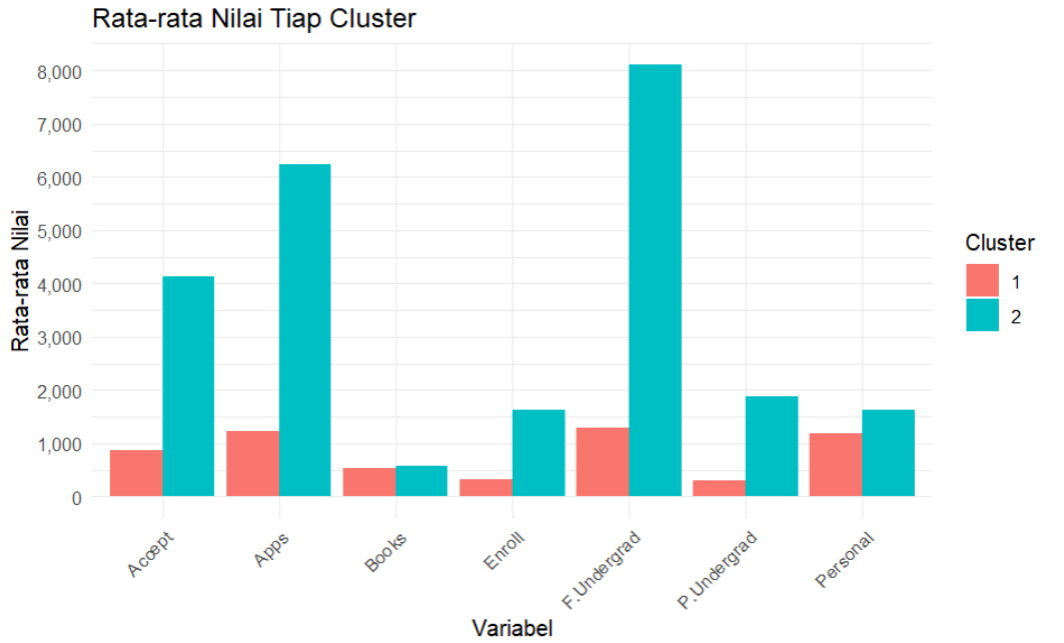
- Tahap terakhir adalah melihat karakteristik yang membedakan tiap cluster. Disini saya membedakannya dari rata-rata tiap kelompok pada tiap variabel.

```{r Rata-rata tiap cluster}
means <- dataku %>% mutate(cluster=final$cluster) %>%
  group_by(cluster) %>% summarise_all("mean")
means
```

A tibble: 2 × 8

cluster <int>	Apps <dbl>	Accept <dbl>	Enroll <dbl>	F.Undergrad <dbl>	P.Undergrad <dbl>	Books <dbl>	Personal <dbl>
1	1235.109	870.3996	321.8807	1301.986	296.9384	534.1630	1180.040
2	6244.573	4127.0073	1620.9234	8101.931	1880.3175	577.3175	1635.471

2 rows



Dari gambar terlihat bahwa tiap kelompok dibedakan dari nilai rata-rata pada variabel yang berbeda-beda. Dapat disimpulkan bahwa kelompok 2 memiliki rata-rata nilai yang lebih besar dibandingkan kelompok 1 dari segi variabel apapun.

5. Discriminant Analysis

- Menampilkan 15 data teratas:

	Total Bilirubin	Direct Bilirubin	Alkphos Alkaline Phosphotase	Sgpt Alamine Aminotransferase	Sgot Aspartate Aminotransferase	Total Protiens	ALB Albumin	Ratio Albumin and Globulin Ra	Result
2	0.7	0.1	187	16	18	6.8	3.3	0.9	1
3	10.9	5.5	699	64	100	7.5	3.2	0.74	1
4	7.3	4.1	490	60	68	7	3.3	0.89	1
5	1	0.4	182	14	20	6.8	3.4	1	1
6	3.9	2	195	27	59	7.3	2.4	0.4	1
7	1.8	0.7	208	19	14	7.6	4.4	1.3	1
8	0.9	0.3	202	14	11	6.7	3.6	1.1	1
9	0.9	0.3	202	22	19	7.4	4.1	1.2	2
10	0.7	0.2	290	53	58	6.8	3.4	1	1
11	0.6	0.1	210	51	59	5.9	2.7	0.8	1
12	2.7	1.3	260	31	56	7.4	3	0.6	1
13	0.9	0.3	310	61	58	7	3.4	0.9	2
14	1.1	0.4	214	22	30	8.1	4.1	1	1
15	0.7	0.2	145	53	41	5.8	2.7	0.87	1
16	0.6	0.1	183	91	53	5.5	2.3	0.7	2

Data memiliki 8 variabel predictor berbentuk numeric dan variabel predictor yaitu Result berbentuk kategorik dengan 2 kategori. Lalu akan dilakukan analisis diskriminan untuk dapat mengklasifikasikan data ke dalam dua kategori yaitu 1 sebagai penderita penyakit dan 2 bukan penderita.

- Sebelum itu cek terlebih dahulu nilai null pada data

```
```{r Menghitung nilai NA pada data}
sum(is.na(data[, 1:8]))

```
```

```
[1] 0
```

Setelah dicek dengan function *is.na* ternyata tidak ada nilai null. Maka lanjut ke analisis.

- Tahap pertama adalah memecah variable yang akan di standarisasi dan tidak. Variabel predictor dari kolom 1 sampai 8 akan dilakukan standarisasi untuk menyamakan skala sehingga data akan diubah sedemikian sehingga memiliki rata-rata (mean) 0 dan standar deviasi 1 menggunakan fungsi *scale()*. Untuk kolom 9 sebagai variabel target tidak akan di standarisasi karena akan merubah nilainya.

```
```{r Standarisasi data}
variables_to_scale <- dataku[,1:8]
variable_to_leave_unscaled <- dataku[,9]
scaled_variables <- scale(variables_to_scale)
head(round(scaled_variables,digits = 3))
```
```

```
     Total Bilirubin Direct Bilirubin  Alkphos Alkaline Phosphotase
[1,]          -0.400           -0.494                       -0.437
[2,]           1.330            1.624                        1.669
[3,]           0.719            1.075                        0.810
[4,]          -0.349           -0.376                       -0.457
[5,]           0.143            0.251                       -0.404
[6,]          -0.214           -0.259                       -0.350
     Sgpt Alamine Aminotransferase Sgot Aspartate Aminotransferase Total Protiens  ALB Albumin
[1,]                        -0.350                         -0.306          0.298        0.177
[2,]                        -0.101                         -0.039          0.941        0.051
[3,]                        -0.121                         -0.143          0.482        0.177
[4,]                        -0.360                         -0.300          0.298        0.303
[5,]                        -0.293                         -0.173          0.757       -0.959
[6,]                        -0.334                         -0.319          1.032        1.565
     A/G Ratio Albumin and Globulin Ratio
[1,]                               -0.178
[2,]                               -0.703
[3,]                               -0.211
[4,]                                0.150
[5,]                               -1.819
[6,]                                1.134
```

- Menggabungkan variabel yang di standarisasi dan yang tidak ke dalam satu data frame untuk dilakukan analisis diskriminan.

```{r Gabung variabel scaled dan unscaled ke satu data frame}
data <- data.frame(scaled_variables, variable_to_leave_unscaled)
round(data,digits = 3)
```

Description: df [1,000 × 9]

Total.Bilirubin <dbl>	Direct.Bilirubin <dbl>	X.Alkphos.Alkaline.Phosphotase <dbl>	X.Sgpt.Alamine.Aminotransferase <dbl>	Sgot.Aspartate.Aminotransferase <dbl>
-0.400	-0.494	-0.437	-0.350	-0.306
1.330	1.624	1.669	-0.101	-0.039
0.719	1.075	0.810	-0.121	-0.143
-0.349	-0.376	-0.457	-0.360	-0.300
0.143	0.251	-0.404	-0.293	-0.173
-0.214	-0.259	-0.350	-0.334	-0.319
-0.366	-0.455	-0.572	0.000	-0.326
-0.366	-0.415	-0.375	-0.360	-0.329
-0.366	-0.415	-0.375	-0.318	-0.303
-0.400	-0.455	-0.013	-0.158	-0.176

1-10 of 1,000 rows | 1-5 of 9 columns

Previous 1 2 3 4 5 6 ... 100 Next

Description: df [1,000 × 9]

X.Sgpt.Alamine.Aminotransferase <dbl>	Sgot.Aspartate.Aminotransferase <dbl>	Total.Protiens <dbl>	X.ALB.Albumin <dbl>	A.G.Ratio.Albumin.and.Globulin.Ratio <dbl>	Result <dbl>
-0.350	-0.306	0.298	0.177	-0.178	1
-0.101	-0.039	0.941	0.051	-0.703	1
-0.121	-0.143	0.482	0.177	-0.211	1
-0.360	-0.300	0.298	0.303	0.150	1
-0.293	-0.173	0.757	-0.959	-1.819	1
-0.334	-0.319	1.032	1.565	1.134	1
0.000	-0.326	0.482	0.429	0.150	1
-0.360	-0.329	0.207	0.555	0.478	1
-0.318	-0.303	0.849	1.187	0.806	2
-0.158	-0.176	0.298	0.303	0.150	1

1-10 of 1,000 rows | 4-9 of 9 columns

Previous 1 2 3 4 5 6 ... 100 Next

- Fungsi lda() digunakan untuk melakukan Analisis Diskriminan Linear (LDA) dengan variabel Result dan beberapa variabel predictor yaitu Total.Bilirubin, Direct.Bilirubin, X.Alkphos.Alkaline.Phosphotase, X.Sgpt.Alamine.Aminotransferase, got.Aspartate.Aminotransferase, Total.Protiens, X.ALB.Albumin, dan A.G.Ratio.Albumin.and.Globulin.Ratio.

```{r Model analisis diskriminan linear}
lda_model <- lda(data$Result ~ data$Total.Bilirubin+data$Direct.Bilirubin+data$X.Alkphos.Alkaline.Phosphotase+
                 data$X.Sgpt.Alamine.Aminotransferase+data$Sgot.Aspartate.Aminotransferase+data$Total.Protiens+data$X.ALB.Albumin+
                 data$A.G.Ratio.Albumin.and.Globulin.Ratio)

# Print the results
print(lda_model)
```

```
Call:
lda(data$Result ~ data$Total.Bilirubin + data$Direct.Bilirubin +
    data$X.Alkphos.Alkaline.Phosphotase + data$X.Sgpt.Alamine.Aminotransferase +
    data$Sgot.Aspartate.Aminotransferase + data$Total.Protiens +
    data$X.ALB.Albumin + data$A.G.Ratio.Albumin.and.Globulin.Ratio)

Prior probabilities of groups:
        1         2
0.5980392 0.4019608
```

Probabilitas awal untuk setiap kelompok:
Kelompok 1: 0.5980392
Kelompok 2: 0.4019608
Ini menunjukkan bahwa proporsi data dalam kelompok 1 adalah sekitar 59.8%, sedangkan dalam kelompok 2 adalah sekitar 40.2%.

```
Group means:
  data$Total.Bilirubin data$Direct.Bilirubin data$X.Alkphos.Alkaline.Phosphotase data$X.Sgpt.Alamine.Aminotransferase data$Sgot.Aspartate.Aminotransferase
1          -0.005451757            0.3320914                          0.2388238                           0.2047910                          0.1806204
2           0.019319451           -0.4940871                         -0.3553232                          -0.3046891                         -0.2687279
  data$Total.Protiens data$X.ALB.Albumin data$A.G.Ratio.Albumin.and.Globulin.Ratio
1         -0.02726322         -0.2053957                           -0.2327600
2          0.04056235          0.3055888                            0.3463015
```

Rata-rata setiap variabel prediktor untuk masing-masing kelompok menunjukkan perbedaan dalam nilai rata-rata setiap variabel antara dua kelompok. Misalnya, Direct.Bilirubin lebih tinggi pada kelompok 1 dibandingkan kelompok 2.

```
Coefficients of linear discriminants:
                                              LD1
data$Total.Bilirubin                     0.09728920
data$Direct.Bilirubin                   -0.43518625
data$X.Alkphos.Alkaline.Phosphotase     -0.28600096
data$X.Sgpt.Alamine.Aminotransferase    -0.39180702
data$Sgot.Aspartate.Aminotransferase     0.13190448
data$Total.Protiens                     -0.50008726
data$X.ALB.Albumin                       0.70847529
data$A.G.Ratio.Albumin.and.Globulin.Ratio -0.02752357
```

Koefisien-koefisien ini menunjukkan kontribusi masing-masing variabel prediktor terhadap fungsi diskriminan linear. Misalnya, X.ALB.Albumin memiliki kontribusi positif terbesar (0.70847529) terhadap LD1, sedangkan Total.Protiens memiliki kontribusi negatif terbesar (-0.5000876).

- Ini adalah hasil evaluasi menggunakan confusion matrix

```
```{r Analisis performa menggunakan confusion matrix}
fit.val <- predict(lda_model, data[, 1:8])
ct <- table(data$Result,fit.val$class)
ct
```
```

```
    1   2
1 337  90
2 103 184
```

Baris pertama dan kedua dari confusion matrix masing-masing mewakili kelas aktual 1 dan 2.
Kolom pertama dan kedua dari confusion matrix masing-masing mewakili kelas yang diprediksi 1 dan 2.

Akurasi (Accuracy):
Akurasi dapat dihitung sebagai total prediksi yang benar dibagi dengan total observasi.
Akurasi = (TP1 + TP2) / (Total Observasi)
Akurasi = (337 + 184) / (337 + 90 + 103 + 184) = 521 / 714 ≈ 0.729

Precision:
Precision untuk kelas 1 = TP1 / (TP1 + FP1) = 337 / (337 + 103) ≈ 0.766
Precision untuk kelas 2 = TP2 / (TP2 + FP2) = 184 / (184 + 90) ≈ 0.671

Recall (Sensitivity):
Recall untuk kelas 1 = TP1 / (TP1 + FN1) = 337 / (337 + 90) ≈ 0.789
Recall untuk kelas 2 = TP2 / (TP2 + FN2) = 184 / (184 + 103) ≈ 0.641

F1-Score:
F1-Score adalah harmonic mean dari Precision dan Recall.
F1-Score untuk kelas 1 = 2 * (Precision1 * Recall1) / (Precision1 + Recall1) ≈ 2 * (0.766 * 0.789) / (0.766 + 0.789) ≈ 0.777
F1-Score untuk kelas 2 = 2 * (Precision2 * Recall2) / (Precision2 + Recall2) ≈ 2 * (0.671 * 0.641) / (0.671 + 0.641) ≈ 0.656

- Kemudian melihat nilai akurasi pada tiap kategori dan total akurasinya.

```{r Performa keseluruhan model}
diag(prop.table(ct,1))
sum(diag(prop.table(ct)))
```

```
        1         2
0.7892272 0.6411150
[1] 0.7296919
```

Confusion matrix menunjukkan bahwa model LDA memiliki akurasi sekitar 71.8%. Model ini cenderung lebih baik dalam memprediksi kelas 1 (dengan precision dan recall yang lebih tinggi) dibandingkan dengan kelas 2. Tingkat kesalahan false positive dan false negative menunjukkan bahwa ada ruang untuk peningkatan, terutama dalam memprediksi kelas 2.

6. Singular Value Decomposition

- Menampilkan 15 data teratas:

1	College Name	Apps	Accept	Enroll	F.Undergrad	P.Undergrad	Books	Personal
2	Abilene Christian University	1660	1232	721	2885	537	450	2200
3	Adelphi University	2186	1924	512	2683	1227	750	1500
4	Adrian College	1428	1097	336	1036	99	400	1165
5	Agnes Scott College	417	349	137	510	63	450	875
6	Alaska Pacific University	193	146	55	249	869	800	1500
7	Albertson College	587	479	158	678	41	500	675
8	Albertus Magnus College	353	340	103	416	230	500	1500
9	Albion College	1899	1720	489	1594	32	450	850
10	Albright College	1038	839	227	973	306	300	500
11	Alderson-Broaddus College	582	498	172	799	78	660	1800
12	Alfred University	1732	1425	472	1830	110	500	600
13	Allegheny College	2652	1900	484	1707	44	400	600
14	Allentown Coll. of St. Francis de Sales	1179	780	290	1130	638	600	1000
15	Alma College	1267	1080	385	1306	28	400	400
16	Alverno College	494	313	157	1317	1235	650	2449

Data yang digunakan sama seperti saat melakukan analisis PCA yaitu menggunakan tujuh variable dengan skala yang berbeda-beda dan menghapus kolom “College Name” saat melakukan analisis. Total observasi adalah sebanyak 777 buah diwakili oleh nama kampus. Akan dilakukan pengelompokan menggunakan analisis cluster menggunakan non-hirarki clustering atau biasa disebut k-means clustering untuk melihat beberapa kelompok yang berbeda berdasarkan kemiripan antar data. Analisis ini berbeda dengan hirarki clustering dalam hal proses pembentukan kelompok atau cluster.

- Data dikonversi ke dalam format matriks dan dibulatkan ke tiga desimal untuk tujuan tampilan.

```{r Jadikan matriks}
data_matrix <- as.matrix(data)
head(round(data_matrix,digits=3))
```

```
      Apps Accept Enroll F.Undergrad P.Undergrad Books Personal
[1,] 1660   1232    721        2885         537   450     2200
[2,] 2186   1924    512        2683        1227   750     1500
[3,] 1428   1097    336        1036          99   400     1165
[4,]  417    349    137         510          63   450      875
[5,]  193    146     55         249         869   800     1500
[6,]  587    479    158         678          41   500      675
```

- Singular Value Decomposition dilakukan pada matriks data lalu mendekomposisi matriks tersebut menjadi tiga matriks yaitu u, d, dan v. Komponen-komponen disimpan di dalam variabel *svd_result* dan hasil ukuran SVD dilihat.

```{r Menerapkan svd}
svd_result <- svd(data_matrix)
u <- svd_result$u
d <- svd_result$d
v <- svd_result$v
View(svd_result)
```

svd_result	list [3]	List of length 3
d	double [7]	235158 54875 35750 29957 17235 7026 ...
u	double [777 x 7]	-0.016486 -0.018185 -0.009131 -0.003715 -0.002900 -0.004731 0.011862 0.002540 ...
v	double [7 x 7]	-0.555678 -0.366216 -0.140689 -0.707968 -0.140584 -0.043947 -0.697444 -0.243263 ...

Untuk ukuran matriks d adalah 1x7, matriks u adalah 777x7, dan matriks v adalah 7x7.

- Menampilkan nilai teratas dari matriks orthogonal kiri atau matriks u:

```{r Matriks orthogonal kiri}
cat("\nMatriks orthogonal kiri\n")
head(u)
```

```
Matriks orthogonal kiri
              [,1]          [,2]        [,3]       [,4]         [,5]         [,6]         [,7]
[1,] -0.016485971  0.0118624074 -0.02756958 0.04640674 -0.003922234  0.034928818 -0.028326931
[2,] -0.018184737  0.0025401894 -0.03463575 0.01471831  0.023752412 -0.026303552  0.027522712
[3,] -0.009131203 -0.0090076268 -0.01850975 0.02637718  0.011213813  0.002631372 -0.004791022
[4,] -0.003714533  0.0006497649 -0.01497972 0.02298813  0.001559744 -0.019926773  0.005667185
[5,] -0.002900463  0.0073972900 -0.04470393 0.02355857 -0.001804957 -0.039895507  0.014948796
[6,] -0.004731168 -0.0007737371 -0.01043539 0.01889416  0.003505697 -0.034648707  0.011135620
```

- Lalu menampilkan nilai teratas dari vektor nilai singular atau vektor u:

```{r Vektor nilai singular}
cat("\nVektor nilai singular\n")
d
```

```
Vektor nilai singular
[1] 235157.928  54875.296  35750.149  29957.397  17234.629   7026.430   5583.858
```

- Lalu menampilkan nilai teratas dari matriks orthogonal kanan atau vektor v:

```{r Matriks orthogonal kanan}
cat("\nMatriks orthogonal kanan\n")
head(v)
```

```
Matriks orthogonal kanan
            [,1]          [,2]         [,3]        [,4]        [,5]         [,6]        [,7]
[1,] -0.55567761 -0.6974440314 -0.113492858 -0.13305429 -0.41683165  0.005756101 -0.02069760
[2,] -0.36621611 -0.2432625071 -0.009426061  0.01944035  0.88584242  0.061950261  0.13299517
[3,] -0.14068884  0.0547789651  0.040841570  0.05505974  0.11279065 -0.168668274 -0.96505730
[4,] -0.70796789  0.5817665158  0.336402795  0.07449590 -0.15171153  0.003856236  0.13631380
[5,] -0.14058401  0.3211726763 -0.683499628 -0.63850103  0.04071007 -0.013920694 -0.01943832
[6,] -0.04394658  0.0008328463 -0.202274993  0.24241646  0.01328500 -0.931247496  0.17603608
```

- Matriks asli direkonstruksi dari komponen SVD dan beberapa baris awal dari matriks yang direkonstruksi dengan mengalikan lagi matriks u, d, dan v lalu ditampilkan sebagai perbandingan.

```{r Mengonstruksi ulang matriks asli}
reconstructed_matrix <- u %*% diag(d) %*% t(v)
head(reconstructed_matrix)
```

```
     [,1] [,2] [,3] [,4] [,5] [,6] [,7]
[1,] 1660 1232  721 2885  537  450 2200
[2,] 2186 1924  512 2683 1227  750 1500
[3,] 1428 1097  336 1036   99  400 1165
[4,]  417  349  137  510   63  450  875
[5,]  193  146   55  249  869  800 1500
[6,]  587  479  158  678   41  500  675
```

Dapat dilihat jika dibandingkan dengan matriks awal memiliki nilai yang sama.

- Melihat ukuran data awal dan data hasil rekonstruksi

```{r Melihat dimensi data}
dim(data)
dim(reconstructed_matrix)
```

```
[1] 777   7
[1] 777   7
```

Disini terlihat ukuran dari matriks juga sudah sama yang berarti data berhasil dilakukan dekomposisi menggunakan SVD.