Principele componenten analyse (PCA): Heart disease dataset

Rosa de Haan (rosa.de.haan@student.nhlstenden.com)
Lars Rotgers (lars.rotgers@student.nhlstenden.com)

4 juni 2019

Contents

1 Inleiding

In het voorgaande onderzoek is er gekeken met meervoudige regressie, en logistische regressie, of er verbanden zijn te vinden binnen de dataset.

Met de meervoudige regressie was het lastig om een goed model op te stellen. De oorzaak hiervan is dat er bijna geen onderlinge correlatie tussen de variabelen is te vinden. Het doel was om `thalanch` te schatten a.d.h.v. de overige ratio variabelen. Het best verkregen model hiervoor had een R^2 van 0.229, wat in onze ogen onbruikbaar is.

Met de tweede opdracht, logistische regressie, is er gekeken of er een verband bestaat tussen `thalach` (maximaal behaalde hartslag) and `exang` (optreden van borstpijn tijdens de oefening). Hiervoor is een logistisch regressiemodel opgesteld en is een correlatiecoëfficiënt van 0.9 gevonden, wat aantoont dat er een sterk positief verband is tussen de variabelen.

In de laatste opdracht, de principele componenten analyse, wordt er gekeken welke variabelen invloed hebben om de `target` kolom (wel/geen hartziekte). Op deze manier kan er misschien worden bepaald welke variabelen invloed hebben of op iemand een hartziekte heeft.

2 Data inlezen

Om te beginnen met de principele componenten analyse (PCA), worden eerst de gegevens ingeladen. Vervolgens wordt de kolomnaam voor `age` hersteld.

```
In [1]: df = read.csv('heart.csv')

        # kolomnaam herstellen; er staat '`..age'
        names = colnames(df);
        names[1] = 'age'
        colnames(df) = names

        head(df)
        print(paste('Er zijn', nrow(df), 'rijen, en', length(df), ' kolommen.'))
```

age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
57	1	0	140	192	0	1	148	0	0.4	1	0	1	1

```
[1] "Er zijn 303 rijen, en 14  kolommen."
```

Er is te zien dat er 303 rijen (aantal samples, $N = 303$) zijn en 14 kolommen (aantal variabelen, $v = 14$) in de dataset zitten.

Omdat er voorspeld moet worden of iemand een hartziekte heeft, wordt de kolom `target` niet in het model opgenomen. Echter wordt `target` wel gebruikt als groepering voor de samples. Alle binaire variabelen worden wel opgenomen in het model.

Tot slot worden er alvast een aantal functies gedeclareerd, die later gebruikt worden.

```
In [2]: # Calculate the trace of a matrix M.
        trace = function(M) { (sum(diag(M))) }

        # Calculate the variance of a matrix M.
        matrix.var = function(M) { (trace(t(M) %*% M) / (ncol(M) * nrow(M))) }

        # Calculate the standard deviation of a matrix M.
        matrix.sd  = function(M) { (sqrt(matrix.var(M))) }

        # Calculate the mean for every row of a matrix M.
        rowMeans = function(M) { (rowSums(M) / ncol(M)) }

        # Calculate the mean for every column of a matrix M.
        colMeans = function(M) { (colSums(M) / nrow(M)) }
```

3 Beschrijving van de kolommen

Op de website waar de dataset is verkregen, is de volgende informatie gevonden over de kolommen. De dataset bevat in totaal 14 kolommen.

1. `age`: leeftijd. (Ratio)
2. `sex`: geslacht. (Nominaal)
3. `cp`: chest pain type (4 values). (Nominaal)
4. `trestbps`: resting blood pressure. (Ratio)
5. `chol`: serum cholestoral in mg/dl. (Ratio)
6. `fbs`: fasting blood sugar > 120 mg/dl. (Nominaal)
7. `restecg`: resting electrocardiographic results (values 0, 1, 2). (Nominaal)
8. `thalanch`: maximum heartrate achieved. (Ratio)
9. `exang`: exercise induced angina. (Nominaal)
10. `oldpeak`: ST depression induced by exercise relative to test. (Ratio)
11. `slope`: the slope of the peak exercise ST segment. (Nominaal)
12. `ca`: number of major vessels (0-3) color by flourosopy. (Ordinaal)
13. `thal`: 3 = normal, 6 = fixed defect, 7 = reversable defect. (Nominaal)
14. `target`: indicated if someone has heart disease, 0 = false, 1 = true. (Nominaal)

Van de meeste variabelen is het niet echt duidelijk waar het voor staat. Neem bijvoorbeeld `oldpeak`, in dit geval is `ST depression` een fenomeen dat voorkomt in een ECG. Een ECG is een electrocardiogram, een grafiek van de electrische activiteit van het hart. In dit geval wordt met `ST depression` een bepaalt patroon in de grafiek bedoeld. [1] Een aantal van deze patronen kunnen duiden op een hartziekte. [2]

Een andere variabele is `exang`, wat aangeeft of er angina is voorkomen tijdens de oefening. Een angina is een pijn of oncomfortabelheid in de borst, mogelijk veroorzaakt doordat er te weinig zuurstof-rijk bloed bij die spier komt. Angina is echter geen ziekte, maar mogelijk een symptoom van een onderliggend hartprobleem. [3]

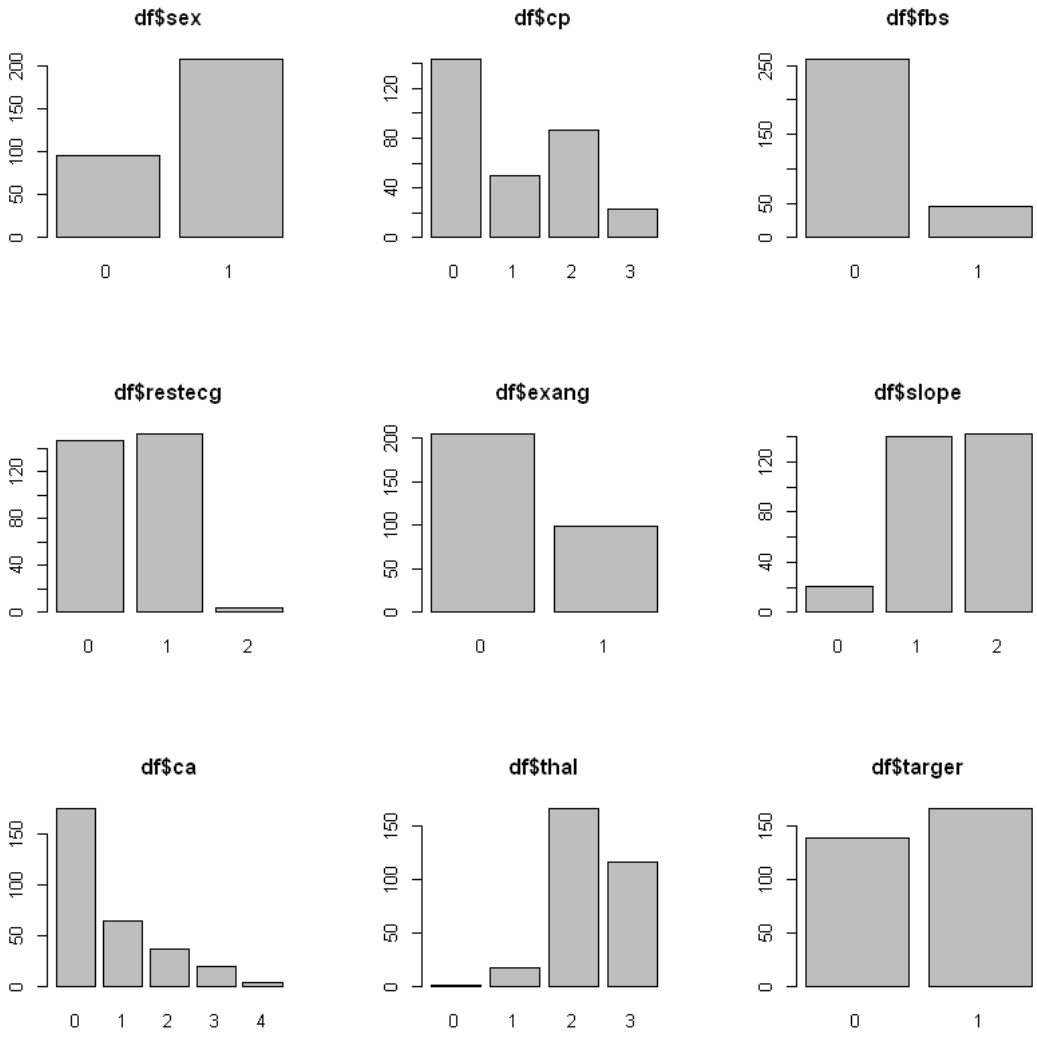
4 Verdelingen van de gegevens

In deze paragraaf wordt er gekeken naar de verdelingen van de gegeven binnen de verschillende variabelen. Indien de verdelingen sterk qua schaal verschillen, is het wenselijk om de gegevens te standaardiseren. Op deze manier wordt er voorkomen, dat de variabele met een grote schaal, zwaarder worden meegeteld binnen het model.

4.1 Nominale/ordinale variabelen

In het onderstaande figuur staan staafdiagrammen voor elk van de nominale/ordinale variabelen.

```
In [3]: par(mfrow=c(3,3))
        barplot(table(df$sex), main="df$sex")
        barplot(table(df$cp), main="df$cp")
        barplot(table(df$fbs), main="df$fbs")
        barplot(table(df$restecg), main="df$restecg")
        barplot(table(df$exang), main="df$exang")
        barplot(table(df$slope), main="df$slope")
        barplot(table(df$ca), main="df$ca")
        barplot(table(df$thal), main="df$thal")
        barplot(table(df$target), main="df$targer")
```

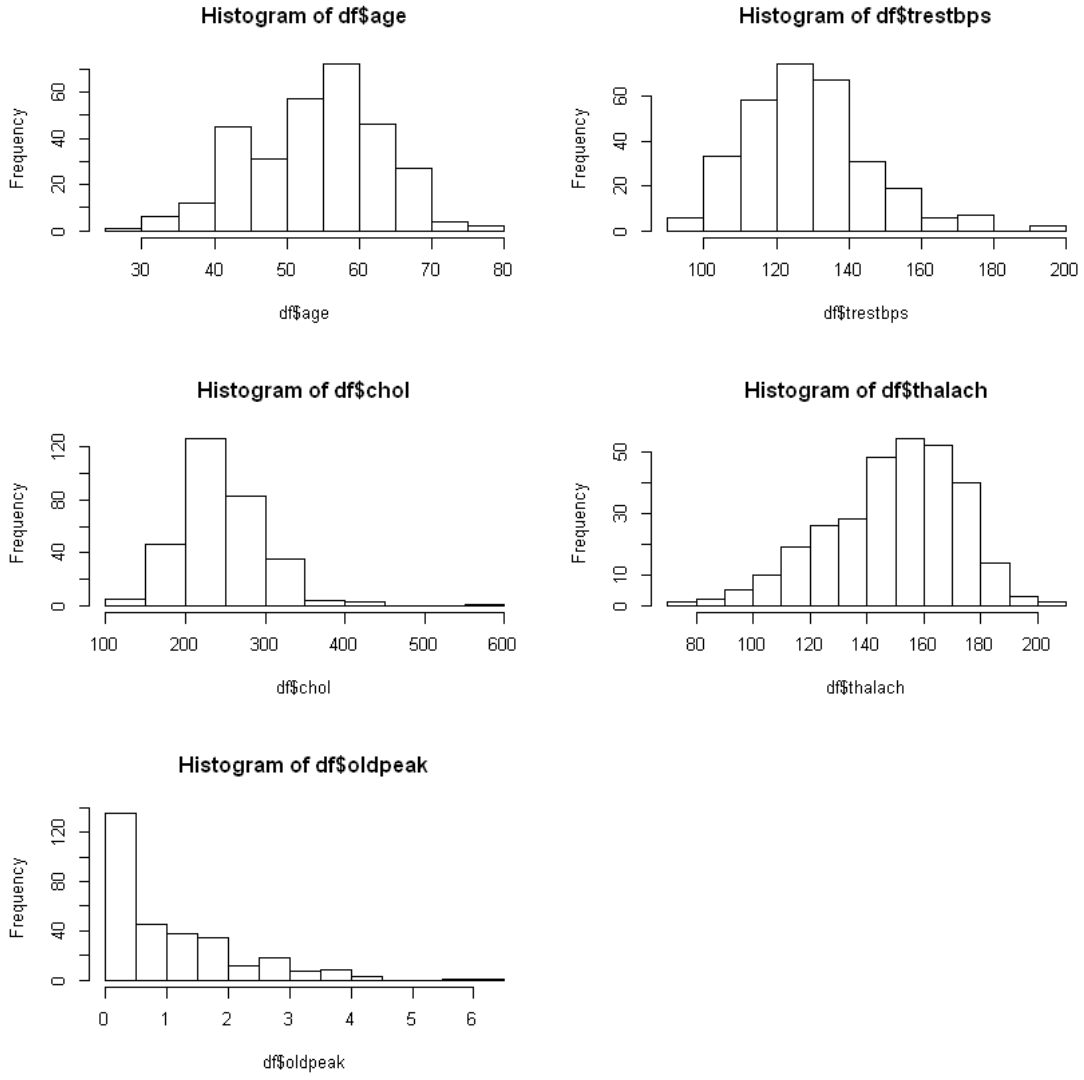


Hier valt het op dat er een combinatie is van binaire variabelen, en variabelen met meerdere schaalpunten. Als er niet gestandaardiseerd wordt, dan worden de variabelen met meerdere groepen zwaarder meegeteld in het model.

4.2 Ratio variabelen

In het onderstaande figuur staan histogrammen van de interval/ratio variabelen.

```
In [4]: par(mfrow=c(3, 2))
        hist(df$age)
        hist(df$trestbps)
        hist(df$chol)
        hist(df$thalach)
        hist(df$oldpeak)
```



Het valt op dat de schaal waarop de waarnemingen zijn gemeten sterk verschillen. Vanuit dit oogpunt is het verstandig om de gegevens te standaardiseren.

5 Centreren en standaardiseren

Vanuit de vorige paragraaf is gebleken dat het verstandig is om de gegevens, naast het centreren, ook te standaardiseren. Het centreren en standaardiseren, wordt ook wel normalizeren genoemd. Hiervoor wordt de z-score berekend. De formule hiervoor is $z = \frac{x-\bar{x}}{s}$, waarbij $\bar{x}$ het gemiddelde is en s de standaardafwijking.

Eerst wordt de matrix X opgesteld met de gegevens die relevant zijn. Daarna worden de gegevens gecentreerd, aangeduidt met X_c. Door de gegevens te centreren, worden alle hieropvolgende berekeningen vereenvoudigd. Daarnaast worden de gegevens gestandaardiseerd, aangeduidt met X_{cs}, zodat elke variabele evenzwaar meetelt binnen het model.

```
In [5]: X = data.matrix(df[,1:13]) # alle gegevens behalve de 'target' kolom
        m = apply(X,2,mean) # gemiddelde berekenen, voor elke kolom
        s = apply(X,2,sd) # standaard deviatie berekenen, voor elke kolom
        Xc = sweep(X,2,m,"-") # centreren
        Xcs = sweep(Xc,2,s,"/") # standaardiseren
```

In de onderstaande tabel staat een klein voorbeeld van hoe de gegevens van X_{cs} er nu uitzien. Voor elke kolom is nu het gemiddelde 0.

```
In [6]: head(Xcs[,1:5])
```

age	sex	cp	trestbps	chol
0.9506240	0.6798805	1.96986425	0.76269408	-0.25591036
-1.9121497	0.6798805	1.00092128	-0.09258463	0.07208025
-1.4717230	-1.4659924	0.03197832	-0.09258463	-0.81542377
0.1798773	0.6798805	0.03197832	-0.66277043	-0.19802967
0.2899839	-1.4659924	-0.93696465	-0.66277043	2.07861109
0.2899839	0.6798805	-0.93696465	0.47760118	-1.04694656

6 Berekenen van principele componenten

In deze paragraaf worden de principele componenten bepaald voor de gestandaardiseerde gegevens. Het eerste principele component PC_1, is de eigenvector $p_1' = [\beta_1, \beta_2, \ldots, \beta_{13}]$ met de hoogste eigenwaarde van de covariantie matrix van X_{cs}. De covariantiematrix wordt als volgt bepaald:

$$\text{cov}(\mathbf{X}) = \frac{1}{N-1}\mathbf{X}^T\mathbf{X},$$

maar gelukkig kan dit ook in R met `cov(X)`. Vervolgens wordt van de covariantiematrix van X_{cs}, de eigenvectoren en eigenwaarden bepaald. Om de eigenvectoren en eigenwaarden te vinden, is er de functie `eigen(X)`. De vectoren worden opgeslagen in `eigen(X)$vectors` en staan al in aflopende volgorde.

```
In [7]: p = eigen(cor(Xcs))$vectors # eigenvectoren van de covariantiematrix van Xcs
                                   # bepalen, dit zijn ook direct de PCAs

        colnames(p) = c('PC1', 'PC2', 'PC3', 'PC4', 'PC5', 'PC6', 'PC7',
                        'PC8', 'PC9', 'PC10', 'PC11', 'PC12', 'PC13')

        rownames(p) = colnames(df[,1:13])

        p[,1:6] # eerste x PCAs weergeven
        paste('Totaal:', ncol(p), 'kolommen')
```

	PC1	PC2	PC3	PC4	PC5	PC6
age	0.31420252	-0.40614872	-0.09407661	0.02066180	0.30715312	-0.12829615
sex	0.09083783	0.37779171	0.55484915	0.25530873	-0.05070440	0.05496875
cp	-0.27460749	-0.29726609	0.35697431	-0.28790041	-0.16317945	-0.19341117
trestbps	0.18392019	-0.43818675	0.20384930	-0.02260103	-0.18813809	-0.17945982
chol	0.11737503	-0.36451402	-0.40782498	0.34340982	-0.32006670	-0.10472957
fbs	0.07363999	-0.31743328	0.48173624	0.06860532	0.23344184	0.24961364
restecg	-0.12772792	0.22088181	-0.08919083	-0.26609555	0.39366727	-0.66681339
thalach	-0.41649811	-0.07787618	0.15825529	0.18412539	-0.32328431	-0.12098445
exang	0.36126745	0.26311790	-0.12635610	0.11505621	-0.03453568	0.23069914
oldpeak	0.41963899	0.05225497	0.11034290	-0.32629597	-0.25057927	-0.17007984
slope	-0.37977222	-0.04837415	-0.07381839	0.49484894	0.24682275	-0.06406935
ca	0.27326172	-0.09414721	0.18356934	0.32801632	0.43536515	-0.18210750
thal	0.22202375	0.20072042	0.12501113	0.38919138	-0.33195049	-0.50885654

'Totaal: 13 kolommen'

Er is te zien dat er in totaal 13 principele componenten zijn, namelijk het aantal variabelen binnen de gegevens. Vanuit de tabel kan de vector voor PC_1 worden afgelezen, namelijk:

$$p_1 = \begin{bmatrix} 0.314 \\ 0.090 \\ -0.274 \\ 0.183 \\ 0.117 \\ 0.073 \\ -0.127 \\ -0.416 \\ 0.361 \\ 0.419 \\ -0.379 \\ 0.273 \\ 0.222 \end{bmatrix}.$$

Alle elementen in p_1 worden ook wel *loadings* genoemd. Aan de loadings valt te zien dat de waarde 0.419 de hoogste is. Dit betekend dat `oldpeak` het meeste invloed heeft op PC_1. Daarnaast heeft `thalach` de hoogste negatieve invloed op PC_1, met een waarde van -0.416. Om te controleren of dit de juiste vector voor PC_1 is, kan er met R worden gekeken met `prcomp(...)`:

```
In [8]: data.matrix(prcomp(df[,1:13], scale=T)$rotation[,1]) # PCA1
        # data.matrix wordt gebruikt voor de opmaak hieronder.
```

age	-0.31420252
sex	-0.09083783
cp	0.27460749
trestbps	-0.18392019
chol	-0.11737503
fbs	-0.07363999
restecg	0.12772792
thalach	0.41649811
exang	-0.36126745
oldpeak	-0.41963899
slope	0.37977222
ca	-0.27326172
thal	-0.22202375

Dit komt overeen met de eigenvector met de grootste eigenwaarde van de covariantiematrix van X_{cs}, alleen het teken is omgedraaid. In de volgende paragraaf wordt hiervoor een oplossing besproken.

7 Rotatieambivalentie

Omdat eigenvectoren allemaal loodrecht op elkaar staan, zijn er altijd twee vectoren mogelijk. Om dezelfde resultaten te krijgen als `prcomp(...)`, is het nodig om alle eigenvectoren om te keren. Dit concept wordt *rotatieambivalentie* genoemdt. Dit is eenvoudig op te lossen met $p := -p$.

```
In [9]: p = -p
```

8 Biplot: score plot + loading plot

Om een bi-plot te maken, zijn er twee grafieken nodig. Er is een score plot tussen PC_1 en PC_2, en een loading plot.

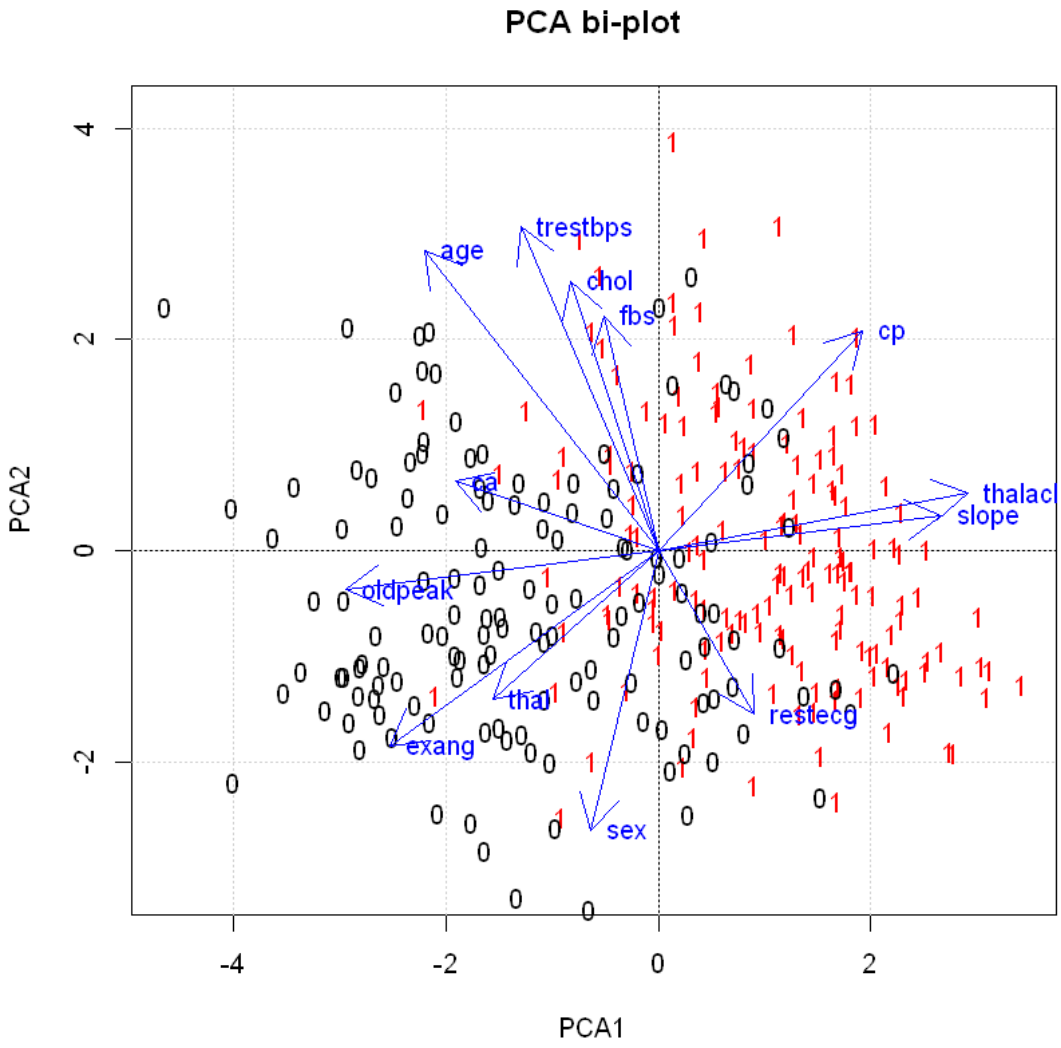
Om de scores t_1 voor PC_1 te bepalen, worden alle waarnemingen loodrecht op p_1 geprojecteerd. De afstand vanaf de oorsprong O tot aan de loodrechte projectie op p_1 is het nieuwe x-coordinaat van de waarneming in het p_1, p_2 vlak. Ditzelfde wordt gedaan om de scores t_2 te bepalen voor p_2, en dient als y-coordinaat.

```
In [10]: t1 = Xcs %*% p[,1] # projectie op p1
         t2 = Xcs %*% p[,2] # projectie op p2
         t = Xcs %*% p # voor alle p
```

Als deze nieuwe scores in een grafiek worden weergegeven, heet dit een score plot. De tweede grafiek, de loading plot, laat zien hoe de loadings van PC_1 en PC_2 bepalen waar de waarnemingen in de score plot belanden. De loadings mogen geschaald worden voor een weergave in een bi-plot. Het gaat is immers de onderline verhouding die belangrijk is. Met de onderstaande code, worden beide grafieken geplot in een bi-plot:

```
In [11]: # score plot
         plot(t1, t2, pch=NA, xlab="PCA1", ylab="PCA2" )
         abline(h=0,lty=1)
         abline(v=0,lty=1)
```

```
title(main='PCA bi-plot')
grid()

# zet de groepnamen in de plot + rood markeren als target=1
text(t1, t2, labels=df$target, pos=1, xpd=NA, col=c('black', 'red')[df$target+1])

# loading plot
scale = 7
arrows(0,0,p[,1] * scale, p[,2] * scale, col='blue', lwd=1)
text(p[,1] * scale, p[,2] * scale, labels=colnames(Xcs), col='blue', pos=4)
```

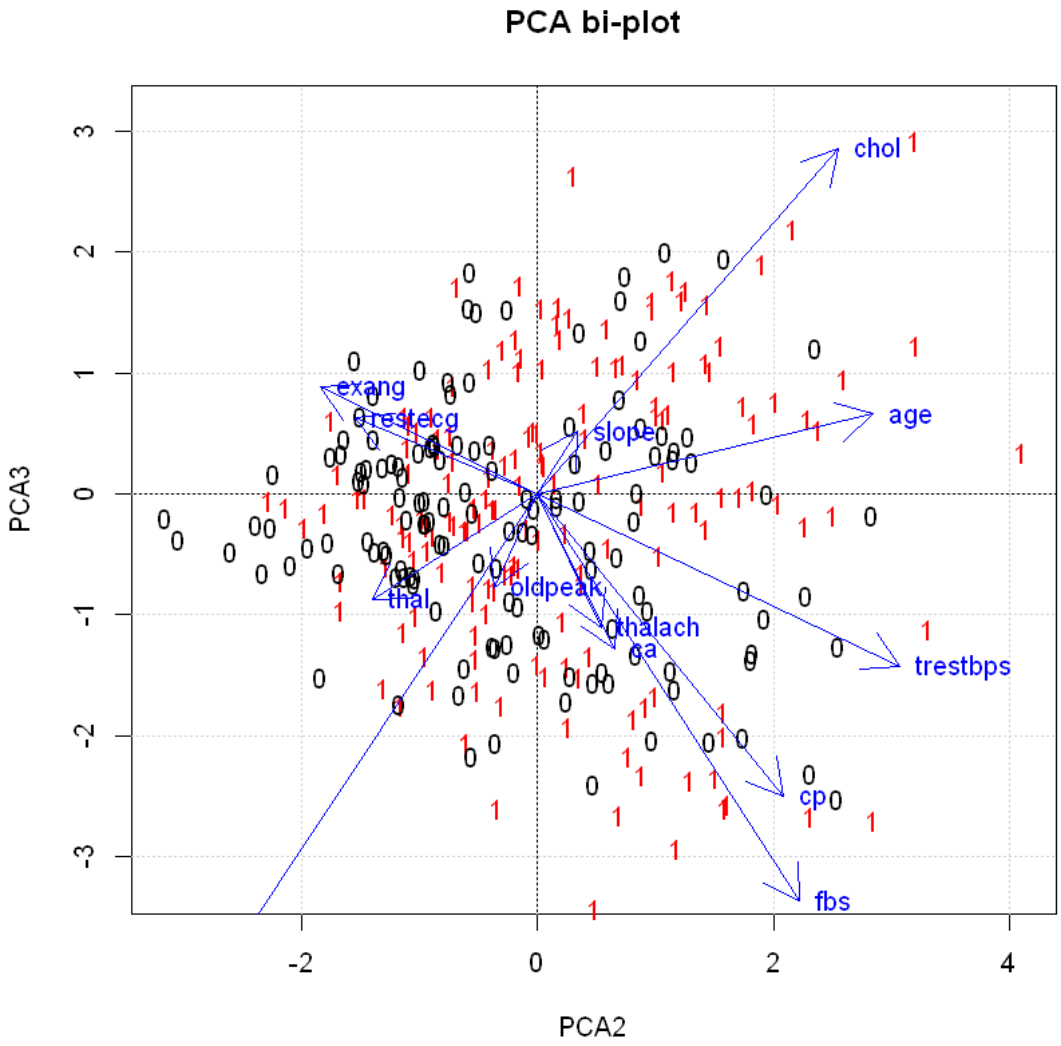


Wat direct opvalt is dat bijna alle waarnemingen die een hartziekte hebben, een positieve waarde hebben voor PC_1. Eerder bij de loadings werdt al duidelijk dat `oldpeak` en `thalach` het meeste invloed uit oefenenen, en ook hier valt te zien richting hiervan bijna horizontaal is. Twee

andere variabelen die ook een grote loading hebben zijn `slope` en `exang`, maar dit zijn nominale variabelen.

Ditzelfde kan ook gedaan worden voor PC_2 en PC_3:

```
In [12]: # score plot
         plot(t[,2], t[,3], pch=NA, xlab="PCA2", ylab="PCA3" )
         abline(h=0,lty=1)
         abline(v=0,lty=1)
         title(main='PCA bi-plot')
         grid()

         # zet de groepnamen in de plot + rood markeren als target=1
         text(t[,2], t[,3], labels=df$target, pos=1,
              xpd=NA, col=c('black', 'red')[df$target+1])

         # loading plot
         scale = 7
         arrows(0,0,p[,2] * scale, p[,3] * scale, col='blue', lwd=1)
         text(p[,2] * scale, p[,3] * scale, labels=colnames(Xcs),
              col='blue', pos=4)
```

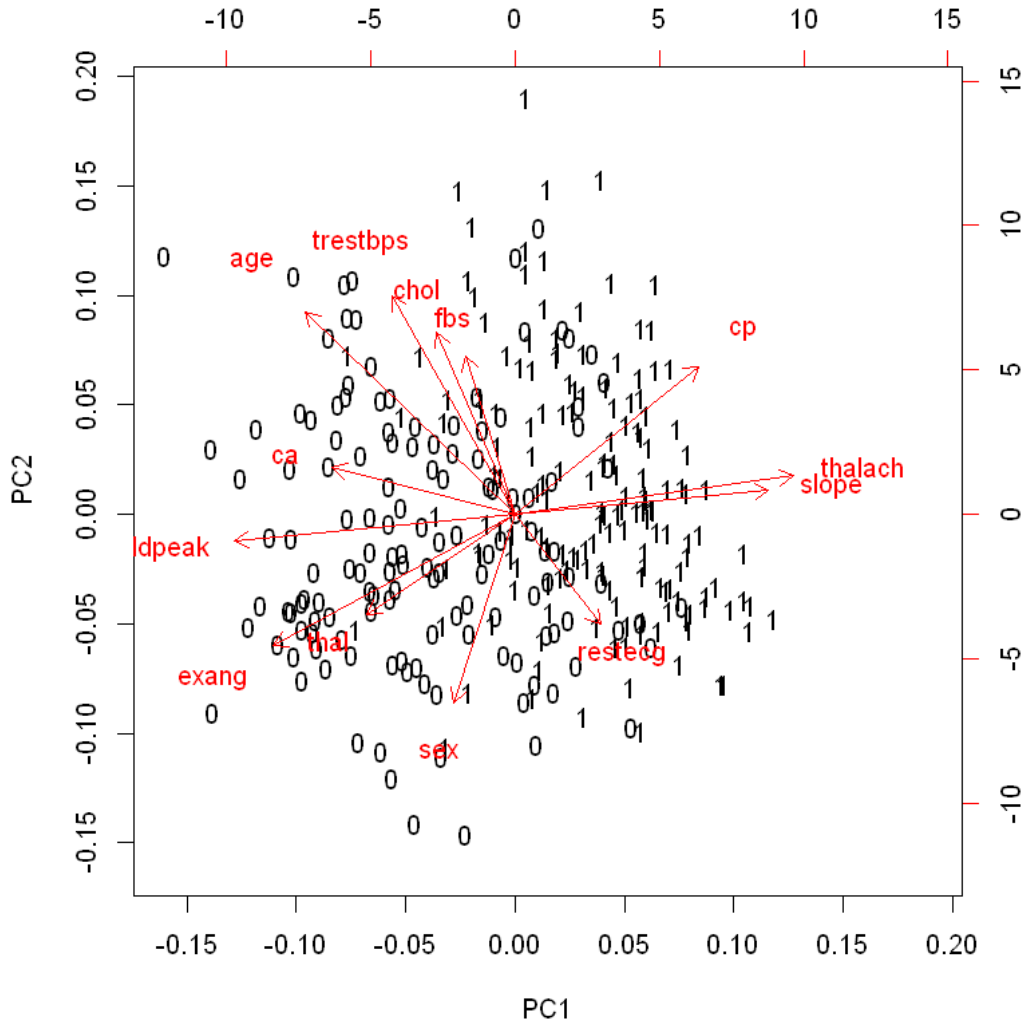


Hier valt te zien dat er niet echt duidelijk kan worden gemaakt wanneer iemand een hartziekte heeft.

9 Biplot: score plot + loading plot (controle met R)

Om te controleren of alles tot zover goed is gegaan, kan er met R snel een bi-plot worden gemaakt met behulp van `prcomp(...)` en `biplot(model)`.

```
In [13]: biplot(prcomp(df[,1:13], scale=T), xlabs=df$target)
```



De bi-plot die door R is gegenereerd, komt goed overeen met de zelfgemaakte bi-plot.

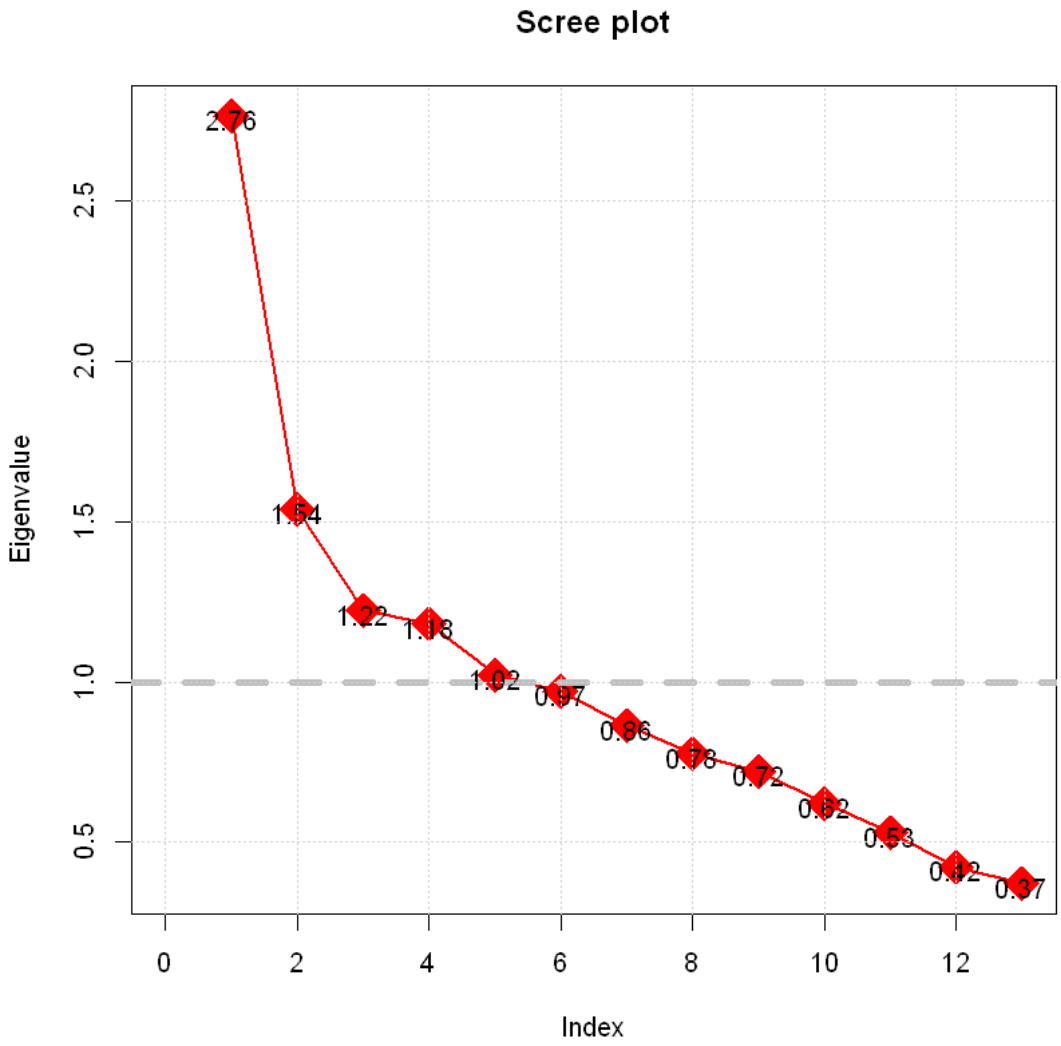
10 Scree plot en grenswaarden

Met een scree plot wordt er bepaald hoeveel PCs er worden meegenomen in een model. De grenswaarde die hiervoor geldt is als volgt bepaald:

$$grens = \frac{\text{trace}(\text{Covariantie matrix van } A)}{v}$$

waarbij v het aantal variabelen is. Met de onderstaande code wordt een scree plot gegenereerd:

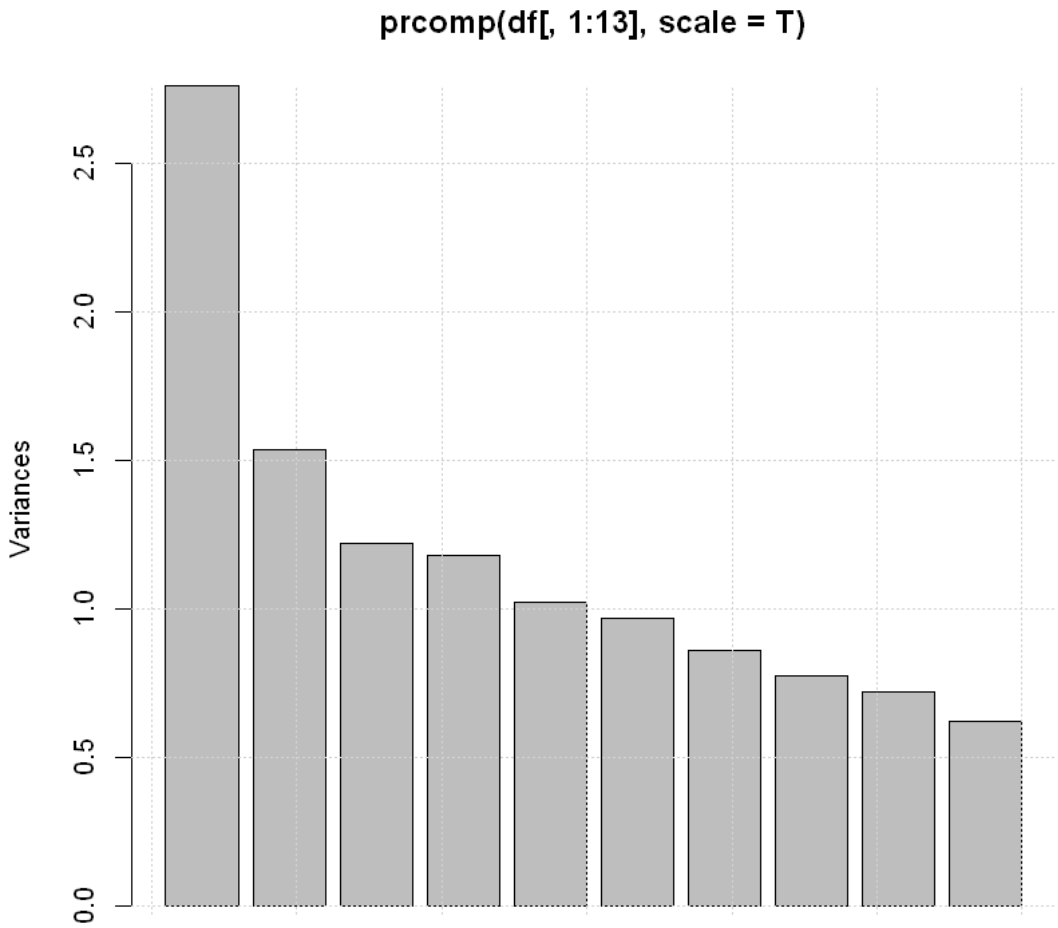
```
In [14]: #eigenwaarden bepalen
         lambdas = eigen(cov(Xcs))$values
         lambdas = sort(lambdas, decreasing=TRUE)

         # staafdiagram maken
         plot(lambdas, type='o', pch=18, col='red', cex=3, lwd=2
              , main="Scree plot", xlab="Index", ylab="Eigenvalue",
             xlim=c(0, length(lambdas)))

         # grenswaarde bepalen
         lim = trace((cov(Xcs))) / length(lambdas)

         # grenswaarde tekenen
         abline(h=lim, lt='dashed', lwd=4, col='gray')

         # eigenwaarden in de grafiek plotten
         text(lambdas - 0.01, labels=round(lambdas,2))
         grid()
```



Hier is te zien, omdat er is gestandardiseerd, dat de grenswaarde 1 is. Dit betekend dat er 5 PCs zijn waarvan de eigenwaarde groter is dan 1, en worden alleen deze meegenomen.

Met R kan ook een scree plot snel worden gecontroleerd:

```
In [15]: screeplot(prcomp(df[,1:13], scale=T))
         grid()
```



Met een visuele inspectie valt te zien dat de waarden overeen komen met de zelfgemaakte scree plot.

11 Errormatrices berekenen

De waarnemingen kunnen we ook opschrijven in een model als lineaire combinaties van de PCs en de t-scores. Het model hiervoor is:

$$X_{cs} = \hat{X}_{cs} + E,$$

waarbij $\hat{X}_{cs}$ de verklaarde variantie bevat en E de onverklaarde variantie bevat. Voor dit model worden er 5 PCs gebruikt, dus het gehele model is als volgt:

$$X_{cs} = t_1 \cdot p_1^T + t_2 \cdot p_2^T + \cdots + t_5 \cdot p_5^T + E_5.$$

Omdat alle t scores al zijn berekend, is het bepalen van de error matrices nog eenvoudiger. De formule hiervoor is $E_n = E_{n-1} - t_n \cdot p_n^T$. Vanuit de scree plot is er bepaald dat de eerste vijf PCs worden meegenomen in het model. Met de onderstaande code, worden de error matrices berekend.

```
In [16]: E0 = Xcs
         E1 = E0 - t1    %*% t(p[,1])
         E2 = E1 - t[,2] %*% t(p[,2])
         E3 = E2 - t[,3] %*% t(p[,3])
         E4 = E3 - t[,4] %*% t(p[,4])
         E5 = E4 - t[,5] %*% t(p[,5])
         E = list(E0, E1, E2, E3, E4, E5)
```

12 MSE berekenen en variantie-analyse

Om te bepalen hoeveel procent van de variante wordt verklaard per principeel component, wordt er een variantie-analyse uitgevoerd. Hiervoor worden voor alle error matrices $E_1, \ldots, E_5$ de variantie bepaald.

Met de volgende formule wordt de variantie voor een matrix bepaald:

$$\text{matrix.var}(\mathbf{E}) = \frac{\sum_{i=1}^{N}\sum_{j=1}^{v} e_{ij}^2}{N \cdot v} = \frac{\text{trace}(E^T \cdot E)}{N \cdot v}.$$

Vervolgens wordt dit gebruik om een tabel op te stellen met de volgende kolommen:

- Aantal PCs in het model
- Totale variantie
- Resterende variantie in %
- Verklaarde variantie in %
- Verklaarde variantie per PC

Met de onderstaande code wordt deze tabel opgesteld.

```
In [17]: # aantal PCs
         pcs = 0:5

         # totale variantie bepalen
         total.var = c(matrix.var(E0), matrix.var(E1), matrix.var(E2),
                       matrix.var(E3), matrix.var(E4), matrix.var(E5))

         # resterende variantie bepalen
         rest.var = c(1, total.var[2] / total.var[1], total.var[3] / total.var[1],
                       total.var[4] / total.var[1], total.var[5] / total.var[1], total.var[6])

         # verklaarde variantie bepalen
         expl.var = 1 - rest.var

         # verklaarde variantie per PC bepalen
```

```
var.per.pc = c(0, diff(expl.var))

# tabel opstellen
df.vars = data.frame(pcs, total.var, rest.var, expl.var, var.per.pc)
colnames(df.vars) = c("PCs in model", "Total variance", "Rest variance in %",
                      "Explained variance in %", "Explained variance per PC")
df.vars
```

PCs in model	Total variance	Rest variance in %	Explained variance in %	Explained variance per PC
0	0.9966997	1.0000000	0.0000000	0.00000000
1	0.7848606	0.7874595	0.2125405	0.21254053
2	0.6670436	0.6692524	0.3307476	0.11820708
3	0.5732899	0.5751882	0.4248118	0.09406418
4	0.4827324	0.4843309	0.5156691	0.09085735
5	0.4043791	0.4043791	0.5956209	0.07995181

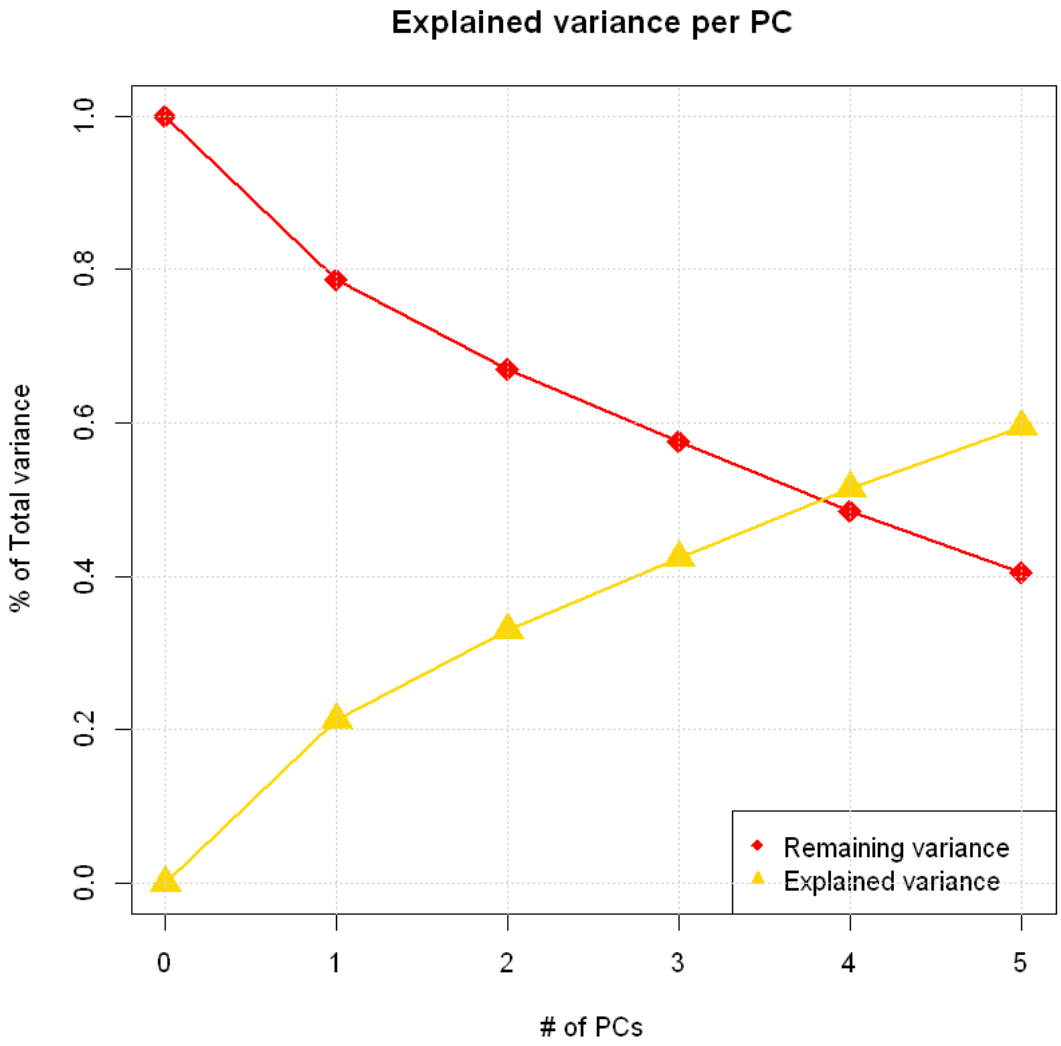
Wat opvalt is de PC_1 een redelijk deel van de variantie verklaard, namelijk 21%. De volgende, PC_2, voegt slechts 12% toe.

Ook is de tabel te controleren met R. Dezelfde kan tabel kan worden gevonden met `summary(...)`.

```
In [18]: summary(prcomp(df[,1:13], scale=T))

Importance of components:
                          PC1    PC2     PC3     PC4     PC5     PC6     PC7
Standard deviation     1.6622 1.2396 1.10582 1.08681 1.01092 0.98489 0.92885
Proportion of Variance 0.2125 0.1182 0.09406 0.09086 0.07861 0.07462 0.06637
Cumulative Proportion  0.2125 0.3307 0.42481 0.51567 0.59428 0.66890 0.73527
                           PC8    PC9    PC10    PC11    PC12   PC13
Standard deviation     0.88088 0.8479 0.78840 0.72808 0.65049 0.6098
Proportion of Variance 0.05969 0.0553 0.04781 0.04078 0.03255 0.0286
Cumulative Proportion  0.79495 0.8503 0.89807 0.93885 0.97140 1.0000
```

Hier valt te zien dat de verklaarde variantie in % voor een model met één PC overeenkomt met 0.212. De tabel kan ook worden weergegeven in een grafiek. Dit wordt gedaan met de onderstaande code.

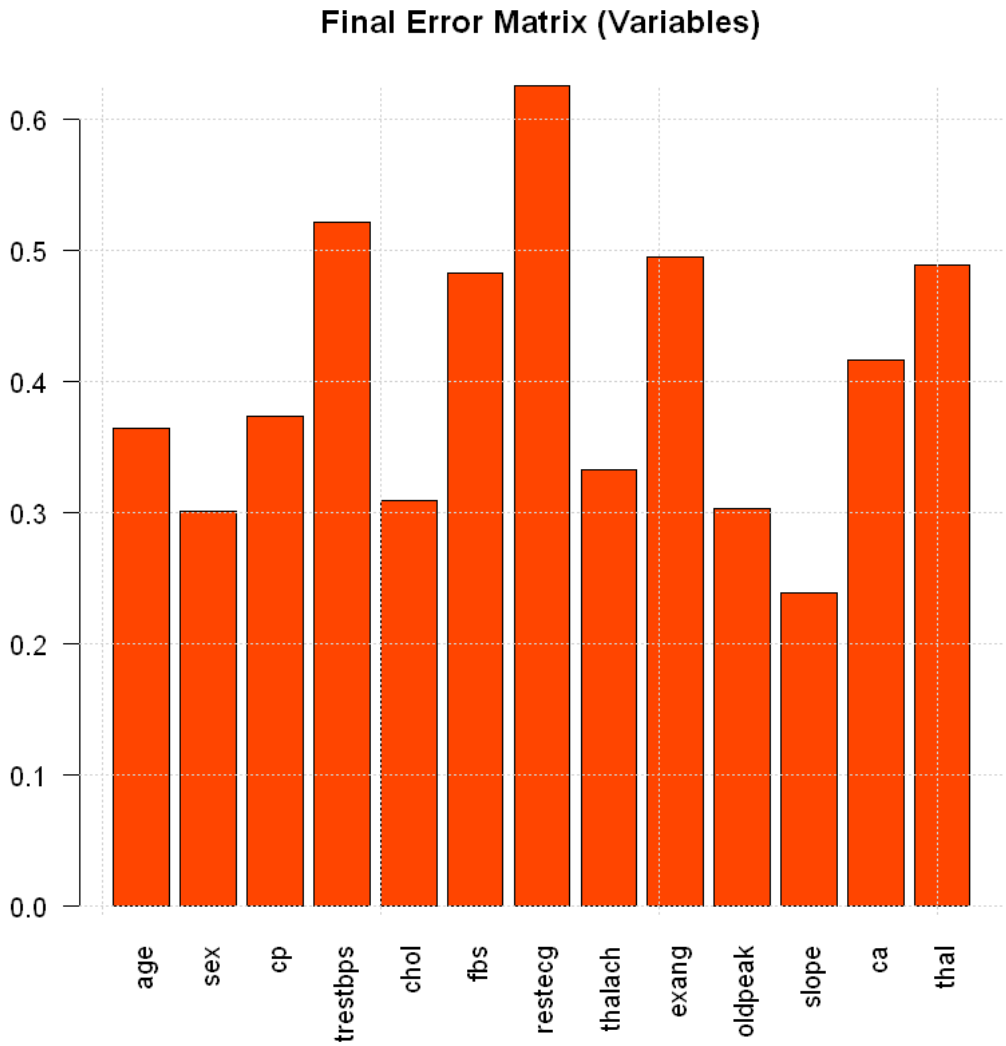
```
In [25]: # lijngrafiek maken voor verklaarde variantie
         plot(df.vars[,1], df.vars[,3], type='o', col='red', lwd=2, pch=18, cex=2,
              main="Explained variance per PC", ylim=c(0,1),
              ylab="% of Total variance", xlab="# of PCs")

         # lijngrafiek toevoegen voor onverklaarde variantie
         lines(df.vars[,1], df.vars[,4], type='o', col='gold', pch=17, lwd=2, cex=2)

         legend("bottomright", col=c('red', 'gold'),
                legend=c('Remaining variance', 'Explained variance'), pch=c(18, 17))
         grid()
```



Wat opvalt is dat bij de grenswaarde, waardoor er 5 PCs werden opgenomen, de lijnen elkaar kruisen. Op het moment dat er 5 PCs in het model zijn opgenomen, is de verklaarde variantie 59%.

Een andere aspect is het kijken naar mogelijke uitbijters. Hiervoor wordt er gekeken naar de gemiddelde kolom of rij varianties van de error matrices. Omdat er nogal veel waarnemingen zijn, is een grafiek van de rijen niet overzichtelijk. Echter is het wel mogelijk om van de kolommen een error matrix plot te maken. Met de onderstaande code wordt deze grafiek gegenereerd.

```
In [20]: rownames(E5) = 1:nrow(df)
         barplot(colMeans(E5^2), las=2, col='orangered1')
         grid()
         title(main="Final Error Matrix (Variables)")
```



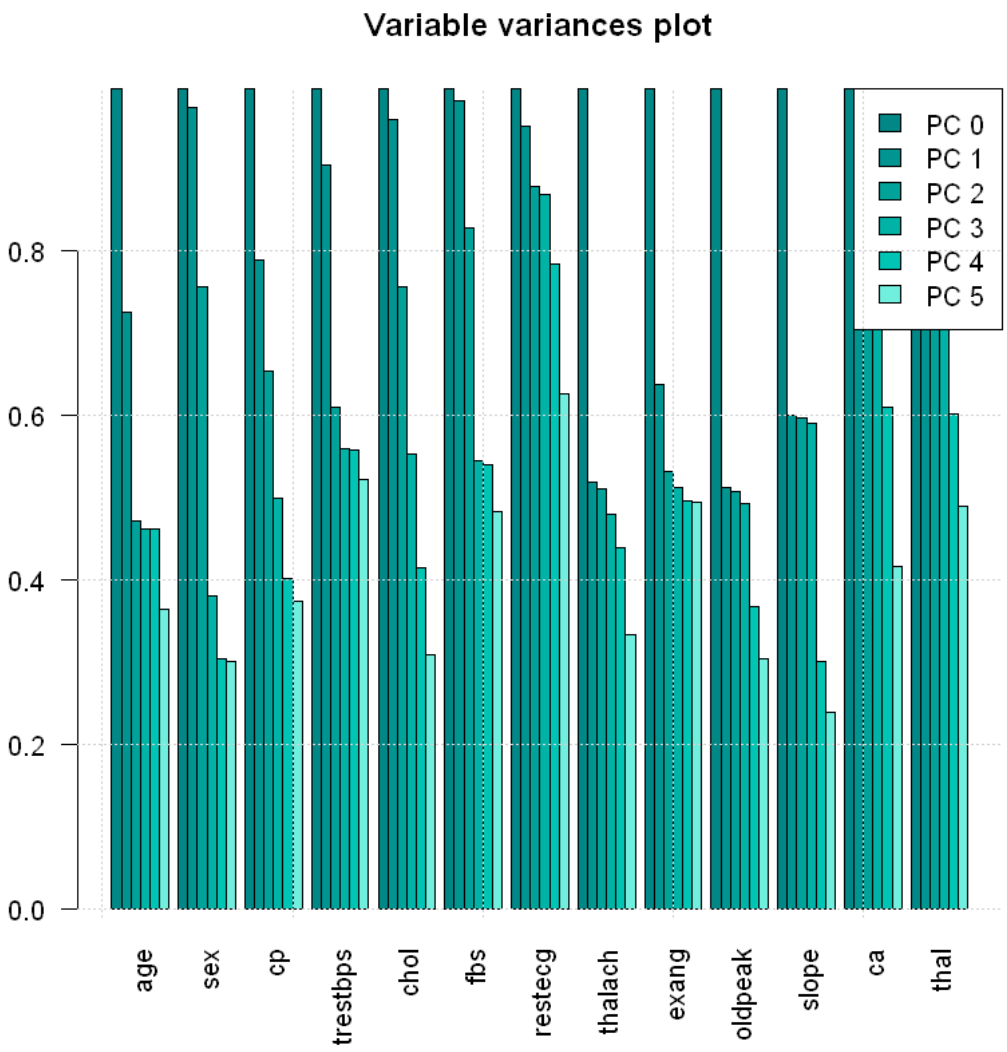
Hier is te zien dat de fout over alle variabelen goed verdeeld is. Er is geen variabele die een grote fout veroorzaakt in de laatste error matrix (E_5). Vanuit dit oogpunt zitten er geen uitbijters in de gegevens.

13 Samplevarianties en variabele varianties

Een andere grafiek laat zien hoe de variantie afneemt bij het toevoegen van meerdere PCs. Ook hier is er de mogelijkheid om dit voor de kolommen en de rijen te doen. Aangezien er veel rijen zijn, wordt dit alleen voor de kolommen gedaan. De grafiek die anders ontstaat is niet duidelijk.

Met de onderstaande code wordt deze grafiek gegenereerd voor de variabelen.

In [21]:
```
# kolomgemiddelden bepalen voor de error matrices
variable.vars = data.frame(colMeans((E0)^2),colMeans((E1)^2),
                           colMeans((E2)^2),colMeans((E3)^2),
                           colMeans((E4)^2),colMeans((E5)^2))

# kleuren
colors = c('#018786', '#019592', '#01a299',
           '#00b3a6', '#00c4b4', '#70efde')

# staafdiagram maken
barplot(t(variable.vars), las=2, beside=T,
        main="Variable variances plot", col=colors)
legend("topright", legend=c("PC 0", "PC 1", "PC 2",
                            "PC 3", "PC 4", "PC 5"), fill=colors)
grid()
```



Hier valt te zien dat een gedeelte van de variantie is opgenomen in het eerste principele component PC_1. Voornamelijk voor de variabele met een grote loading in p_1, is er te zien dat een deel van de variantie wordt verklaard met PC_1. Doordat de variabelen `thalach`, `exang`, `oldpeak` en `slope` een grote loading hebben, hebben ze vermoedelijk een grote invloed op het feit of iemand een hartziekte heeft. De analyse laat zien dat deze variabelen het zwaarst mee wegen in het eerste model.

Vanuit de loading plot is ook goed te zien dat juist deze variabelen ook de correcte richting uitoefenen op waar de waarnemingen belanden in de score plot.

14 Verschillen en verbanden

In de vorige paragraaf is gebleken dat een gedeelte van de variantie in het model met 1 PC wordt verklaard door de variabelen `thalach`, `exang`, `oldpeak` en `slope`. Om te kijken of er inderdaad een significant verschil is tussen deze variabelen en de splitsingsvariabele `target`, is er met SPSS een t-toets uitgevoerd voor het verschil in gemiddelden. Hiervoor is alleen van `slope` geen toets uitgevoerd, omdat in dit geval beide variabelen nominaal zijn.

Het resultaat van de toets tussen alle ratio variabelen en de splitsingsvariabele `target` is te vinden in de onderstaande tabel:

Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
									Lower	Upper
age	Equal variances assumed	7.326	0.007	4.015	301	0	4.104	1.022	2.093	6.116
	Equal variances not assumed			4.08	300.998	0	4.104	1.006	2.125	6.084
trestbps	Equal variances assumed	2.75	0.098	2.541	301	0.012	5.096	2.005	1.15	9.041
	Equal variances not assumed			2.508	272.558	0.013	5.096	2.031	1.096	9.095
chol	Equal variances assumed	0.036	0.849	1.484	301	0.139	8.857	5.967	-2.886	20.599
	Equal variances not assumed			1.495	298.03	0.136	8.857	5.925	-2.803	20.517
thalach	Equal variances assumed	5.652	0.018	-8.07	301	0	-19.365	2.4	-24.088	-14.643
	Equal variances not assumed			-7.953	269.903	0	-19.365	2.435	-24.159	-14.571
oldpeak	Equal variances assumed	38.209	0	8.28	301	0	1.0025	0.1211	0.7642	1.2407
	Equal variances not assumed			7.939	215.677	0	1.0025	0.1263	0.7536	1.2514

Hier is te zien dat `age`, `trestbps`, `thalach` en `oldpeak` allen een significant verschil vertonen. In de loading plot is te zien dat `trestbps` en `age` beide een richting uitoefenen die bijna loodrecht

op `thalach` en `oldpeak` staat. Wat ook opvalt is dat voor de groep met een hartziekte, de maximaal behaalde hartslag tijden de oefening, gemiddeld bijna 20 BPM hoger is dan bij de groep zonder hartziekte. Ditzelfde geldt voor `oldpeak`, deze scoort gemiddeld `1` lager bij de groep met een hartziekte. Dit geeft de volgende vraag, is er een verband tussen `oldpeak` en `thalach`? Vanuit het onderzoek met meervoudige regressie is gebleken dat de correlatiecoëfficiënt hiertussen -0.344 is. Een snelle toets in SPSS toont aan dat dit resultaat significant is met een p-waarde van 0.00 (toetsresultaat achterwege gelaten). Er is dus een negatief zwak verband tussen `oldpeak` en `thalach`.

15 Conclusie

Vanuit de principele componenten analyse en de verschiltoets is gebleken dat de variabelen die de grootste invloed uitoefenen op de mogelijkheid of iemand een hartziekte heeft, `oldpeak` en `thalach` zijn. Uit de loading plot valt te zien dat `slope` en `exang` een grote invloed uitoefenen op de mogelijkheid of iemand een hartziekte heeft. Echter zijn dit nominale variabelen en is hiervan geen verschiltoets uitgevoerd.

De variabele `oldpeak` staat voor *ST depression induced by exercise relative to test*. Hiermee wordt een specifiek fenomeen bedoeld wat voorkomt in een ECG (electrocardiograph) wat een indicatie is voor een onderliggend probleem genaamd ischemia. [1] Als een van de bloedvatten verstoopt raakt, onstaat er een ernstig probleem genaamd *ischemia*. [4] De variabele `slope` duidt een ander fenomeen aan in een ECG, wat ook duidt op een onderliggend hartprobleem.

De andere variabele `thalach` geeft de maximaal behaalde hartslag tijdens de oefening aan. Hier valt het op dat bij de mensen met een hartprobleem, deze gemiddeld 20 BPM hoger is.

Beide variabelen geven een goede indicatie op mogelijke hartproblemen. Aangezien niet iedereen thuis een ECG heeft, kan je dit zelf niet controleren. Echter is het wel mogelijk om zelf de maximaal behaalde hartslag tijdens een oefening bij te houden. Als deze hoger ligt dan het gemiddelde, wat bepaald is per leeftijd, is dat een goede reden voor een doktersbezoek.

De laatste variabele, `exang`, geeft aan of iemand pijn of oncomfortabelheden in zijn of haar borst heeft ervaren gedurende de oefening. Dit fenomeen wordt angina genoemd. Dit word mogelijk veroorzaakt doordat er te weinig zuurstof-rijk bloed bij die spier komt. Ook dit is een symptoom van een onderliggend hartprobleem. [3]

16 Referenties

In dit rapport zijn de volgende referenties geraadpleegd:

1. https://ecgwaves.com/ecg-topic/ecg-st-segment-depression-ischemia-infarction-differential-diagnoses/
2. https://www.webmd.com/heart-disease/electrocardiogram-ekgs#1
3. https://www.heart.org/en/health-topics/heart-attack/angina-chest-pain
4. https://www.webmd.com/heart-disease/what-is-ischemia#1