Visualizing the approach and exceedance of thresholds

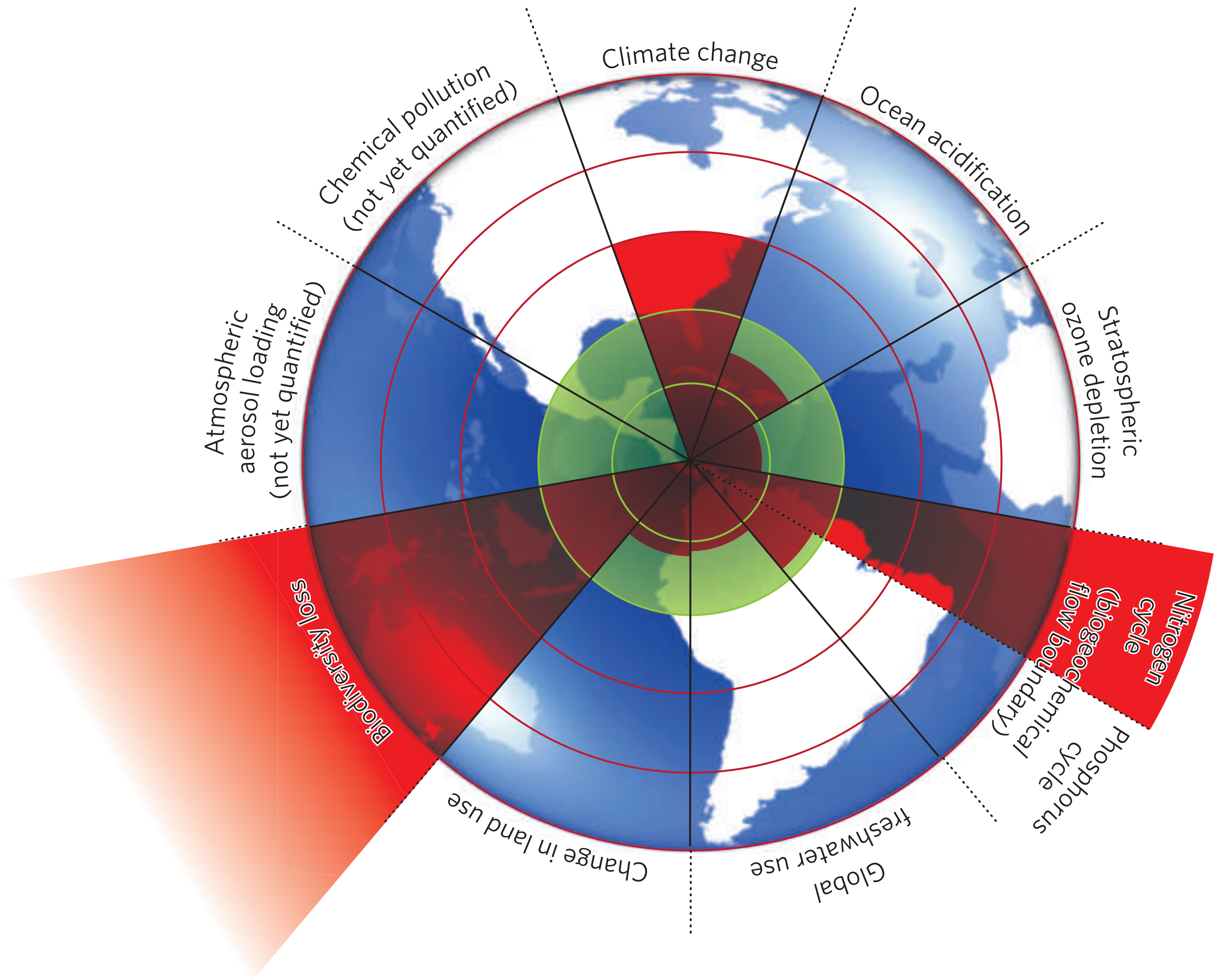
Glenn Moncrieff
Ixio Analytics

satRday Cape Town
18 Feb 2017

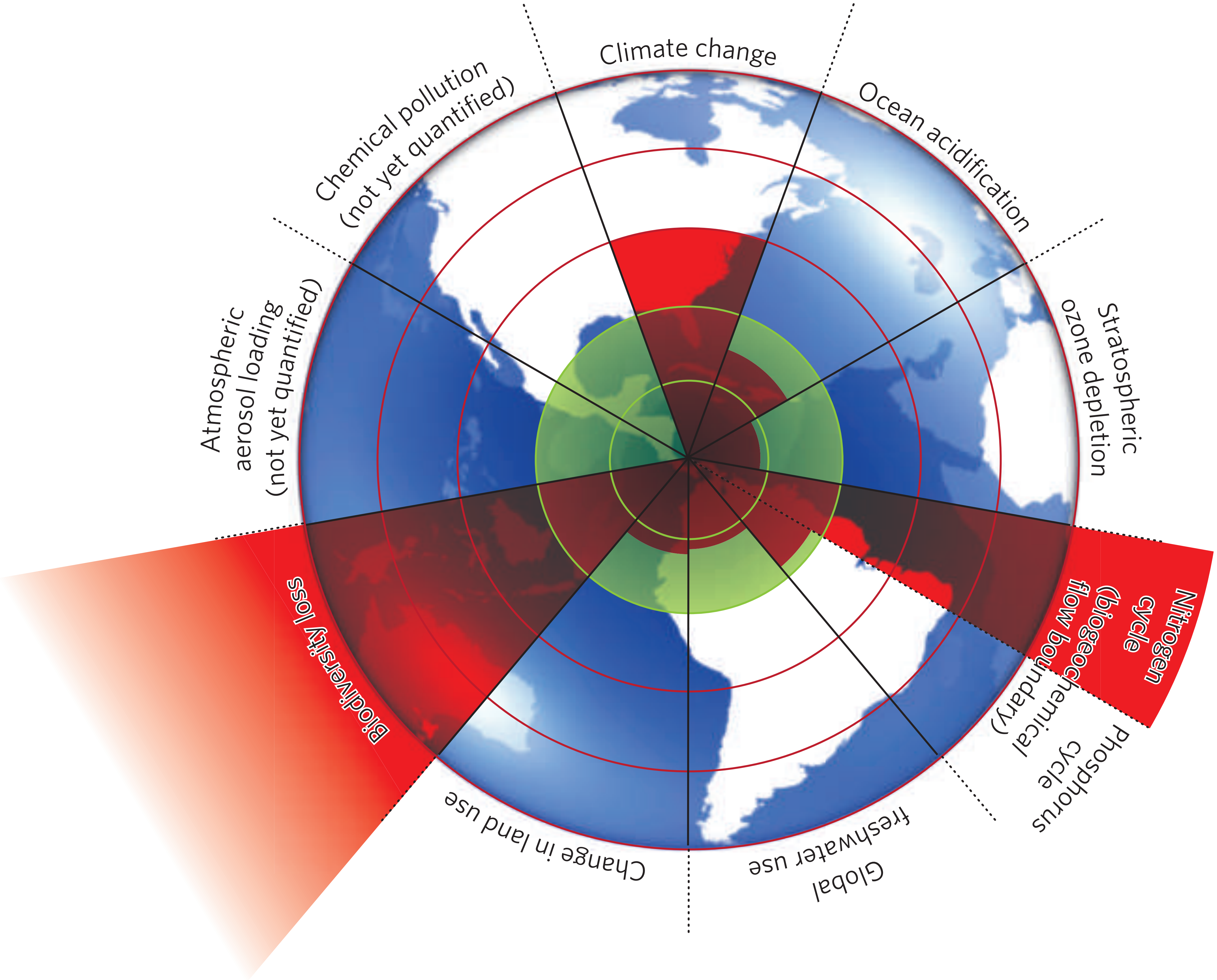
Planetary boundaries

A safe operating space for humanity

Identifying and quantifying planetary boundaries that must not be transgressed could help prevent human activities from causing unacceptable environmental change, argue **Johan Rockström** and colleagues.



Climate change
Ocean acidification
Stratospheric ozone depletion
Nitrogen cycle (biogeochemical flow boundary)
Phosphorus cycle
Global freshwater use
Change in land use
Biodiversity loss
Atmospheric aerosol loading (not yet quantified)
Chemical pollution (not yet quantified)

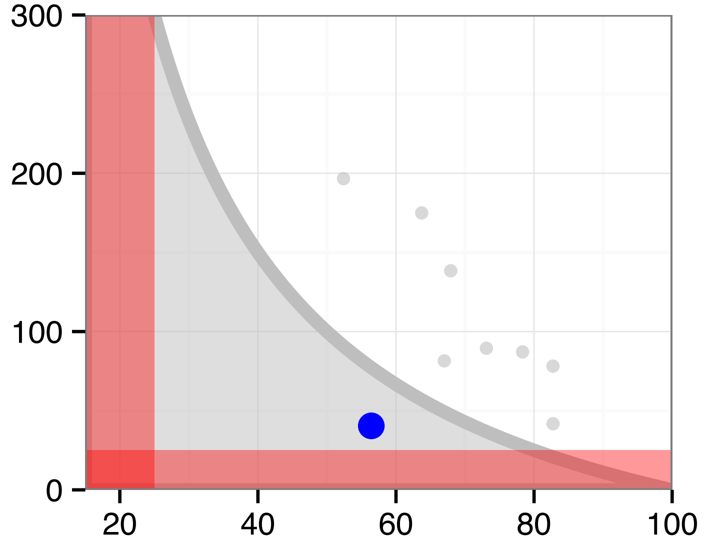

Climate change
Ocean acidification
Stratospheric ozone depletion
Nitrogen cycle (biogeochemical flow boundary)
Phosphorus cycle
Global freshwater use
Change in land use
Biodiversity loss
Atmospheric aerosol loading (not yet quantified)
Chemical pollution (not yet quantified)

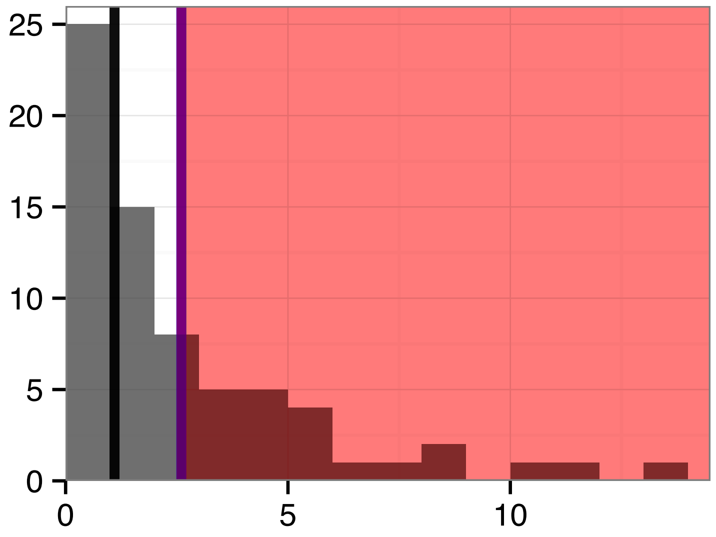
ixio ANALYTICS

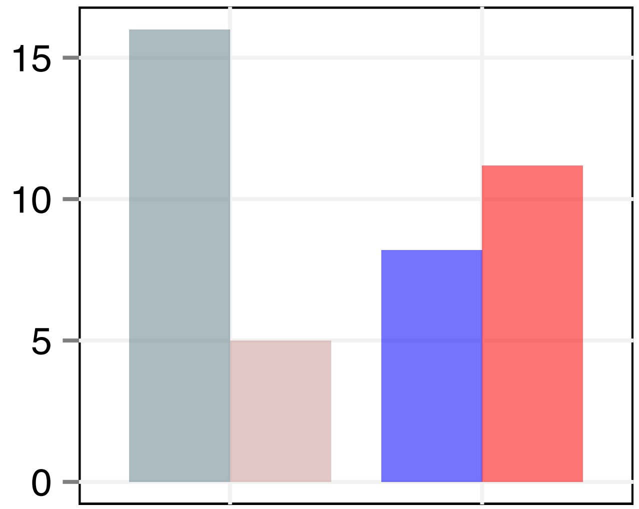
Application to currency traders

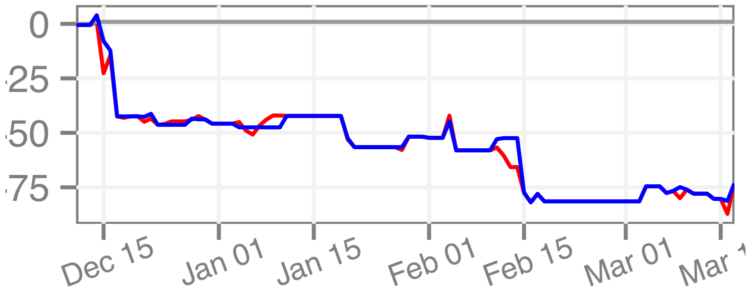
- Malaysian bank providing online forex trading facility
- Attempting to encourage better trading behavior for clients
- Asked to automate report generation and encourage behavioral change

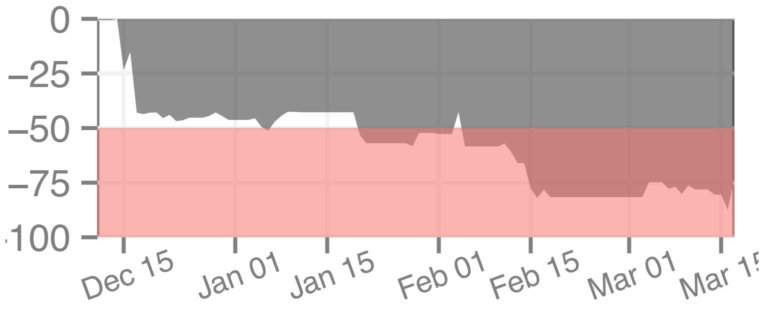
Comparing trading behaviour to rules

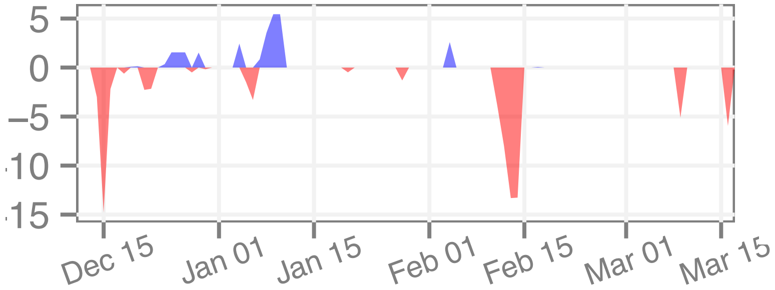






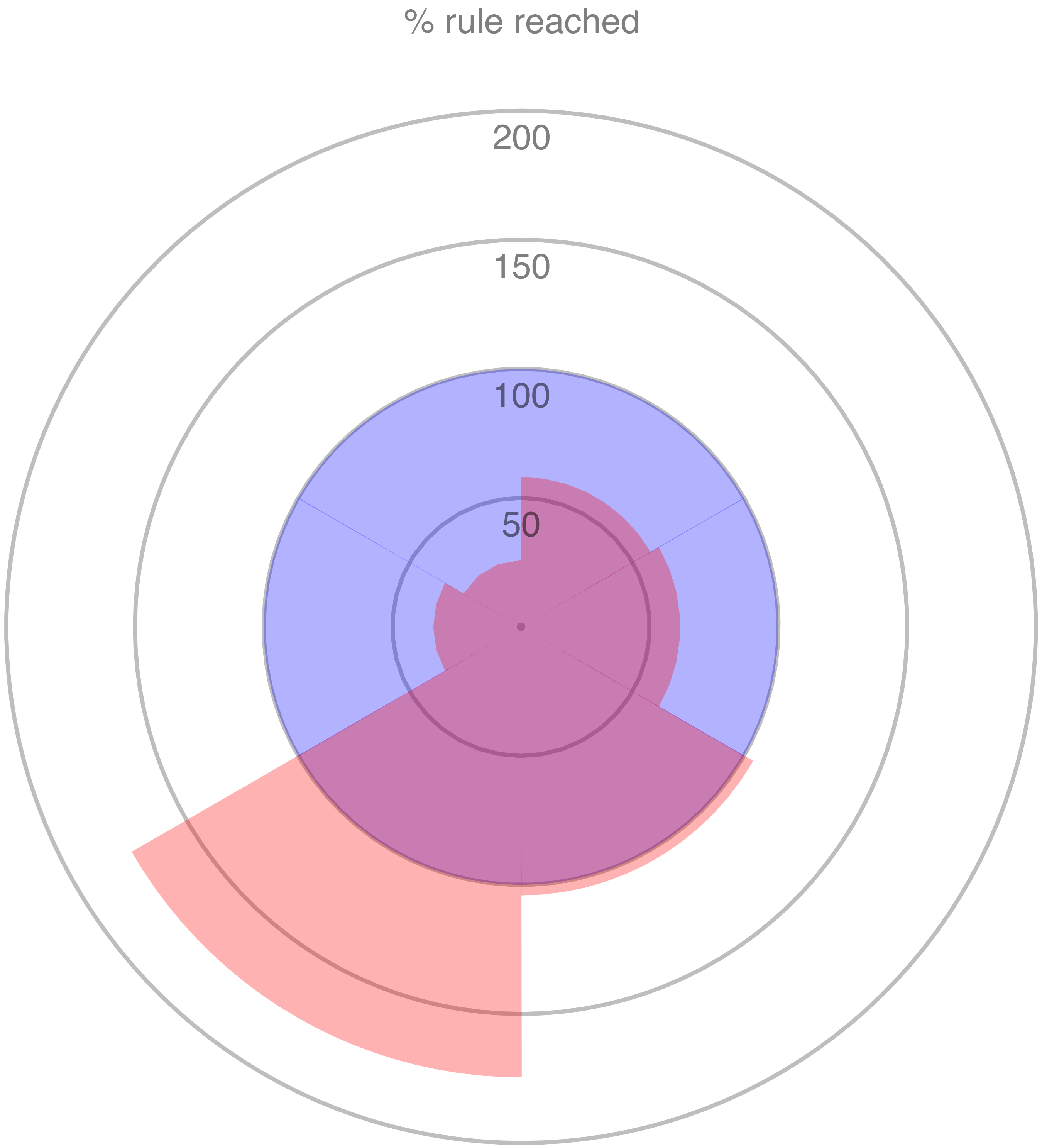






ixio ANALYTICS

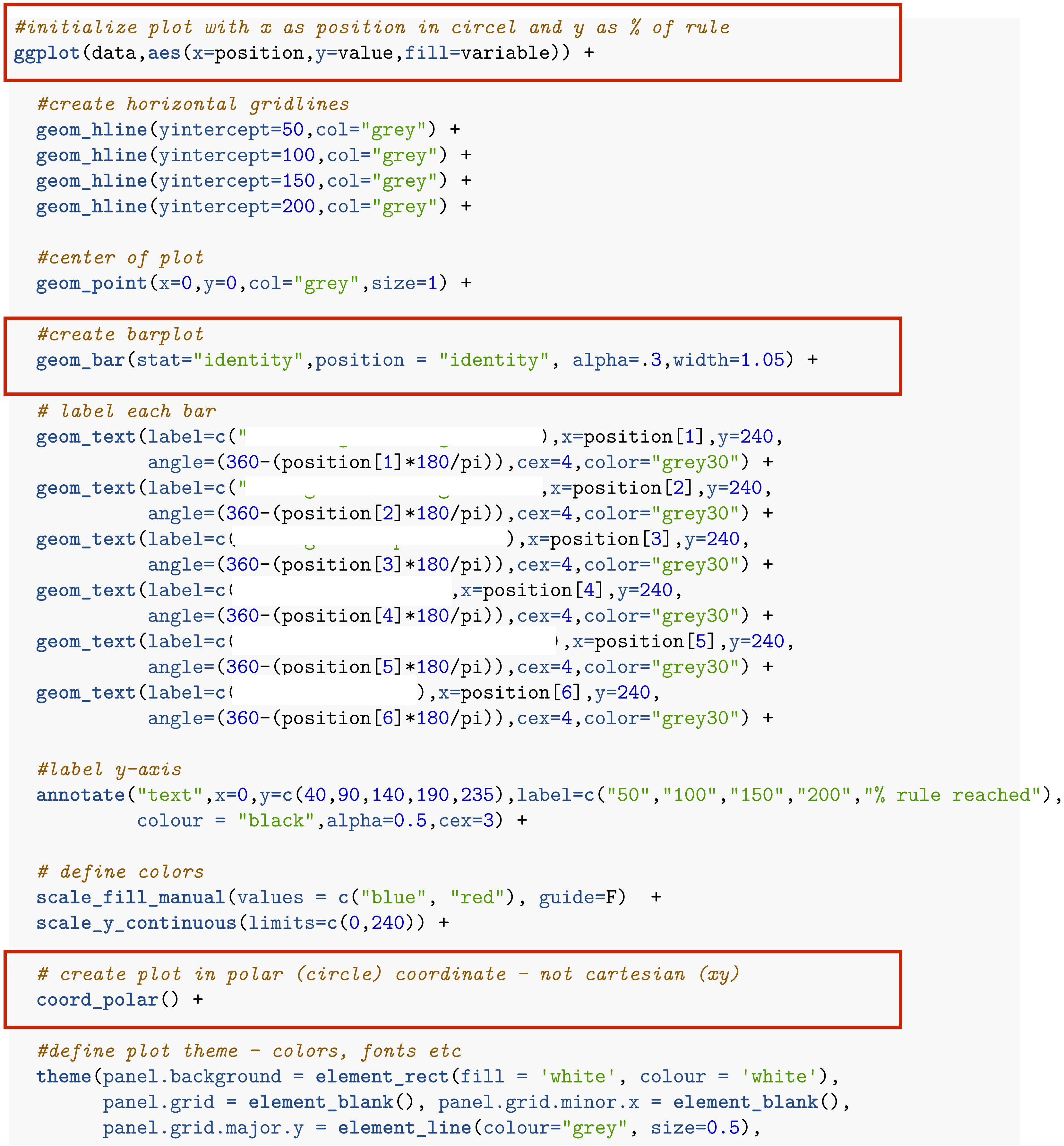
Summarising thresholds



Summarising thresholds

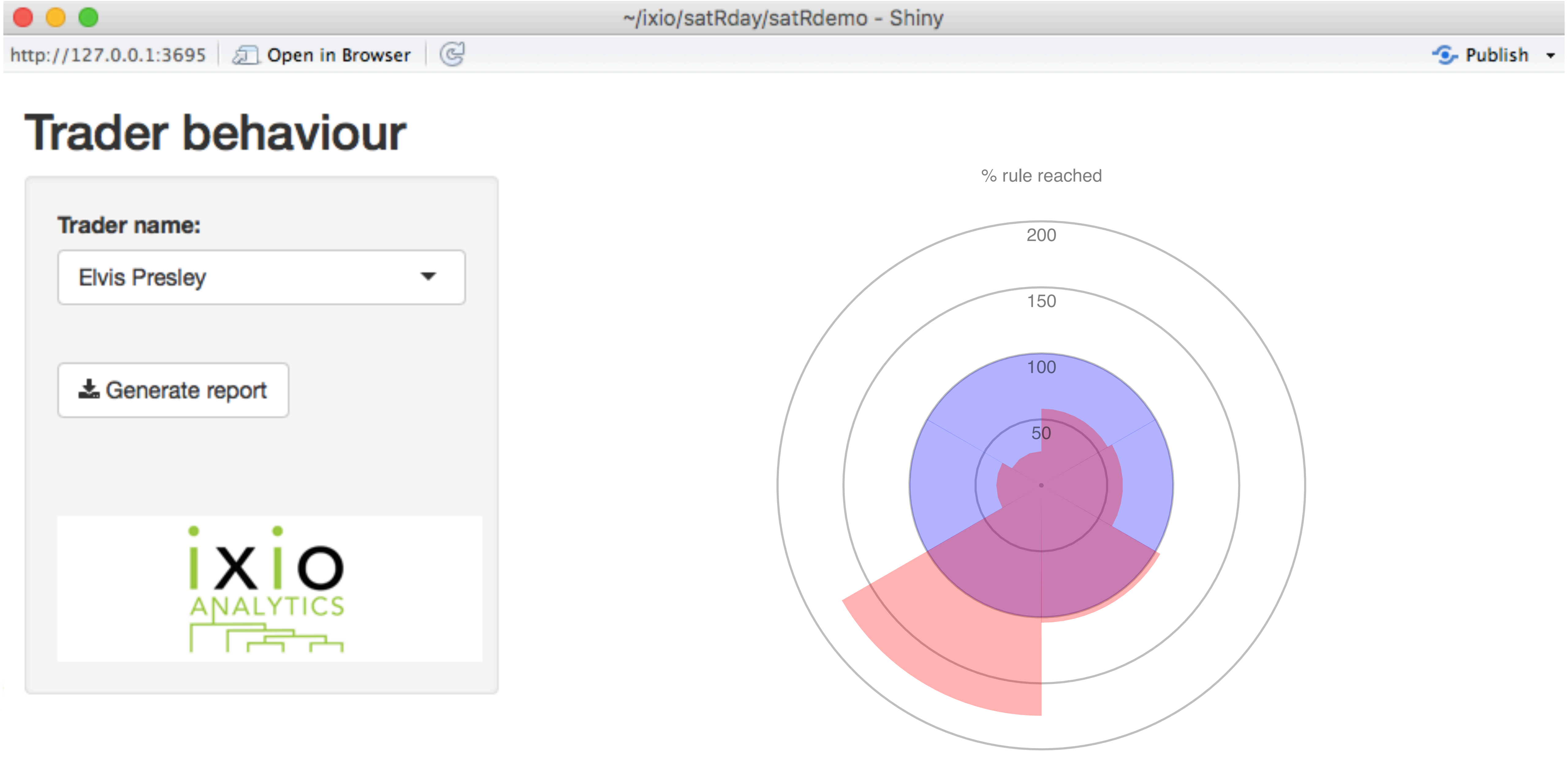
```
#initialize plot with x as position in circel and y as % of rule
ggplot(data,aes(x=position,y=value,fill=variable)) +

  #create horizontal gridlines
  geom_hline(yintercept=50,col="grey") +
  geom_hline(yintercept=100,col="grey") +
  geom_hline(yintercept=150,col="grey") +
  geom_hline(yintercept=200,col="grey") +

  #center of plot
  geom_point(x=0,y=0,col="grey",size=1) +

  #create barplot
  geom_bar(stat="identity",position = "identity", alpha=.3,width=1.05) +

  # label each bar
  geom_text(label=c("Percentage winning trades"),x=position[1],y=240,
            angle=(360-(position[1]*180/pi)),cex=4,color="grey30") +
  geom_text(label=c("Average win/Average loss"),x=position[2],y=240,
            angle=(360-(position[2]*180/pi)),cex=4,color="grey30") +
  geom_text(label=c("Average loss per trade"),x=position[3],y=240,
            angle=(360-(position[3]*180/pi)),cex=4,color="grey30") +
  geom_text(label=c("Maximum drawdown"),x=position[4],y=240,
            angle=(360-(position[4]*180/pi)),cex=4,color="grey30") +
  geom_text(label=c("Loss duration/Win duration"),x=position[5],y=240,
            angle=(360-(position[5]*180/pi)),cex=4,color="grey30") +
  geom_text(label=c("Unrealized P&L"),x=position[6],y=240,
            angle=(360-(position[6]*180/pi)),cex=4,color="grey30") +

  #label y-axis
  annotate("text",x=0,y=c(40,90,140,190,235),label=c("50","100","150","200","% rule reached"),
           colour = "black",alpha=0.5,cex=3) +

  # define colors
  scale_fill_manual(values = c("blue", "red"), guide=F)  +
  scale_y_continuous(limits=c(0,240)) +

  # create plot in polar (circle) coordinate - not cartesian (xy)
  coord_polar() +

  #define plot theme - colors, fonts etc
  theme(panel.background = element_rect(fill = 'white', colour = 'white'),
        panel.grid = element_blank(), panel.grid.minor.x = element_blank(),
        panel.grid.major.y = element_line(colour="grey", size=0.5),
```

Summarising thresholds



```
#initialize plot with x as position in circel and y as % of rule
ggplot(data,aes(x=position,y=value,fill=variable)) +

  #create horizontal gridlines
  geom_hline(yintercept=50,col="grey") +
  geom_hline(yintercept=100,col="grey") +
  geom_hline(yintercept=150,col="grey") +
  geom_hline(yintercept=200,col="grey") +

  #center of plot
  geom_point(x=0,y=0,col="grey",size=1) +

  #create barplot
  geom_bar(stat="identity",position = "identity", alpha=.3,width=1.05) +

  # label each bar
  geom_text(label=c("[redacted]),x=position[1],y=240,
            angle=(360-(position[1]*180/pi)),cex=4,color="grey30") +
  geom_text(label=c("[redacted],x=position[2],y=240,
            angle=(360-(position[2]*180/pi)),cex=4,color="grey30") +
  geom_text(label=c([redacted]),x=position[3],y=240,
            angle=(360-(position[3]*180/pi)),cex=4,color="grey30") +
  geom_text(label=c([redacted],x=position[4],y=240,
            angle=(360-(position[4]*180/pi)),cex=4,color="grey30") +
  geom_text(label=c([redacted]),x=position[5],y=240,
            angle=(360-(position[5]*180/pi)),cex=4,color="grey30") +
  geom_text(label=c([redacted]),x=position[6],y=240,
            angle=(360-(position[6]*180/pi)),cex=4,color="grey30") +

  #label y-axis
  annotate("text",x=0,y=c(40,90,140,190,235),label=c("50","100","150","200","% rule reached"),
           colour = "black",alpha=0.5,cex=3) +

  # define colors
  scale_fill_manual(values = c("blue", "red"), guide=F)  +
  scale_y_continuous(limits=c(0,240)) +

  # create plot in polar (circle) coordinate - not cartesian (xy)
  coord_polar() +

  #define plot theme - colors, fonts etc
  theme(panel.background = element_rect(fill = 'white', colour = 'white'),
        panel.grid = element_blank(), panel.grid.minor.x = element_blank(),
        panel.grid.major.y = element_line(colour="grey", size=0.5),
```

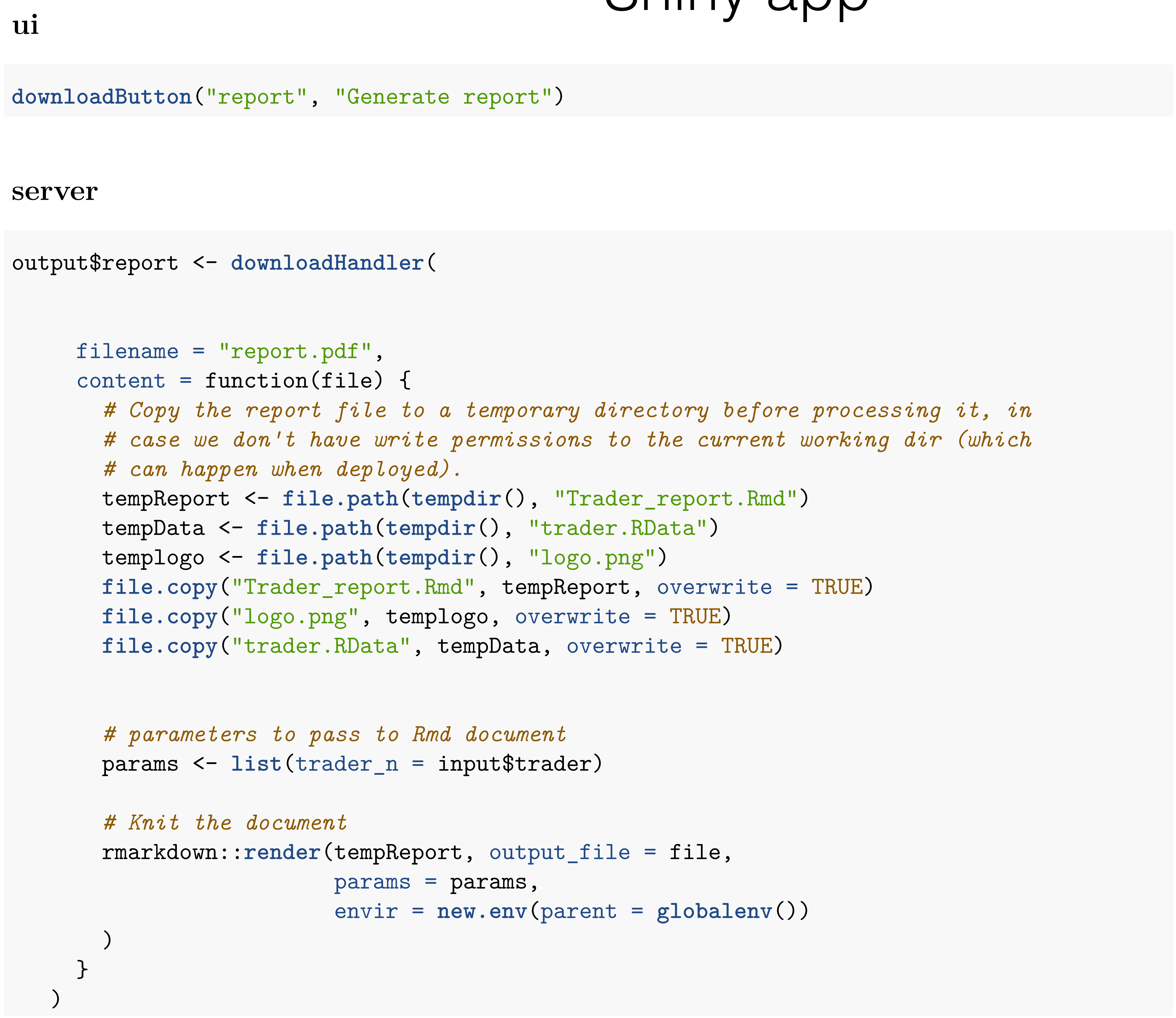
Shiny app



Trader behaviour

Trader name: Elvis Presley

Shiny app



ui

```
downloadButton("report", "Generate report")
```

server

```
output$report <- downloadHandler(


     filename = "report.pdf",
     content = function(file) {
       # Copy the report file to a temporary directory before processing it, in
       # case we don't have write permissions to the current working dir (which
       # can happen when deployed).
       tempReport <- file.path(tempdir(), "Trader_report.Rmd")
       tempData <- file.path(tempdir(), "trader.RData")
       templogo <- file.path(tempdir(), "logo.png")
       file.copy("Trader_report.Rmd", tempReport, overwrite = TRUE)
       file.copy("logo.png", templogo, overwrite = TRUE)
       file.copy("trader.RData", tempData, overwrite = TRUE)


       # parameters to pass to Rmd document
       params <- list(trader_n = input$trader)

       # Knit the document
       rmarkdown::render(tempReport, output_file = file,
                         params = params,
                         envir = new.env(parent = globalenv())
       )
     }
   )
```