FIFA Shiny App

Project Group 1

Maximilian Becker
Stefan Ilic
Romano Girardi
Tobias Schär

Introduction

As we discussed what data we want to analyse and which questions we want to ask ourselves we came to the following research questions we had in our project proposal.

- Which section of performances do typically decrease/increase with age?
- How is heading precision related to weight/height and other factors?
- How does salary and the overall rating correlate?
- Which positions do have which average stats (preferred performance factors)

This and an additional question will be answered in the following document. Furthermore, we explain how we did this in our Shiny app with the corresponding code excerpts.

Overview

As you will see, we opted for a tab layout and a bootstrap css we took from bootswatch.org.

Every code excerpt from this document from the server part (where the calculations are done) only. We opted out of showing the UI here since there are only the Inputs (i.e.: Slider) and text paragraphs anyway.

In the Tab “Fifa Tabelle” is the raw dataset where you can get a general feeling what data we’re dealing with.

Code:

```r
fifaDataRaw <- read.csv("data/data.csv",stringsAsFactors=FALSE,encoding="UTF-8")

fifaDataRaw <- as_tibble(fifaDataRaw)

fifaDataRaw <- fifaDataRaw %>%
    separate(Height, c("HFeet", "HInches"), "'")
fifaDataRaw <- fifaDataRaw %>%
    mutate(HeightMetric = ((as.numeric(HFeet)*30) + (as.numeric(HInches)*2.54)) )

fifaDataRaw <- fifaDataRaw %>%
    separate(Weight, c("Weight"), "lbs",convert=TRUE)
fifaDataRaw <- fifaDataRaw %>%
    mutate(WeightMetric = (as.numeric(Weight)/2.2) )

#Sperate the Value and Wage column with the euro sign and multiplicator (either M or K)
fifaDataRaw <- fifaDataRaw %>%
  separate(Wage,c(NA,"Wage"),"€",convert=TRUE)
fifaDataRaw <- fifaDataRaw %>%
  separate(Wage,c("Wage"),"K",convert=TRUE)
fifaDataRaw <- fifaDataRaw %>%
  separate(Value,c(NA,"Value"),"€",convert=TRUE)
fifaDataRaw <- fifaDataRaw %>%
  separate(Value,c("Value", "multiplicator"), "(?<=[0-9])(?=[A-Z])")
fifaDataRaw <- fifaDataRaw %>%
  mutate(
    Value = ifelse(multiplicator=="K",as.numeric(Value)*1000,
                   ifelse(multiplicator=="M",as.numeric(Value)*1000000,0))
  )

newdata <- (fifaDataRaw %>% filter(Wage > quantile(Wage , 0.25 )))

newdata <- newdata[order(newdata$Wage),]
head(newdata)

print(newdata$Wage)

fifaDataRaw$WeightMetric <- as.numeric(as.character(fifaDataRaw$WeightMetric))
fifaDataRaw$HeightMetric <- as.numeric(as.character(fifaDataRaw$HeightMetric))
fifaDataRaw$HeadingAccuracy <- as.numeric(as.character(fifaDataRaw$HeadingAccuracy))

str(fifaDataRaw)

#Make Dataset Reactive
fifaData <- reactive({
  fifaData=fifaDataRaw
})
```

First the dataset is read and saved into a variable and converted to a Tibble.

Then the columns for height is separated since the values were shown in an American format (i.e. 5’2”) and converted into metric in a new column afterwards.

The same transformation was done for the weight.

Then, the Value had to be separated from the Euro-sign and the multiplicator had to be taken out for later calculations.

Q1: How is heading precision related to weight/height and other factors?

Interactive Fifa-Data Analysis

Select a tab which analysis of the Fifa Data of 2019 you want to see

- Fifa Tabelle
- Heading Precision Relations
- Correlation of Overall Rating and Salary
- Which section of performances do typically decrease/increase with age?
- Which positions do have which average stats (preferred performance factors)
- Value Prediction

In order to visualize whether the heading performance is closely related to weight and height we need to see if these factors correlate. The easiest way to do that is by having a relationship between these factors. Luckily there already is an established factor for these values - it's called the BMI. So we added a column to our dataset and calculated each BMI for every player. The heading performance is on a scale from 0-100

Show 10 entries Search:

	HeadingAccuracy	WeightMetric	HeightMetric	BMI
103	94	91.3636363636364	195.24	23.9682197947171
205	94	89.0909090909091	192.7	23.9921938214764
500	93	86.3636363636364	185.08	25.2122693675547
701	93	85	190.16	23.5061004801837
819	93	85	185.08	24.8141809038565
13	92	78.1818181818182	185.08	22.8237385853653
317	92	78.1818181818182	180	24.1301907968575
725	92	82.2727272727273	190.16	22.7518940476645
9	91	82.2727272727273	180	25.3928170594837

Code (without the text paragraphs):

```
##########################################
#Heading Performance
#Add BMI. BMI is a relationship between Weight and Height
headPerf <- fifaDataRaw[, c("HeadingAccuracy","WeightMetric","HeightMetric")]
headPerf <- headPerf %>% mutate(BMI = as.numeric(WeightMetric)/(as.numeric(HeightMetric)/100)^2)
#Output
output$headingPerfTable <- renderDataTable(
  headPerf
)
headPerf.agg <- aggregate(.~HeadingAccuracy,headPerf,FUN=mean,na.rm=TRUE,na.action=NULL)
#Output
output$headingPerfCorr <- renderPlot(
  ggcorr(headPerf.agg,label=TRUE,hjust=1,size=5)
)
#Output and gradient color
output$headingPerf <- renderPlot({
  palHP <- wes_palette("Zissou1", 100, type = "continuous")
  ggplot(fifaData(), aes(x = WeightMetric, y = HeightMetric, fill = HeadingAccuracy)) +
    geom_tile() +
    scale_fill_gradientn(colours = palHP) +
    coord_equal()+
    xlab("Weight in kilogramms")+
    ylab("Height in meters")+
    geom_point(aes(x=input$weightM,y=input$heightM),size=3,colour="purple")
})
```

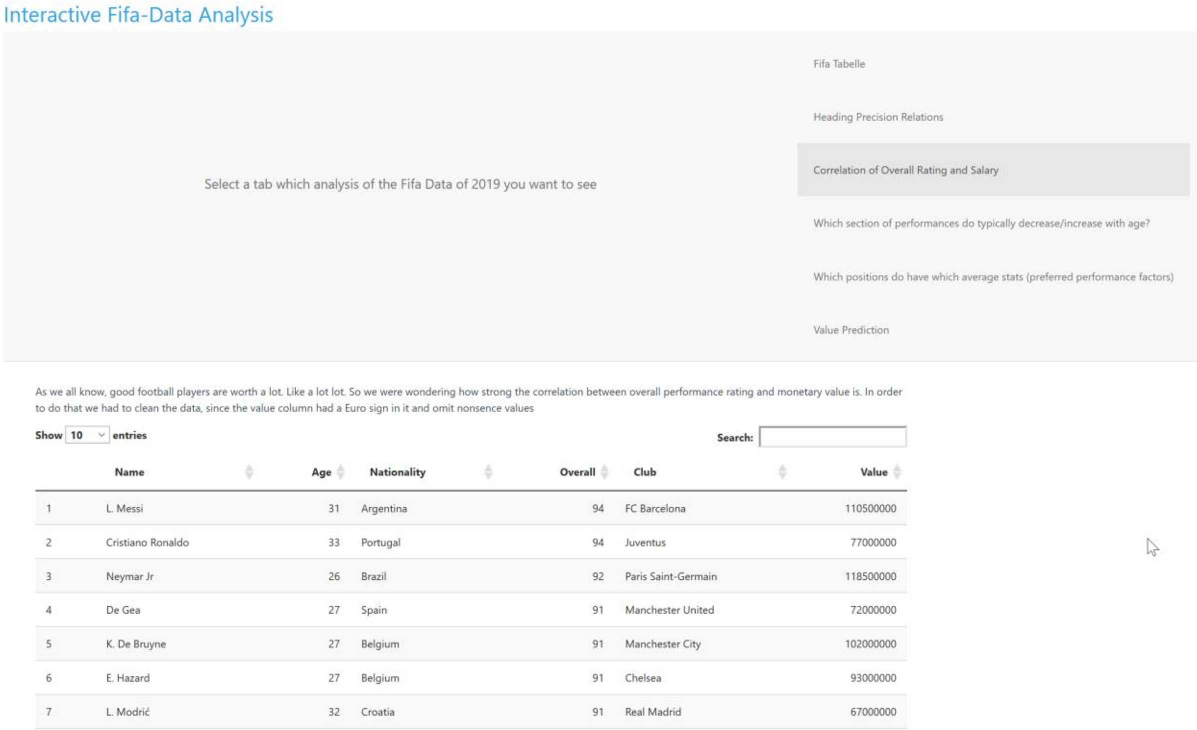
Here an additional column is added to the dataset. The column BMI is calculated and added for each row.

Then the table is aggregated to the HeadingAccuracy. We explicitly chose the mean function to have the outliers calculated too.

The correlation graph (ggcorr) is being put out.

The colour palette is being loaded (from a library) and the tile graph is created.

Q2: How does salary and the overall rating correlate?



	Name	Age	Nationality	Overall	Club	Value
1	L. Messi	31	Argentina	94	FC Barcelona	110500000
2	Cristiano Ronaldo	33	Portugal	94	Juventus	77000000
3	Neymar Jr	26	Brazil	92	Paris Saint-Germain	118500000
4	De Gea	27	Spain	91	Manchester United	72000000
5	K. De Bruyne	27	Belgium	91	Manchester City	102000000
6	E. Hazard	27	Belgium	91	Chelsea	93000000
7	L. Modrić	32	Croatia	91	Real Madrid	67000000

Code:

```
##########################################################
#How does Value and the overall rating correlate ?
output$valueTable <- renderDataTable(
  datatable(fifaData()[,c("Name","Age","Nationality","Overall","Club","Value")]
  )
)

valueRat <- fifaDataRaw[,c("Name","Age","Nationality","Overall","Club","Value","Wage")]
valueRat <- valueRat %>%
  mutate( topTen=
    case_when(
      Nationality == "Belgium" ~ TRUE,
      Nationality == "France" ~ TRUE,
      Nationality == "Brazil" ~TRUE,
      Nationality == "England" ~TRUE,
      Nationality == "Uruguay" ~TRUE,
      Nationality == "Croatia" ~TRUE,
      Nationality == "Portugal" ~TRUE,
      Nationality == "Spain" ~TRUE,
      Nationality == "Argentina" ~TRUE,
      Nationality == "Colombia" ~TRUE,
      TRUE ~ FALSE
      )
  )

valueRatR <- reactive({
  valueRatR=valueRat
})

Value.agg <- aggregate(.~Overall, valueRat[,c("Overall","Value","Wage")], FUN=mean, na.rm=TRUE, na.action=NULL)
output$corrGraphValue <- renderPlot(
  ggcorr(Value.agg,label=TRUE)
)
```

Here the raw data is first being put out with the relevant columns.

Then an additional column is added to either see from the nationality of a player whether it is a top ten country or not.

Afterwards, the data is being made reactive.

Then the data is being aggregated to the Overall-Rating.

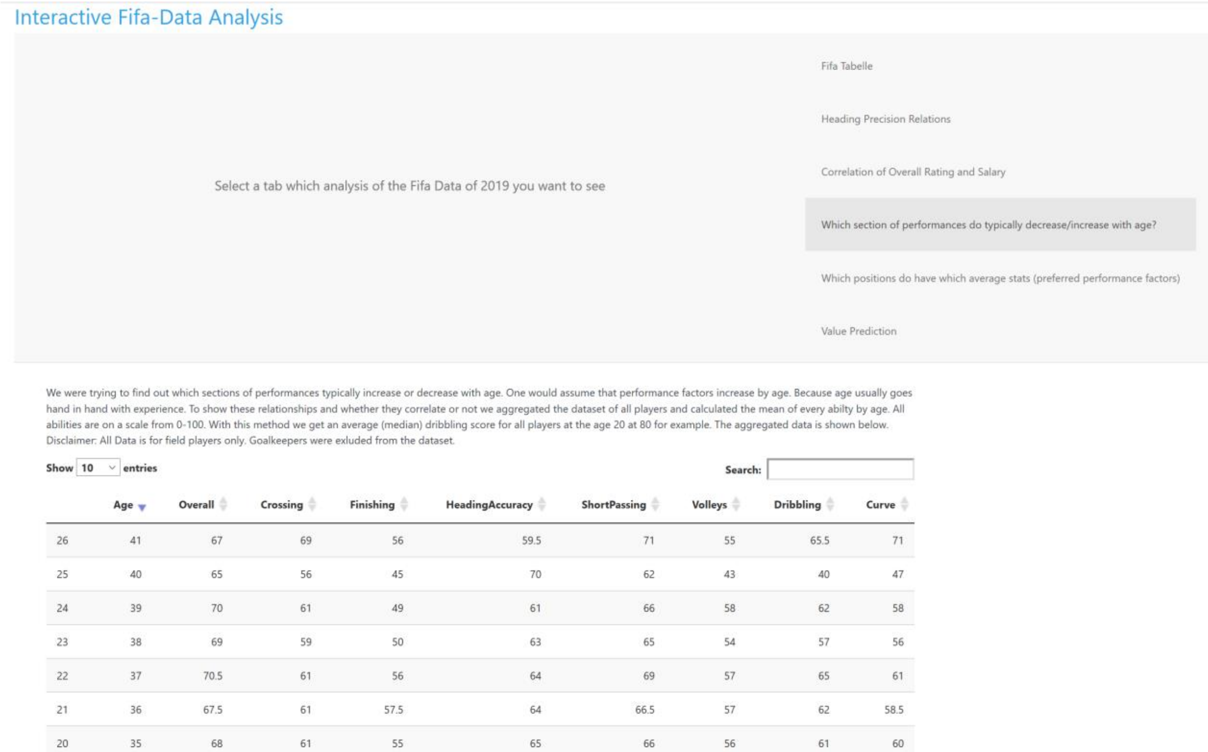
```
output$ratingValue <- renderPlot({
  ggplot(valueRatR() %>% filter(Value>quantile(Value,input$valueQuant/100,na.rm= TRUE)),
         aes(x=Value,y=Overall,size=Wage,color=ifelse(topTen==TRUE,Nationality,"other")))+
    geom_point(alpha=0.5)+
    scale_fill_viridis(discrete=TRUE, guide=FALSE, option="A")+
    theme_ipsum()+
    labs(colour="Fifa Top Ten (2020)",size="Wage in thousands")+
    scale_x_continuous(labels=comma,trans='log10')
})

output$densGraph <- renderPlot(
  ggplot(
    valueRatR()
    %>%
    filter(
      case_when(
        input$selectValueWage == "Value" ~ Value>quantile(Value,input$valueQuant2/100,na.rm=TRUE),
        input$selectValueWage == "Wage" ~ Wage>quantile(Wage,input$valueQuant2/100,na.rm=TRUE)
      )
      )
    ,aes_string(input$selectValueWage))+
    geom_point(aes(y=Overall),alpha=0.5)+
    theme_ipsum()+
    geom_density_2d(aes(y=Overall),colour="red")+
    scale_x_continuous(labels=comma,trans='log10')
)
```

The plot is being generated with the chosen quantile input and the colours for the countries, the size for the wage and the points for the value.

The second graph is created with the chosen points and a density graph.

Q3: Which section of performances do typically decrease/increase with age?


Interactive Fifa-Data Analysis

Select a tab which analysis of the Fifa Data of 2019 you want to see

Fifa Tabelle

Heading Precision Relations

Correlation of Overall Rating and Salary

Which section of performances do typically decrease/increase with age?

Which positions do have which average stats (preferred performance factors)

Value Prediction

We were trying to find out which sections of performances typically increase or decrease with age. One would assume that performance factors increase by age. Because age usually goes hand in hand with experience. To show these relationships and whether they correlate or not we aggregated the dataset of all players and calculated the mean of every abilty by age. All abilities are on a scale from 0-100. With this method we get an average (median) dribbling score for all players at the age 20 at 80 for example. The aggregated data is shown below. Disclaimer: All Data is for field players only. Goalkeepers were exluded from the dataset.

Show 10 entries

Search:

	Age	Overall	Crossing	Finishing	HeadingAccuracy	ShortPassing	Volleys	Dribbling	Curve
26	41	67	69	56	59.5	71	55	65.5	71
25	40	65	56	45	70	62	43	40	47
24	39	70	61	49	61	66	58	62	58
23	38	69	59	50	63	65	54	57	56
22	37	70.5	61	56	64	69	57	65	61
21	36	67.5	61	57.5	64	66.5	57	62	58.5
20	35	68	61	55	65	66	56	61	60

Code:

```r
#####################################################
#Question Which section of Performance....
#Omit Goalkeepers and select Age and Performance Factors
AgeStats <-  fifaDataRaw %>% filter(Position != "GK")
AgeStats <- AgeStats[,c("Age","Overall","Crossing","Finishing","HeadingAccuracy",
            "ShortPassing","Volleys","Dribbling","Curve")]

#AgeStats <- AgeStats %>% filter(Age < quantile(Age , 0.90 ))
#Aggregate the mean of every ability by age
AgeStats.agg <- aggregate(.~Age, AgeStats, FUN=median, na.rm=TRUE, na.action=NULL)
#Output
output$AgeStatsTable <- renderDataTable(
  AgeStats.agg
)
#Correlation matrix
AgeStats.corr <-cor(AgeStats.agg)
#Output
output$AgeStatsCorr1 <- renderDataTable(
  AgeStats.corr
)
paletteHM1 = colorRampPalette(brewer.pal(3, "BuGn"))(20)
#Output
output$HeatmapCorr1 <- renderPlot(
  heatmap(x = AgeStats.corr, col = paletteHM1, symm = TRUE)
)
#print(AgeStats[order(-AgeStats$Age),])
#Correlation Matrix only by age, splitting the dataframe and putting
#it back together as tibble in order to create the geom_point graph
correlationAge <- AgeStats.corr[,'Age']
colAge <- colnames(AgeStats.corr)
Age.corr <- tibble(colAge, correlationAge)
#Output
output$CorrelationGraph1 <- renderPlot(
ggplot(data=Age.corr) +
  geom_point(aes(x=colAge, y=correlationAge)) + geom_hline(yintercept=0, color="blue", size=2)+
  theme(axis.text.x = element_text(size = 20, angle = 45, hjust = 1)) +
  xlab("Ability")+
  ylab("Corrleation Value")
)
```

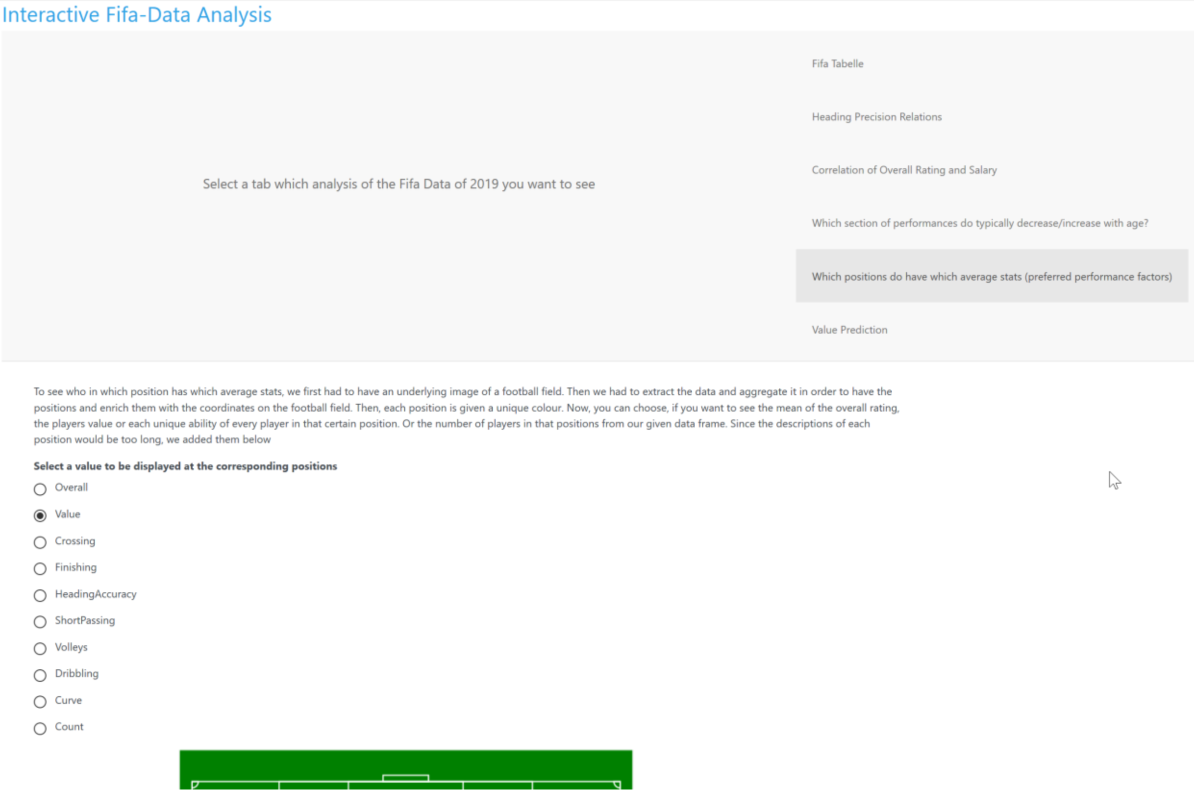
Again, the dataset is being reduced for performance and only the relevant columns.

There is a lot going on here but basically the data is being aggregated again and put into correlation matrices and correlation graphs are being created.

```r
#Output
output$CorrelationGraph2 <- renderPlot(
ggcorr(AgeStats.agg,label=TRUE,hjust=1,size=5)
)
#The mean of every ability by 'Overall' stat
OveStats.agg <- aggregate(.~Overall, AgeStats, FUN=median, na.rm=TRUE, na.action=NULL)
#Output
output$OveStatsTable <- renderDataTable(
  OveStats.agg
)
#Correlation matrix and heatmap
OveStats.corr <-cor(OveStats.agg)
#Output
output$OveStatsCorr1 <- renderDataTable(
  OveStats.corr
)
# #Output
# paletteHM2 = colorRampPalette(brewer.pal(3, "BuGn"))(20)
# output$HeatmapCorr2 <- renderPlot(
# heatmap(x = OveStats.corr, col = paletteHM2, symm = TRUE)
# )
#Output
output$CorrelationGraph3 <- renderPlot(
  ggcorr(OveStats.agg,label=TRUE,hjust=1,size=5)
)
print(OveStats.corr)
#Correlation data by Overall and create tibble again in order to create the geom_point
correlationOve <- OveStats.corr[,'Overall']
col <- colnames(OveStats.corr)
Ove.corr <- tibble(col, correlationOve)
#Output
output$CorrelationGraph4 <- renderPlot(
  ggplot(data=Ove.corr) +
    geom_point(aes(x=col, y=correlationOve)) + geom_hline(yintercept=0, color="blue", size=2)+
    theme(axis.text.x = element_text(size = 20, angle = 45, hjust = 1))+
    xlab("Ability")+
    ylab("Correlation Value")
)

#Output
output$AgeOverall <- renderPlot(
ggplot(AgeStats.agg,aes(x=Age,y=Overall))+
  geom_line()+
  geom_point(aes(x=AgeStats.agg[which.max(Overall),1],
                 y=AgeStats.agg[which.max(Overall),2]),
             col="red",size=3,show.legend=TRUE)+
  geom_text(aes(label=ifelse(Overall==AgeStats.agg[which.max(Overall),2],
                             round(Overall),'')),hjust=1,vjust=2,color="red",size=5,fontface="bold")+
  geom_text(aes(label=ifelse(Overall==AgeStats.agg[which.max(Overall),2],
                             Age,'')),hjust=1,vjust=4,color="blue",size=5,fontface="bold")
)
```

Q4: Which positions do have which average stats (preferred performance factors)



Code:

```
###################################
##Stats on preferred position
posData <- fifaDataRaw[,c("Position","Overall","Value","Crossing",
                          "Finishing","HeadingAccuracy","ShortPassing",
                          "Volleys","Dribbling","Curve"
                          )]

#Adding count column for later use in aggregat
posData <- posData %>% mutate(count=1)

#count the occurences of each position
posData.temp1 <- aggregate(count ~ Position, posData,sum,na.rm=TRUE)
#aggregate the mean of each other column
posData.temp2 <- aggregate(. ~ Position,posData,mean)
#omit the empty positions
posData.temp1 <- posData.temp1 %>% filter(Position != "")
#set count column to null
posData.temp2$count <- NULL
#merging the aggregated dataframes
posData.agg <- merge(posData.temp1,posData.temp2)
```

- Only relevant Data
- The data is being counted and filtered

```
posData.agg <- posData.agg %>%
  mutate(
    posCoordX =
      case_when(
        Position == "LF" ~ 20,
        Position == "LB" ~ 20,
        Position == "LCB" ~ 40,
        Position == "LCM" ~ 40,
        Position == "LAM" ~ 40,
        Position == "LDM" ~ 40,
        Position == "LM" ~ 20,
        Position == "LS" ~ 20,
        Position == "LW" ~ 20,
        Position == "LWB" ~ 20,
        Position == "RB" ~ 80,
        Position == "RWB" ~ 80,
        Position == "RCB" ~ 60,
        Position == "RCM" ~ 60,
        Position == "RAM" ~ 60,
        Position == "RDM" ~ 60,
        Position == "RM" ~ 80,
        Position == "RS" ~ 80,
        Position == "RF" ~ 80,
        Position == "RW" ~ 80,
        Position == "CF" ~ 50,
        Position == "CAM" ~ 50,
        Position == "CB" ~ 50,
        Position == "CDM" ~ 50,
        Position == "CM" ~ 50,
        Position == "ST" ~ 50,
        Position == "GK" ~ 50,
        TRUE ~ 0
      )
    posCoordY =
      case_when(
        Position == "LF" ~ 160,
        Position == "LB" ~ 80,
        Position == "LCB" ~ 60,
        Position == "LCM" ~ 100,
        Position == "LAM" ~ 120,
        Position == "LDM" ~ 80,
        Position == "LM" ~ 100,
        Position == "LS" ~ 140,
        Position == "LW" ~ 150,
        Position == "LWB" ~ 60,
        Position == "RB" ~ 80,
        Position == "RWB" ~ 60,
        Position == "RCB" ~ 80,
        Position == "RCM" ~ 100,
        Position == "RAM" ~ 120,
        Position == "RDM" ~ 60,
        Position == "RM" ~ 100,
        Position == "RS" ~ 140,
        Position == "RF" ~ 160,
        Position == "RW" ~ 150,
        Position == "CF" ~ 160,
        Position == "CAM" ~ 120,
        Position == "CB" ~ 80,
        Position == "CDM" ~ 80,
        Position == "CM" ~ 100,
        Position == "ST" ~ 160,
        Position == "GK" ~ 20,
        TRUE ~ 0
      )
  )
```

- Here, the positions and their respective coordinates are being added to the dataset for later usage on the football pitch image/graph

```r
field <- readPNG("data/field2.png")
# grid <- rasterGrob(field, width=unit(1,"npc"), height=unit(1,"npc"))
grid <- rasterGrob(field, interpolate=TRUE, height = 1, width = 1)

# ggplot(posData.agg, aes(posCoordX,posCoordY,color=Position)) +
#   annotation_custom(grid) +
#   geom_point(aes(size=Overall)) +
#   scale_x_continuous(expand=c(0,0), lim=c(0,100)) +
#   scale_y_continuous(expand=c(0,0), lim=c(0,200)) +
#   theme_void() +
#   theme(aspect.ratio = nrow(field)/ncol(field))+
#   scale_fill_viridis(discrete=TRUE, guide=FALSE, option="A")+
#   guides(color=guide_legend(order=1),
#          size=guide_legend(order=2))

#Make aggregated Data reactive
posDataR <- reactive({
  posDataR=posData.agg
})

#make input reactive
fieldChoice <- reactive(input$fieldChoice)

#make graph reactive for later if statement (scaling of values)
graph <- reactive({
  ggplot(posDataR(), aes(posCoordX,posCoordY,color=Position)) +
    annotation_custom(grid) +
    geom_point(aes_string(size=fieldChoice())) +
    scale_x_continuous(expand=c(0,0), lim=c(0,100)) +
    scale_y_continuous(expand=c(0,0), lim=c(0,200)) +
    theme_void() +
    theme(aspect.ratio = nrow(field)/ncol(field),legend.key.size = unit(1, "cm"),legend.key.width = unit(1,"cm"))+
    # scale_fill_viridis(discrete=TRUE, guide=FALSE, option="A")+
    guides(color=guide_legend(order=1),
           size=guide_legend(order=2))
})
```

- Here the picture is being loaded for later usage in the graph
- The graph is being created with the custom grid and the picture
- The positions are added and the size of a ball is the chosen value

```r
output$avgStatsField <- renderPlot({
  if(fieldChoice()=="Value"){
    graph()+
      scale_size(range = c(0.1, 20),labels=comma)
  }else{
    graph()+
      scale_size(range = c(0.1, 15))
  }
})

#Input Position
choicePosR <- reactive({
 choicePosR=input$choicePosition
})

#Data for Spiderchart and Rownames
spidDat <-
  posData.agg[,c("Overall","Crossing","Finishing","HeadingAccuracy","ShortPassing","Dribbling","Curve")]
spidDat
rownames(spidDat) <- posData.agg$Position
spidDat

#First two lines have to have minimun and maximum values for spiderchart
spidDat <- rbind(rep(100,7), rep(0,7) ,spidDat)

spidDatR <- reactive({
  spidDatR = spidDat
})

colors_borderSpid <- brewer.pal(3, "Set3")
colors_inSpid <- alpha(colors_borderSpid,0.5)

output$posSpider <- renderPlot({
  radarchart(spidDatR()%>%filter(rownames(spidDatR()) %in% c(1,2,choicePosR())), axistype=1 ,
             #custom polygon
             pcol=colors_borderSpid , pfcol=colors_inSpid , plwd=4 , plty=1,
             #custom the grid
             cglcol="grey", cglty=1, axislabcol="grey", caxislabels=seq(0,10,100), cglwd=0.8,
             #custom labels
             vlcex=0.8)
             # Add a legend
             legend(x=-1.9, y=0, legend = choicePosR(), bty = "n", pch=20 , col=colors_borderSpid , text.col = "gre
})
```

- The spider graph is being created with the colour palette and the respective positions.
- Each Positions can be chosen to be added

Value Prediction

This was a tricky one. We chose a linear regression model and added the most correlated variables one by one and things got tricky:

```
##################################################
######--- Prediction Value
# Create training and test data

set.seed(69)
forecData <- fifaDataRaw[,c("Age","Value","Crossing","Finishing","HeadingAccuracy","ShortPassing","Vol
##Remove Value = NULL
forecData <- forecData %>% filter(Value!="")
##Remove outliers
outliers <- boxplot(forecData$Value, plot = TRUE)$out
x <- forecData
forecData <- x[-which(x$Value %in% outliers),]

#Train and Test Data
train.data<- sample_frac(forecData,0.7) #select 70% random samples
test.data <- setdiff(forecData,train.data)

#Create Correlation Table
forecData.corr<- cor(forecData,use = "complete.obs")
# view(forecData.corr[,c("Value")])

#View Correlations
ggcorr(forecData.corr,label=TRUE)
```

- Here again, only the relevant data. We wanted to find out which ability correlates most with the value and with those extrapolate a player's value.
- Here the massive outliers of the value have to be taken out in order to have usable data.
- Train and test data is being created.
- Correlation table is calculated.

```
#strating with the most correlated variable as predictor
#wage would be the the most correlated. but since wage and value are naturally
#dependend on each other (since the more value, the more wage)
#and it would be meaningless to predict ones value from the wage we go from the 2nd highest
lm1 <- lm(Value~ShortPassing,data=train.data )
summary(lm1)
##All Coefficients are positive
```

```
C:/HSLU/DASB/FIFA_Project/ShinyFifaApp/
lm(formula = Value ~ ShortPassing, data = train.data)

Residuals:
     Min       1Q   Median       3Q      Max
-1288042  -598332  -310361   196813  4921090

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)  -565401.8    37374.4  -15.13   <2e-16 ***
ShortPassing   26485.5      639.4   41.42   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 936700 on 10837 degrees of freedom
  (35 observations deleted due to missingness)
Multiple R-squared:  0.1367,	Adjusted R-squared:  0.1366
F-statistic:  1716 on 1 and 10837 DF,  p-value: < 2.2e-16
```

As soon as we added more variables, things got tricky:

In the next model, all variables were positive. But as soon as we added more, some became negative coefficients:

```
lm2 <- lm(Value~ShortPassing+Curve,data=train.data)
summary(lm2)
##All Coefficients are positive

lm3 <- lm(Value~ShortPassing+Curve+Dribbling,data=train.data)
```

689:5 (Untitled)

Console Terminal Jobs

C:/HSLU/DASB/FIFA_Project/ShinyFifaApp/

```
>     summary(lm2)

Call:
lm(formula = Value ~ ShortPassing + Curve, data = train.data)

Residuals:
     Min       1Q   Median       3Q      Max 
-1372195  -600885  -307900   198211  4854608 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  -516342.2    37802.7 -13.659  < 2e-16 ***
ShortPassing   20749.7      975.2  21.278  < 2e-16 ***
Curve           6132.6      788.8   7.774 8.27e-15 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 934100 on 10836 degrees of freedom
  (35 observations deleted due to missingness)
Multiple R-squared:  0.1415,	Adjusted R-squared:  0.1413 
F-statistic: 892.7 on 2 and 10836 DF,  p-value: < 2.2e-16
```

```r
lm3 <- lm(Value~ShortPassing+Curve+Dribbling,data=train.data)
summary(lm3)
##The Dribbling Coefficient is negative. Which means greater value effects the value negativels
```

689:5 (Untitled)

Console Terminal Jobs

C:/HSLU/DASB/FIFA_Project/ShinyFifaApp/

```
> summary(lm3)

Call:
lm(formula = Value ~ ShortPassing + Curve + Dribbling, data = train.data)

Residuals:
     Min       1Q   Median       3Q      Max
-1431066  -592067  -296076   196462  4850485

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  -534347.9    37943.9 -14.083  < 2e-16 ***
ShortPassing   23823.4     1160.9  20.521  < 2e-16 ***
Curve           8665.6      944.3   9.176  < 2e-16 ***
Dribbling      -5074.9     1042.6  -4.867 1.15e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 933100 on 10835 degrees of freedom
  (35 observations deleted due to missingness)
Multiple R-squared:  0.1433,	Adjusted R-squared:  0.1431
F-statistic: 604.3 on 3 and 10835 DF,  p-value: < 2.2e-16
```

```r
lm4 <- lm(Value~ShortPassing+Curve+Dribbling+Volleys,data=train.data)
summary(lm4)
##Dribbling Coeficcient is still negative

lm5 <- lm(Value~ShortPassing+Curve+Dribbling+Volleys+Crossing,data=train.data)
summary(lm5)
```

708:5 (Untitled)

Console Terminal Jobs

C:/HSLU/DASB/FIFA_Project/ShinyFifaApp/

```
>     summary(lm4)

Call:
lm(formula = Value ~ ShortPassing + Curve + Dribbling + Volleys, 
    data = train.data)

Residuals:
     Min       1Q   Median       3Q      Max
-1518744  -590341  -292105   190262  4832977

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  -546563.1    37915.5 -14.415  < 2e-16 ***
ShortPassing   24077.8     1159.3  20.769  < 2e-16 ***
Curve           6090.7     1020.4   5.969 2.47e-09 ***
Dribbling      -7796.8     1119.8  -6.963 3.53e-12 ***
Volleys         6331.9      961.9   6.583 4.84e-11 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 931300 on 10834 degrees of freedom
  (35 observations deleted due to missingness)
Multiple R-squared:  0.1467,	Adjusted R-squared:  0.1464
F-statistic: 465.8 on 4 and 10834 DF,  p-value: < 2.2e-16
```

- And so on.
- So we played around with the variables until we got a model with the most variables where none of them is a negative coefficient:

```r
#playing around with the least correlated variables and the variable which were a negative coefficient earli
lm.red <- lm(Value~.-HeightMetric-WeightMetric-Age-Dribbling-Crossing-Finishing,data=train.data)
summary(lm.red)
##Here HeadingAccuracy,ShortPassing,Volleys and Curve are coefficient and are positive

#let's cpompare the models obtained, in a structured way
anova(lm1, lm2, lm3, lm4,lm5, lm.red , lm.tot)
```

712:13 (Untitled) R Script

Console Terminal Jobs

C:/HSLU/DASB/FIFA_Project/ShinyFifaApp/

```
Call:
lm(formula = Value ~ . - HeightMetric - WeightMetric - Age -
    Dribbling - Crossing - Finishing, data = train.data)

Residuals:
     Min       1Q   Median       3Q      Max
-1365589  -600351  -316821   209310  4846585

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)     -529069.9    38016.9 -13.917  < 2e-16 ***
HeadingAccuracy    1812.0      709.8   2.553 0.010704 *
ShortPassing      18446.5     1145.5  16.103  < 2e-16 ***
Volleys            3327.6      919.5   3.619 0.000297 ***
Curve              4251.6      985.6   4.314 1.62e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 933100 on 10834 degrees of freedom
  (35 observations deleted due to missingness)
Multiple R-squared:  0.1434,	Adjusted R-squared:  0.1431
F-statistic: 453.6 on 4 and 10834 DF,  p-value: < 2.2e-16
```

According to the anova function, this model is among the best(nr. 6)

```
Model 1: Value ~ ShortPassing
Model 2: Value ~ ShortPassing + Curve
Model 3: Value ~ ShortPassing + Curve + Dribbling
Model 4: Value ~ ShortPassing + Curve + Dribbling + Volleys
Model 5: Value ~ ShortPassing + Curve + Dribbling + Volleys + Crossing
Model 6: Value ~ (Age + Crossing + Finishing + HeadingAccuracy + ShortPassing +
    Volleys + Dribbling + Curve + HeightMetric + WeightMetric) -
    HeightMetric - WeightMetric - Age - Dribbling - Crossing -
    Finishing
Model 7: Value ~ Age + Crossing + Finishing + HeadingAccuracy + ShortPassing +
    Volleys + Dribbling + Curve + HeightMetric + WeightMetric
  Res.Df        RSS Df   Sum of Sq       F    Pr(>F)
1  10837 9.5076e+15
2  10836 9.4549e+15  1  5.2737e+13 63.6286 1.654e-15 ***
3  10835 9.4342e+15  1  2.0628e+13 24.8884 6.169e-07 ***
4  10834 9.3967e+15  1  3.7582e+13 45.3429 1.738e-11 ***
5  10833 9.3929e+15  1  3.7710e+12  4.5498   0.03295 *
6  10834 9.4330e+15 -1 -4.0148e+13 48.4390 3.604e-12 ***
7  10828 8.9746e+15  6  4.5845e+14 92.1879 < 2.2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
>
```

So we chose this one and added it to our calculations.