**Groep 10**

Wiebe de Boer

Casper Hooft

Nils Hettinga

Kevin Haakma

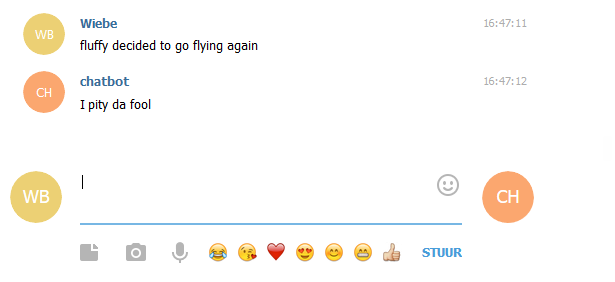
Elon Gielink

**Video**

<https://youtu.be/2__vUtmFNsM>

**Persoonlijkheid**

**Wiebe**

****

**RiveScript**

// brain.rive

//STARTUP

+ (hello|hi|hoi|hey|goeie|goedemorgen|goedendag|goedemiddag|goedenavond|goedenacht|goeiendag|gedag|dag|agoeie|salut|pannenkoek)

- Hi there!

- hello

- hi

- hoi

- hey

- :)

+ (how are you \*|hoe gaat ie?|how are you?|hoe gaat het?|hoe voel je je daarbij?)

- I'm fine.

- Splendid!

- Couldn't be better!

//wiebe

+ [\*] start [\*]

- Ik ben de Movie Chat bot! Stel me vragen over je favoriete film. Type /help voor voorbeeld vragen.

//wiebe

+ fluffy decided to go flying again

- I pity da fool

//HELP

//wiebe

+ [\*] help [\*]

- Je kan me vragen stellen zoals:\n

^ In welke films speelde Joop Braakhekke?\n

^ Welke acteur heeft de langste filmcarriere?\n

^ Welke acteur heeft de meeste dubbelrollen?\n

^ Welke acteur of actrice speelt het meest in de slechtst gewaardeerde films?\n

^ Wat is de gemiddelde waardering van films?\n

^ Welke acteur of actrice speelt het meest in de slechtst gewaardeerde films? Met een waardering lager dan een 5 gemiddelde.\n

^ Welke 3 acteurs of actrices hebben samen in de meeste films gespeeld?\n

^ Wordcloud van acteurs uit een land. Kies een land Nederland\n

^ Maak een visuele weergave van de hoeveelheid acteurs voor een film.\n

^ Klopt het naarmate actrices ouder zijn zij moeilijker een rol krijgen.\n

^ tel actors\n

^ barplot actors\n

^ fluffy decided to go flying again\n

//wiebe beheer FOREIGN KEYS ID

+ [\*] management [\*]

- You can command me:\n

^\n update countries films

^\n update genres films

^\n update ratings films

^\n update directed films

^\n update roles films

^\n update countries series

^\n update genres series

^\n update ratings series

^\n update directed series

^\n update roles series

^\n update director

^\n update actor

//wiebe

//films

+ update countries films

- <call>jdbc localhost 3306 imdb root sql080 update set countries.MovieID = movies.MovieID from countries, movies where countries.Movie = movies.Movie</call>

+ update genres films

- <call>jdbc localhost 3306 imdb root sql080 update set genres.MovieID = movies.MovieID from genres, movies where genres.Movie = movies.Movie</call>

+ update ratings films

- <call>jdbc localhost 3306 imdb root sql080 update set ratings.MovieID = movies.MovieID from ratings, movies where ratings.Movie = movies.Movie</call>

+ update directed films

- <call>jdbc localhost 3306 imdb root sql080 update set directed.MovieID = movies.MovieID from movies, directed where directed.Movie = movies.Movie </call>

+ update roles films

- <call>jdbc localhost 3306 imdb root sql080 update set roles.MovieID = movies.MovieID from roles, movies where roles.Movie = movies.Movie </call>

//wiebe

//director en actor

+ update actor

- <call>jdbc localhost 3306 imdb root sql080 update set directed.DirectorID = directors.DirectorID from directors, directed where directed.Director = movies.Director </call>

+ update director

- <call>jdbc localhost 3306 imdb root sql080 update set roles.ActorID = actors.ActorID from actors, roles where actors.Actor = movies.Actor </call>

//wiebe

//series

+ update countries series

- <call>jdbc localhost 3306 imdb root sql080 update set countries.MovieID = movies.MovieID from countries, movies where countries.SerieName = movies.SerieName</call>

+ update genres series

- <call>jdbc localhost 3306 imdb root sql080 update set genres.MovieID = movies.MovieID from genres, movies where genres.SerieName = movies.SerieName</call>

+ update ratings series

- <call>jdbc localhost 3306 imdb root sql080 update set ratings.MovieID = movies.MovieID from ratings, movies where ratings.SerieName = movies.SerieName</call>

+ update directed series

- <call>jdbc localhost 3306 imdb root sql080 update set directed.MovieID = movies.MovieID from movies, directed where directed.SerieName = movies.SerieName </call>

+ update roles series

- <call>jdbc localhost 3306 imdb root sql080 update set roles.MovieID = movies.MovieID from roles, movies where roles.SerieName = movies.SerieName </call>

//ALGEMEEN

+ \*

- Niet zeker of ik dat begrepen heb. Type /help voor voorbeeldvragen.

+ show files

- <call>system dir</call>

+ (wat is de datum?|date|vandaag|dag|week|maand)

- De datum is: <call>netstat</call>

//kevin

+ tel actors

- <call>jdbc localhost 3306 imdb root sql080 select count(Actor) from actors</call>

//MOVIE QUESTIONS

//wiebe

+ in welke films speelde \_ \*

- <call>jdbc localhost 3306 imdb root sql080 select dictinct(Movie) from roles where Actor like ='<star>'</call>

//wiebe

+ welke acteur heeft de meeste dubbelrollen?

- <call>jdbc localhost 3306 imdb root sql080 select Actor, count(Role) from roles group by Actor having count(Role) >1 order by count(Role) desc limit 0,1</call>

//wiebe

+ welke acteur heeft de langste filmcarriere?

- <call>jdbc localhost 3306 imdb root sql080 select actors.Actor, min(movies.Year) as minmov, max(movies.Year) as maxmov, maxmo - minmov as maxrange from roles, actors, movies where roles.Movie = movies.Movie and roles.Actor = actors.Actor and actors.Actor order by maxrange desc limit 0,1</call>

//wiebe

+ wat is de gemiddelde waardering van films?

- <call>jdbc localhost 3306 imdb root sql080 select avg(Rating) as ratingavg from ratings</call>

//wiebe

+ welke acteur of actrice speelt het meest in de slechtst gewaardeerde \* \_ gemiddelde.

- <call>jdbc localhost 3306 imdb root sql080 select actors.Actor as actorname, count(roles.ActorID) as maxbad from roles, actors, movies, ratings where roles.MovieID = movies.MovieID and roles.ActorID = actors.ActorID and ratings.MovieID = movies.MovieID and ratings.Rating < '<star>' order by maxbad LIMIT 0,1</call>

//wiebe

+ welke \_ acteurs of actrices hebben samen in de meeste films gespeeld?

- <call>jdbc localhost 3306 imdb root sql080 select actors.Actor as actor, count(roles.ActorID) as rolecount from roles,actors,movies where roles.MovieID = movies.MovieID and roles.ActorID = actors.ActorID order by rolecount limit 0,<star></call>

//VISUALISATIES en R

//kevin

+ barplot actors

- <call>system rscript resources/R/test.R</call>

^ <call>send photo C:/temp/output.jpg The graph produced by R</call>

//sql van wiebe, wordcloud script van

+ wordcloud \* kies een land \_

- There it is!

//- <call>jdbc localhost 3306 imdb root sql080 select actors.Actor as actorname, count(roles.ActorID) as countroles from roles, actors, movies, countries where roles.MovieID = movies.MovieID and roles.ActorID = actors.ActorID and countries.MovieID = movies.MovieID and Country ='<star>'</call>

^ <call>system resources/R/wordcloud.R 2&> /dev/null</call>

^ <call>send photo /tmp/wordcloud.jpg The graph produced by R</call>

//sql van wiebe, r script van

+ maak een visuele weergave van de hoeveelheid acteurs voor een film.

- There it is!

//- <call>jdbc localhost 3306 imdb root sql080 select actors.Actor as actorname, count(roles.ActorID) as countroles from roles, actors, movies where roles.MovieID = movies.MovieID and roles.ActorID = actors.ActorID</call>

^ <call>system resources/R/acteurgraaf.R 2&> /dev/null</call>

^ <call>send photo /tmp/acteurgraaf.jpg The graph produced by R</call>

//sql van wiebe, r model van

+ klopt het naarmate actrices ouder zijn zij moeilijker een rol krijgen.

- There it is!

//- <call>jdbc localhost 3306 imdb root sql080 select actors.Actor as actorname, count(roles.ActorID) as countroles, movies.Year as movieyear from roles, actors, movies where roles.MovieID = movies.MovieID and roles.ActorID = actors.ActorID and actors.Gender ='1' </call>

^ <call>system resources/R/oudeactrices.R 2&> /dev/null</call>

^ <call>send photo /tmp/oudeactrices.jpg The graph produced by R</call>

//sql van wiebe, r model van

+ zoek uit welke factoren een rol spelen bij het bepalen hoeveel seizoenen van een tv-serie worden gemaakt.

- There it is!

//- <call>jdbc localhost 3306 imdb root sql080 select actors.Actor as actorname, count(roles.ActorID) as countroles, movies.Year as movieyear, ratings.Rating as serierating from roles, actors, movies where roles.MovieID = movies.MovieID and roles.ActorID = actors.ActorID and SerieName !='NULL'</call>

^ <call>system resources/R/factoren.R 2&> /dev/null</call>

^ <call>send photo /tmp/factoren.jpg The graph produced by R</call>

// SUBSTITUTIONS

! sub i'm = i am

! sub i'd = i would

! sub i've = i have

! sub i'll = i will

! sub don't = do not

! sub isn't = is not

! sub you'd = you would

! sub you're = you are

! sub you've = you have

! sub you'll = you will

! sub what's = what is

! sub whats = what is

! sub what're = what are

! sub what've = what have

! sub what'll = what will

! sub who's = who is

//COMMENTED STUFF

//+

//- <call>jdbc localhost 3306 imdb root sql080 select </call>

//+ which movies did \_ \_ \*

//- <call>jdbc localhost 3306 imdb root sql080 select title from cast\_info, name, title, kind\_type where person\_id = name.id and title.id = movie\_id and kind\_id = kind\_type.id and name like '<star2>, <star>' and kind = 'movie'</call>

//+ [\*] video formats [\*]

//- There it is!

//^ <call>system resources/R/video-format.R 2&> /dev/null</call>

//^ <call>send photo /tmp/video-format.jpg The graph produced by R</call>

**SQL Vragen**

**Kevin + Wiebe**

**wat is de gemiddelde waardering van films?**

//wiebe

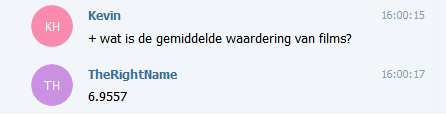
+ wat is de gemiddelde waardering van films?

- <call>jdbc localhost 3306 imdb root sql080 select avg(Rating) as ratingavg from ratings</call>

BOT

6.9557

select avg(Rating) as ratingavg from ratings



**Wiebe**

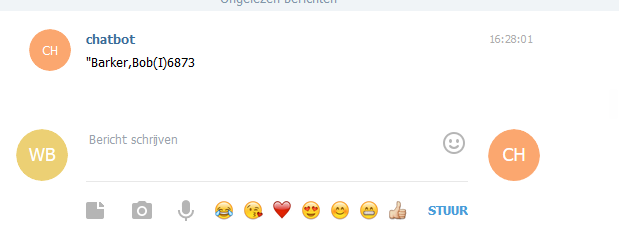
**welke acteur heeft de meeste dubbelrollen?**

//wiebe

+ welke acteur heeft de meeste dubbelrollen?

- <call>jdbc localhost 3306 imdb root sql080 select Actor, count(Role) from roles group by Actor having count(Role) >1 order by count(Role) desc limit 0,1</call>

select Actor, count(Role) from roles group by Actor having count(Role) >1 order by count(Role) desc limit 0,1

"Barker,Bob(I)6873

**Wiebe**

**In welke films speelde El de Chipiona?**

select dictinct(Movie) from roles where Actor like ='<star>'

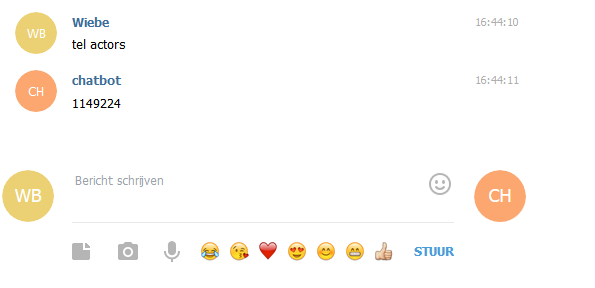
**Wiebe + Kevin**

**tel actors**

+ tel actors

- <call>jdbc localhost 3306 imdb root sql080 select count(Actor) from actors</call>

1149224



**Visualisaties en R**

**Wiebe + Kevin**

**barplot actors**

+ barplot actors

- <call>system rscript resources/R/test.R</call>

^ <call>send photo C:/temp/output.jpg The graph produced by R</call>

**R Script**

library("DBI")

library("RMySQL")

#kevin R

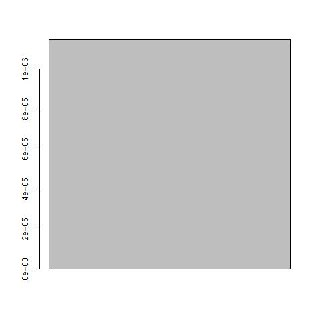
con <- dbConnect(MySQL(), dbname="imdb", user="root", password="sql080")

values <- dbGetQuery(con, "select count(Actor) as count from actors")

invisible(jpeg('C:/temp/output.jpg'))

barplot(values$count, horiz=FALSE, cex.names=0.5)

invisible(dev.off())

****

**Wiebe + Kevin + Casper**

**Wordcloud**

//sql van wiebe, wordcloud script van

+ wordcloud \* kies een land \_

- There it is!

//- <call>jdbc localhost 3306 imdb root sql080 select actors.Actor as actorname, count(roles.ActorID) as countroles from roles, actors, movies, countries where roles.MovieID = movies.MovieID and roles.ActorID = actors.ActorID and countries.MovieID = movies.MovieID and Country ='<star>'</call>

^ <call>system resources/R/wordcloud.R 2&> /dev/null</call>

^ <call>send photo /tmp/wordcloud.jpg The graph produced by R</call>

**R script**

|  |
| --- |
| #!/usr/bin/Rscript |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| args = commandArgs(trailingOnly=TRUE) |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| # install.packages("RMySQL") |
|  |

|  |
| --- |
| library(RMySQL) |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| #wiebe R |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| # Install |
|  |

|  |
| --- |
| #install.packages("tm") # for text mining |
|  |

|  |
| --- |
| #install.packages("SnowballC") # for text stemming |
|  |

|  |
| --- |
| #install.packages("wordcloud") # word-cloud generator |
|  |

|  |
| --- |
| #install.packages("RColorBrewer") # color palettes |
|  |

|  |
| --- |
| # Load |
|  |

|  |
| --- |
| library("tm") |
|  |

|  |
| --- |
| library("SnowballC") |
|  |

|  |
| --- |
| library("wordcloud") |
|  |

|  |
| --- |
| library("RColorBrewer") |
|  |

|  |
| --- |
| library("magick") |
|  |

|  |
| --- |
| library("TraMineRextras") |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| con <- dbConnect(MySQL(), dbname="imdb", user="root", password="kevinhaakma") |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| query <- paste("select name as actorname from actors where name like '%",args,"%'",sep="") |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| values <- dbGetQuery(con, query ) |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| # Load the data as a corpus |
|  |

|  |
| --- |
| docs <- Corpus(VectorSource(values)) |
|  |

|  |
| --- |
| #text transformation |
|  |

|  |
| --- |
| toSpace <- content\_transformer(function (x , pattern ) gsub(pattern, " ", x)) |
|  |

|  |
| --- |
| docs <- tm\_map(docs, toSpace, "/") |
|  |

|  |
| --- |
| docs <- tm\_map(docs, toSpace, "@") |
|  |

|  |
| --- |
| docs <- tm\_map(docs, toSpace, "\\|") |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| # Convert the text to lower case |
|  |

|  |
| --- |
| docs <- tm\_map(docs, content\_transformer(tolower)) |
|  |

|  |
| --- |
| # Remove numbers |
|  |

|  |
| --- |
| docs <- tm\_map(docs, removeNumbers) |
|  |

|  |
| --- |
| # Remove english common stopwords |
|  |

|  |
| --- |
| docs <- tm\_map(docs, removeWords, stopwords("english")) |
|  |

|  |
| --- |
| # Remove your own stop word |
|  |

|  |
| --- |
| # specify your stopwords as a character vector |
|  |

|  |
| --- |
| docs <- tm\_map(docs, removeWords, c("blabla1", "blabla2")) |
|  |

|  |
| --- |
| # Remove punctuations |
|  |

|  |
| --- |
| docs <- tm\_map(docs, removePunctuation) |
|  |

|  |
| --- |
| # Eliminate extra white spaces |
|  |

|  |
| --- |
| docs <- tm\_map(docs, stripWhitespace) |
|  |

|  |
| --- |
| # Text stemming |
|  |

|  |
| --- |
| # docs <- tm\_map(docs, stemDocument) |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| #term document matrix |
|  |

|  |
| --- |
| dtm <- TermDocumentMatrix(docs) |
|  |

|  |
| --- |
| m <- as.matrix(dtm) |
|  |

|  |
| --- |
| v <- sort(rowSums(m),decreasing=TRUE) |
|  |

|  |
| --- |
| d <- data.frame(word = names(v),freq=v) |
|  |

|  |
| --- |
| head(d, 10) |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| #generate word cloud |
|  |

|  |
| --- |
| set.seed(1234) |
|  |

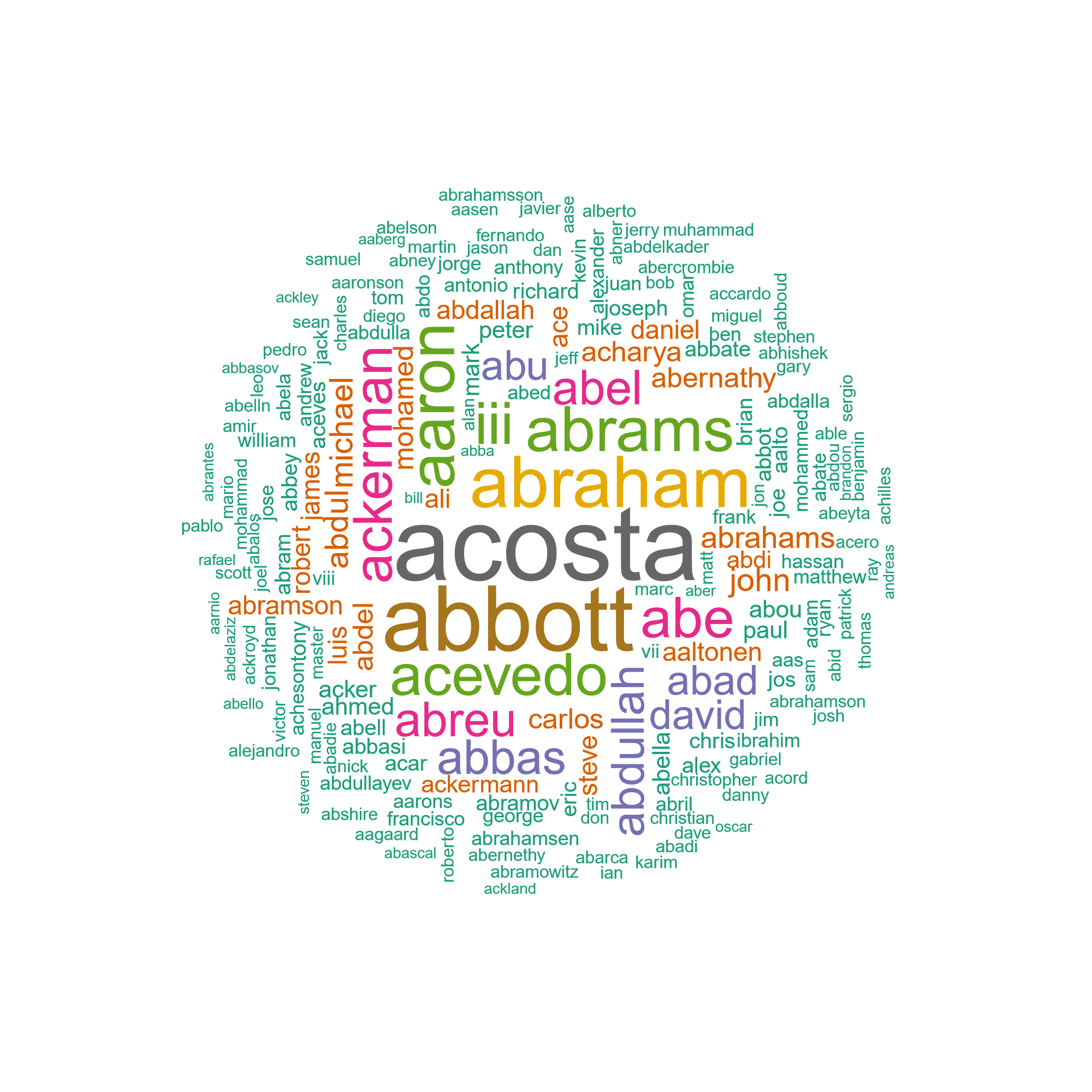
|  |
| --- |
| invisible(jpeg('c:/temp/output.jpg')) |
|  |

|  |
| --- |
| wordcloud(words = d$word, freq = d$freq, min.freq = 1, |
|  |

|  |
| --- |
| max.words=200, random.order=FALSE, rot.per=0.35, |
|  |

|  |
| --- |
| colors=brewer.pal(8, "Dark2")) |
|  |

|  |
| --- |
|  |
|  |

invisible(dev.off())****

**Wiebe**

**Oude actrices**

//sql van wiebe, r model van

+ klopt het naarmate actrices ouder zijn zij moeilijker een rol krijgen.

- There it is!

//- <call>jdbc localhost 3306 imdb root sql080 select actors.Actor as actorname, count(roles.ActorID) as countroles, movies.Year as movieyear from roles, actors, movies where roles.MovieID = movies.MovieID and roles.ActorID = actors.ActorID and actors.Gender ='1' </call>

^ <call>system resources/R/oudeactrices.R 2&> /dev/null</call>

^ <call>send photo /tmp/oudeactrices.jpg The graph produced by R</call>

**R script**

#!/usr/bin/Rscript

# install.packages("RMySQL")

library(RMySQL)

#wiebe R

con <- dbConnect(MySQL(), dbname="imdb", user="root", password="sql080")

values <- dbGetQuery(con, "select actors.Actor as actorname, count(roles.ActorID) as countroles, movies.Year as movieyear from roles, actors, movies where roles.MovieID = movies.MovieID and roles.ActorID = actors.ActorID and actors.Gender ='1'")

#model

model <-glm(serierating~countroles+movieyear,data=values,family=binomial)

summary(model)

#prediction

predictTest <-predict(model,type="response")

#library roc

library(ROCR)

ROCRperfi <-performance(predictTest,"tpr","fpr")

#ROC plot

invisible(jpeg('/tmp/oudeactrices.jpg'))

plot(ROCRperfi,colorize=true,print.cutoffs.at=seq(0,1,0.1))

invisible(dev.off())

abline(0,1)

#AUC

AUC <-as.numeric(performance(predictTelco,”AUC”)@y.values)

str(AUC)

#confusion matrix function

confusion <-function(arg1,arg2,arg3){

table = table(arg1,arg2>arg3)

specificity <-table[1,1]/(table[1,1]+table[1,2])

sensitivity <-table[2,2]/(table[2,1]+table[2,2])

accuracy <-(table[1,1]+table[2,2])/(table[1,1]+table[1,2]+table[2,1]+table[2,2])

return(data.frame(Specificity=specificity,Sensitivity=sensitivity,Accuracy=accuracy))}

#show confusion

confusion(model$serierating,predictTest,0.1)

**Wiebe**

**Welke factoren Herhaling tv serie**

//sql van wiebe, r model van

+ zoek uit welke factoren een rol spelen bij het bepalen hoeveel seizoenen van een tv-serie worden gemaakt.

- There it is!

//- <call>jdbc localhost 3306 imdb root sql080 select actors.Actor as actorname, count(roles.ActorID) as countroles, movies.Year as movieyear, ratings.Rating as serierating from roles, actors, movies where roles.MovieID = movies.MovieID and roles.ActorID = actors.ActorID and SerieName !='NULL'</call>

^ <call>system resources/R/factoren.R 2&> /dev/null</call>

^ <call>send photo /tmp/factoren.jpg The graph produced by R</call>

**R script**

#!/usr/bin/Rscript

# install.packages("RMySQL")

library(RMySQL)

#wiebe R

#db connection

con <- dbConnect(MySQL(), dbname="imdb", user="root", password="sql080")

#values

values <- dbGetQuery(con, "select actors.Actor as actorname, count(roles.ActorID) as countroles, movies.Year as movieyear, ratings.Rating as serierating from roles, actors, movies where roles.MovieID = movies.MovieID and roles.ActorID = actors.ActorID and SerieName !='NULL'")

#model

model <-glm(serierating~countroles,data=values,family=binomial)

summary(model)

#prediction

predictTest <-predict(model,type="response")

#library roc

library(ROCR)

ROCRperfi <-performance(predictTest,"tpr","fpr")

#ROC plot

invisible(jpeg('/tmp/factoren.jpg'))

plot(ROCRperfi,colorize=true,print.cutoffs.at=seq(0,1,0.1))

invisible(dev.off())

abline(0,1)

#AUC

AUC <-as.numeric(performance(predictTelco,”AUC”)@y.values)

str(AUC)

#confusion matrix function

confusion <-function(arg1,arg2,arg3){

table = table(arg1,arg2>arg3)

specificity <-table[1,1]/(table[1,1]+table[1,2])

sensitivity <-table[2,2]/(table[2,1]+table[2,2])

accuracy <-(table[1,1]+table[2,2])/(table[1,1]+table[1,2]+table[2,1]+table[2,2])

return(data.frame(Specificity=specificity,Sensitivity=sensitivity,Accuracy=accuracy))}

#show confusion

confusion(model$serierating,predictTest,0.1)