Visualization of Football Data

Rasmus Bo Adeltoft
Sebastian Seneca Haulund Hansen
Steffen Berg Klenow
Christian Bjørn Moeslund
Andreas Staurby Olesen
Henrik Sejer Pedersen

Supervisor: Marco Chiarandini

25th May 2016

1 Abstract

2 Preface

Contents

3 Introduction

Data is collected at a rapidly increasing rate in all fields and it becomes necessary to present data in different ways in order for humans to make sense of it. One way to do this is through data visualization. Visualization can help human's understanding of large data sets, as the data can be summarized very effectively, and patterns can quickly be recognized by humans. When making visualizations it is important to understand how the human cognitive system works, such that visualizations can be designed to make it easier for humans to understand the data. In order to do this, we will apply principles from the field of visualization to present football data. We will use tools such as R to process data and plot static visualizations, and use D3 to make interactive and dynamic visualizations.

Specifically, we will do this both by making visualizations that can help explore the questions that we present below, and by doing exploratory analysis such that new patterns can be discovered. The specific questions that we will be investigating are:

- How does a team evolve throughout a season in terms of goals, points, etc.?
- How does a team's playing style (for example passes, possession and tackles) change throughout a match?
- How does a winning team differ from a losing team?

During the visualization process we will consider different visualization techniques and choose a suitable one based on principles and analysis tools given by Tamara Munzner in "Visualization Analysis and Design" to make sure that the data is presented in an accurate and easily understandable manner. This includes considerations regarding the human cognitive system.

4 Theory

4.1 Design Process

This section describes the typical work flow of a data scientist. We will focus on the following four faces: Preparation, Analysis, Reflection and Dissemination.

The process of getting the data, understanding the data and produce results is an iterative process. The process is seen on Figure 1.

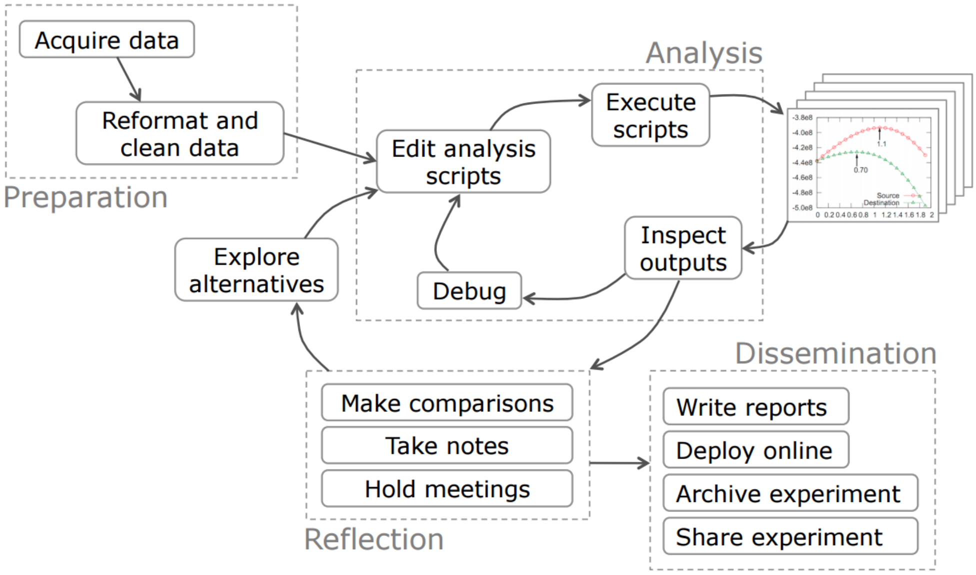

Figure 1: The model showing the iterative process[Guo12, Chapter 2]

- The first face is the preparation face. Here the you have to acquire the data, that could be from hard disks, servers, through an API ect. Where to store and how to organize the data files should be considered, so it is easy to replace the right files if the data gets updated. Then the data should be cleaned, meaning removing tuples with missing values, changing the formatting, sorting the data ect.

- The second face is the analysis face. Here the data is analysed to get more information about it. This is an iterative process, where the you create and run scripts, look at the output, maybe find some mistakes, debug these and run it again.

- The third face is the reflection face. Here the output results is discussed, for example by making comparisons between outputs, and exploring alternatives.

- The fourth and last face is the dissemination face. Here the the results are reported and maybe published in a report. [Guo12, Chapter 2]

4.2 Design

4.3 Data Types and Data Sets

4.4 Idioms

4.5 Analysis and Complexity

4.6 Facets and View Manipulation

When creating visualisations it is not always enough with one idiom to present the data in a understandable way. One way cram more information into the idiom, but still keeping the number of variables low is to either manipulate the view of the idiom or to facet into multiple idioms.

4.6.1 View Manipulation

By creating multiple views in an idiom, the idiom can contain more information, without clutter. The different views can include changes like switching between different idioms, changing the viewpoint, changing the order of the data, changing the number of items showed and so on. For example you can change the way the data is ordered by sorting the data by different variables. This is very powerful because of spatial position being the highest ranked visual channel. Many view manipulations is based on animation. Animating has a trade off, the cognitive load can be very high if too many elements change. This means that we get low cognitive load if either some elements are static and others moving, or some groups of elements are static and others moving. If few elements change by a graduate transition, the viewer can keep the context between the two views. [MM15, Chapter 11]

Selecting one or more elements is common in many interactive visualisations. The result of selecting some elements is then some change in the view. It has to be considered which elements the user can select, and how many the user can select. Choosing how to select items is also subject which has to be considered, clicking to select, hovering over some element with the cursor or something else. Changing the view by highlighting some element and creating pop out could be done by changing the channel, for example the color, the size, the outline or the shape of the element. This change should of course be so dramatic that the element clearly stands out from the rest. Look at Figure 2 for an example of highlighting. [MM15, Chapter 11]

Another option for an interactive idiom is the ability to navigate the view by changing the viewpoint. Here we think as if we have a camera pointed at the view. We can then change the view by zooming, panning or

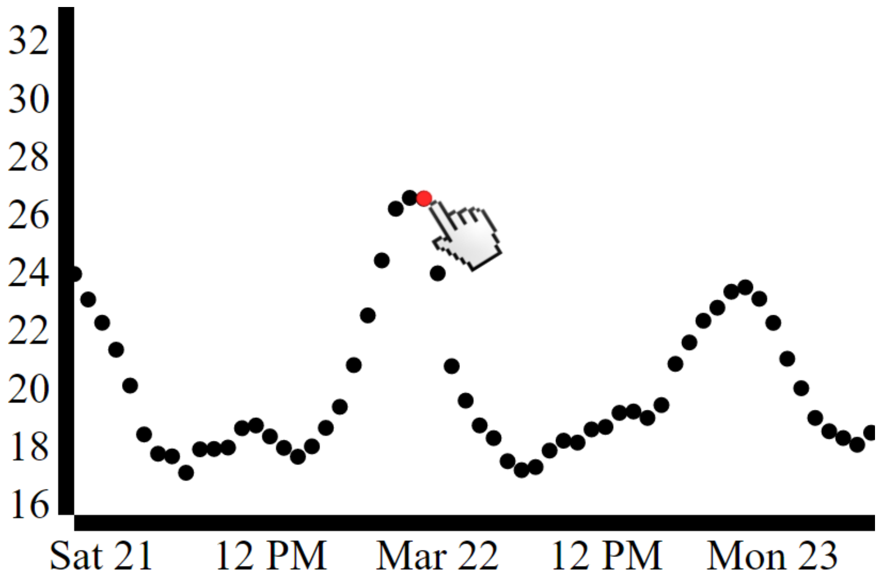

Figure 2: Highlighting the selected element

rotating the the camera around its own axis. There are two kinds of zooming, geometric zooming and semantic zooming. Geometric zooming is straight forward making some elements come closer to the camera. With semantic zooming the not only the size of the elements change, but the semantics too. Semantic zooming changes what is shown, and maybe the representation of it. For example zooming semantically could view more detail about some element showing new information about it. Navigation could also be changed through reduction of attributes, by slicing, cutting or projecting. These are all dimension reduction techniques. To slice, a specific value at a dimension is chosen, and only elements matching this value is shown. A cut is made by placing a plane in front of the camera, all elements in front of the plane is not shown, in this way it is possible to explorer elements behind other elements, or looking inside 3D objects. Projection is done by eliminating some dimension but still showing all the data. This is similar to what the human do when looking at a 3D object.

4.6.2 Facet

Faceting is splitting the view up into multiple views or into multiple layers. One of the main reasons to facet is to compare views. This is much easier than comparing two views in a changing view, because we do not have to remember the prior view, but continuously can compare them. Another reason to facet is to gain more information about the data through a multiform design,

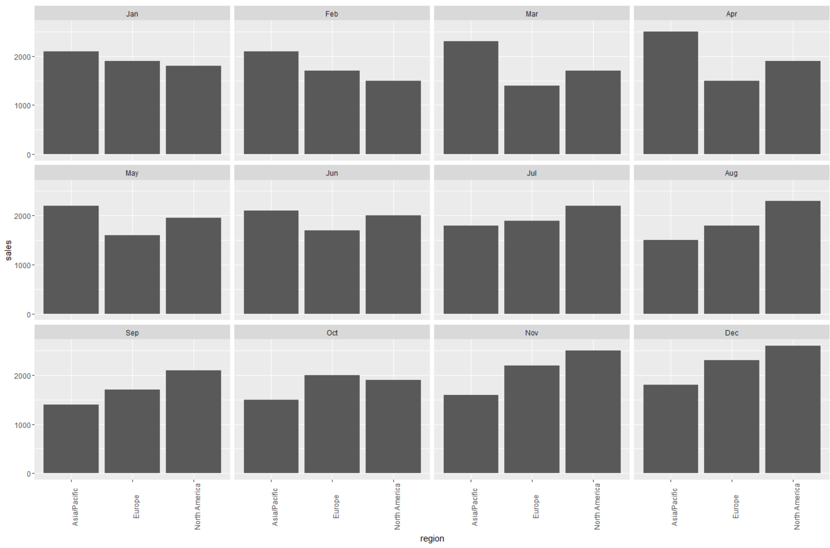

Figure 3: An example of small multiples

where data is shown using different encodings. By having multiple views more attributes can also be shown. Of course when you have multiple views shown beside each other, each view has less space, which is one of the trade off's and why having multiple layers on top of each other, and switching between them sometimes is better. If having juxtaposed views it might be interesting to link the views. This could be done sharing the data, sharing the visual encoding, synchronizing the navigation or highlighting.

Each view could show different subsets of the data, or having different viewpoints like the classic overview in one view combined with detail in another view. Having multiple views sharing encoding but showing different parts of the data is called small multiples and is often structured in a matrix. This could be an alternative to animations, where we lay out all the frames. The cognitive load is smaller with small multiples, and it is easy to go one frame back or forth. An example of small multiples is seen at Figure 3. Instead of juxtaposing the views, it is a option to stack them into a single frame. The views should have the same horizontal and vertical extend and blend blend together as one frame, by being transparent where there are no marks. The problem with stacking is distinguishing between the layers. This is easy with only a few layers, especially if the layers use different visual channels. But distinguishing between more than three layers, can be a real challenge. [MM15, Chapter 12]

4.7 Exploratory Data Analysis

Exploratory data analysis, EDA, is a philosophy about how one can approach the analysis of a data set. It is not a fixed collection of tools that one can use to analyze a data set, but it is a general approach which promotes looking at data in different ways without having any assumptions. It is usually used when one receives a new data set, and wishes to learn something about the underlying structure of the data set, or wishes to discover outliers or trends in the data. In general, EDA is about looking at the data that is presented to one, in many different ways. Both visual and non-visual methods are used in EDA. Examples of non-visual methods are simply to calculate the mean, median, variance, quartiles etc. of the data, while visual methods could be a boxplot, scatterplot, which quickly allows one to discover outliers and trends, or even a simple bar chart. The boxplot is a quick way to present some of the calculated properties mentioned above, such as the median and some of the quartiles. The boxplot was actually developed by John Tukey, who is widely regarded as one of the big promoters of EDA, and it was presented in his book "Exploratory Data Analysis", as a method one could use while doing EDA. Another approach one can use within EDA is principal component analysis, PCA, which is described in another section.

When doing EDA, one will often be able to use the acquired knowledge to formulate new hypotheses about the data, which can then be used to make more specific visualizations or calculate more specific properties of the data set, but it is not a guaranteed outcome. EDA will not always produce new knowledge, which is not a flaw in the approach, but rather a natural consequence of the approach.

4.7.1 Principle Components Analysis

Principal Components Analysis, or in short PCA, is a way to take high dimensional data and reduce it without loosing much information. It is difficult to find clusters, similarities and differences in high dimensional data, but by doing PCA it becomes much easier because of the dimension reduction. [Smi02]

If you have you multidimensional data plotted in a coordinate system, PCA finds the vectors which describes the most variance in the data. These are called the eigenvectors. The eigenvector describing the most variance is the first principle component, the eigenvector describing the second most variance is the second principle component and so on. The principle components are perpendicular. To find out which variables have the most influence on a PC, we can look at the coefficients of the PC, meaning the coefficients of the eigenvector, the higher the coefficient, the higher the influence. When having found the principle components, it is time to find out which to keep. This can bee done by looking at a scree plot. This is a bar chart showing the variance for each of the PC's. Typical it is interesting to look at the first two PC's. We then look at the data expressed in terms of the principle components we have chosen. If we plot the derived data as a scatter plot, with for example the first PC on the x-axis and the second PC on the y-axis, we can easily point out the clusters of the data if there are any. This way we can look at the data in fewer dimensions but without much loss of information. In section 5.1 we do principle components analysis on some of our own data.

4.7.2 Linear Regression

4.8 Tools and Technologies

4.8.1 R

4.8.2 D3.js

5 Results

5.1 Principle Component Analysis

In this section we do Principle Component Analysis on aggregated on one season of football data. The data includes observations like the number of goals, the number of shots, the number of passings and so on. The data is collected for each player, but we have summed it to each team. We would like to find any clusters by using PCA in R.

The resulting scree plot from doing the PCA is seen in Figure 4. The scree plot tells us which eigenvectors have the highest eigenvalue, meaning which principle components describe the most of the data. In our case we find that the first two components describe a lot of the variance in the data, to some extend also component 3.

We know look at the loadings each of the components that we choose to keep. The loadings describe the importance of the different variables in relation to the chosen component. In Figure 5 we see the loadings for PC1. Being that PC1 is the component describing most of the variance in the data, the most important variables of PC1 is also the most important variables for the entire data set. The five most important variables in regards to PC1 is the "Sum.of.Chances", "Sum.of.2nd.Assist.To.Shot", "Sum.of.Fouls.Received.In.Second.Third.Part", "Sum.of.Assist.To.Goal" and "Sum.of.Shots.On.Target".

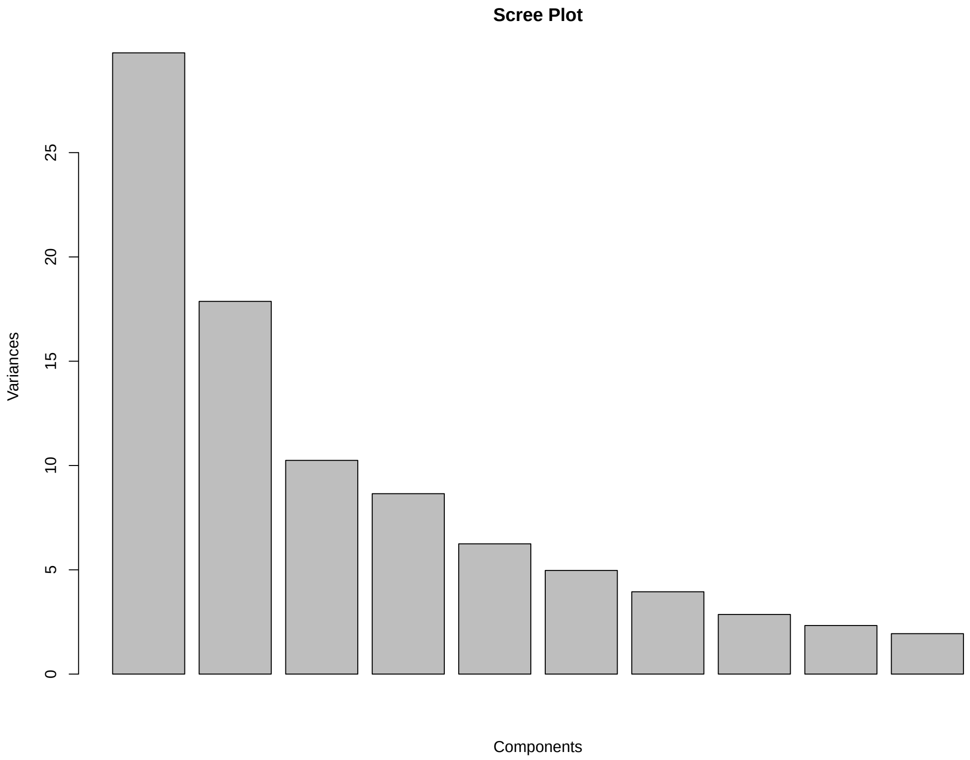

Figure 4: The resulting scree plot

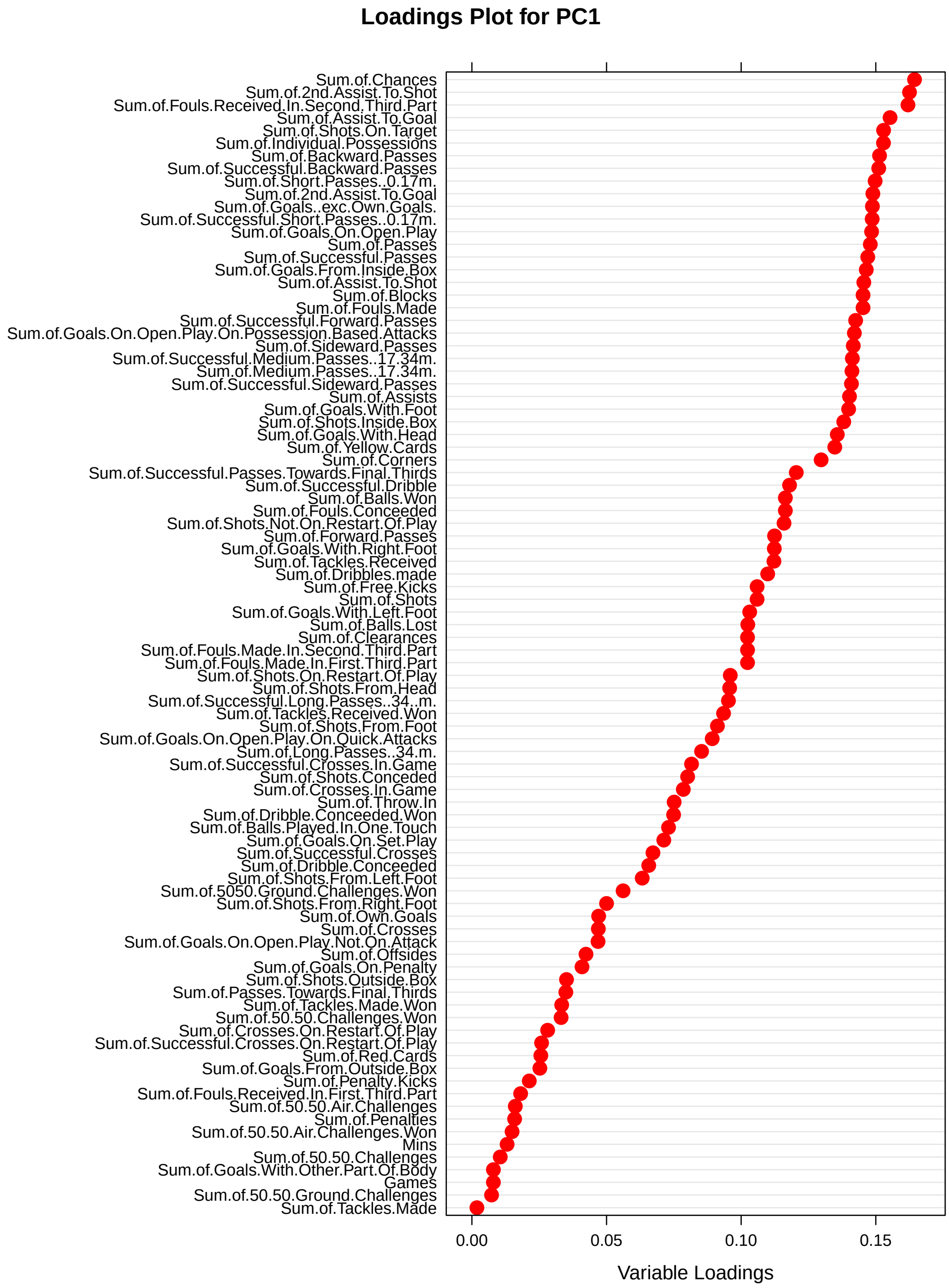

Figure 5: Loadings for PC1

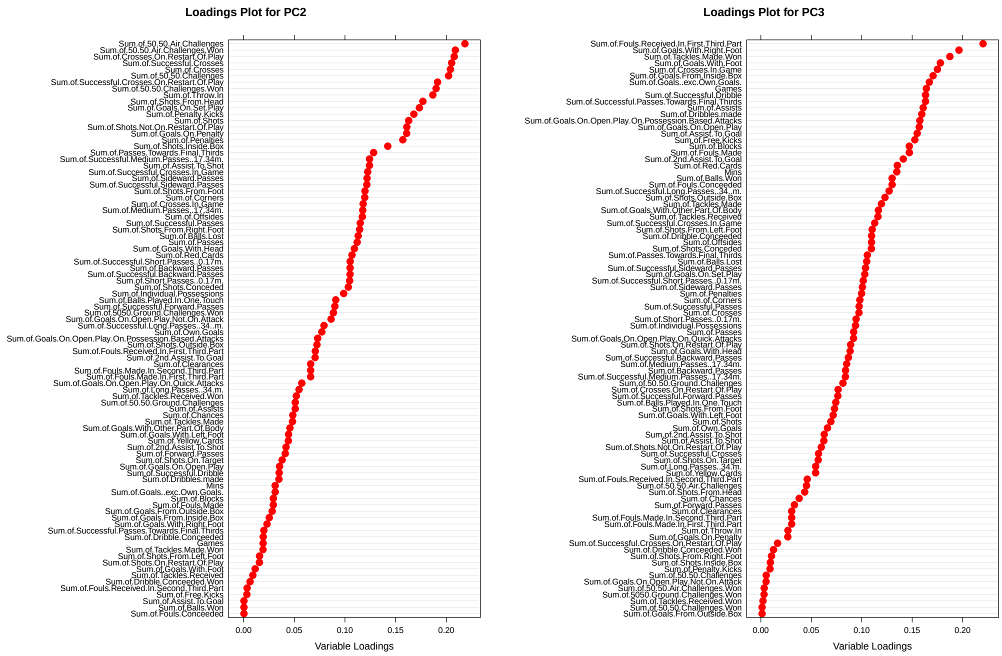

Figure 6: Loadings for PC2 and PC3

We can do the same with PC2 and PC3. These are seen at Figure 6. From the loadings in regard to PC2 we conclude that the five most important variables are the "Sum.of.50.50.Air.Challenges", "Sum.of.50.50.Air.Challenges.Won", "Sum.of.Crosses.On.Restart.Of.Play", "Sum.of.Successful.Crosses" and "Sum.of.Crosses". For PC3 the five most important variables are the "Sum.of.Fouls.Received.In.First.Third.Part", "Sum.of.Goals.With.Right.Foot", "Sum.of.Tackles.Made.Won", "Sum.of.Goals.With.Foot" and "Sum.of.Crosses.In.Game".

We can now plot the data points in regards to the principle components to find clusters. The data is sorted in regards to current position of the teams in the table. This means that we easily can find out if the top/bottom teams cluster. In Figure 7 the data is plotted in regard to PC1 and PC2. We do not have any clear clusterings, some teams are clustered in the middle but there is no correlation between the standings and the teams clustering.

In Figure 8 the data is plotted in regard to PC2 and PC3. Once again we have a little cluster, but there is no relation to the standings.

The PCA has not giving any clusters, but it has giving us some of the most important variables of the dataset.

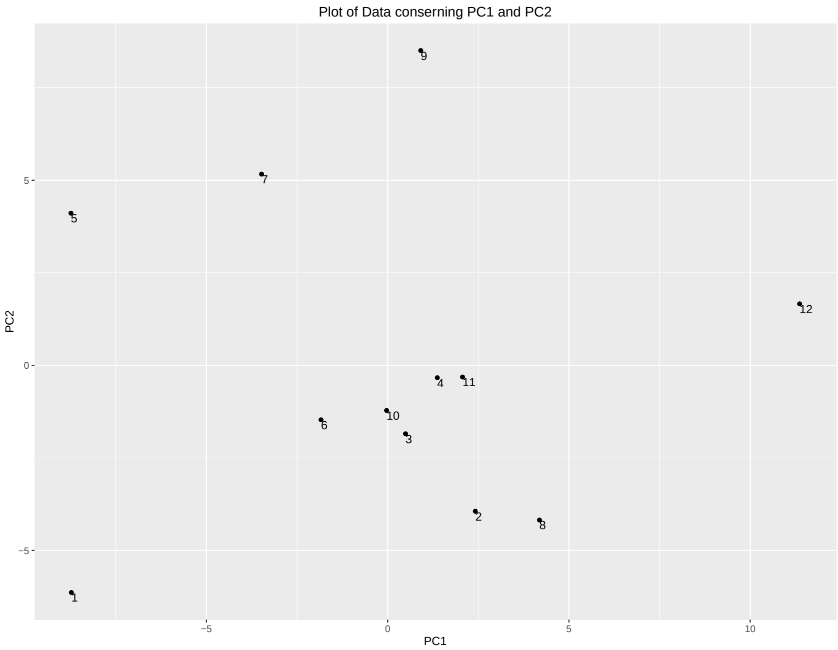

Figure 7: The data plotted in regard to PC1 and PC2

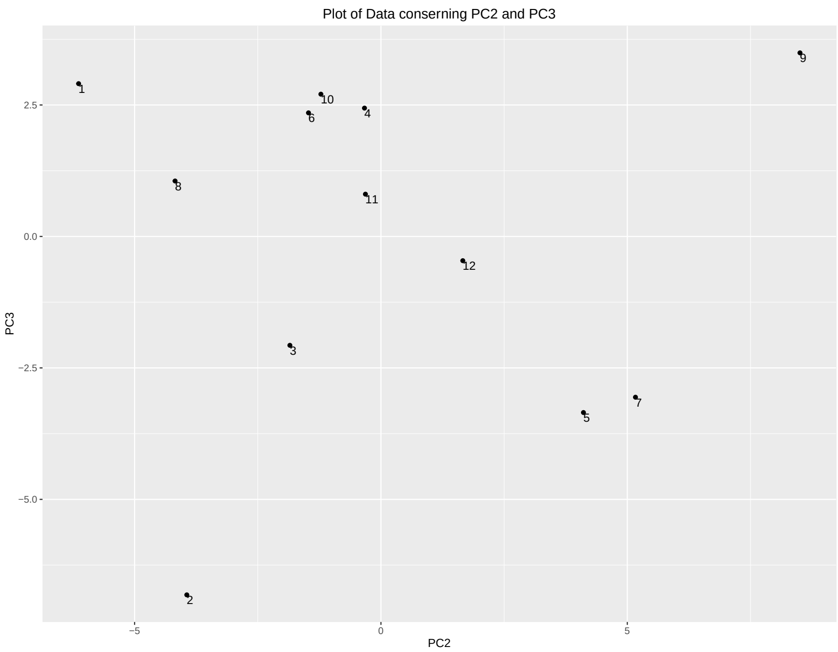

Figure 8: The data plotted in regard to PC2 and PC3

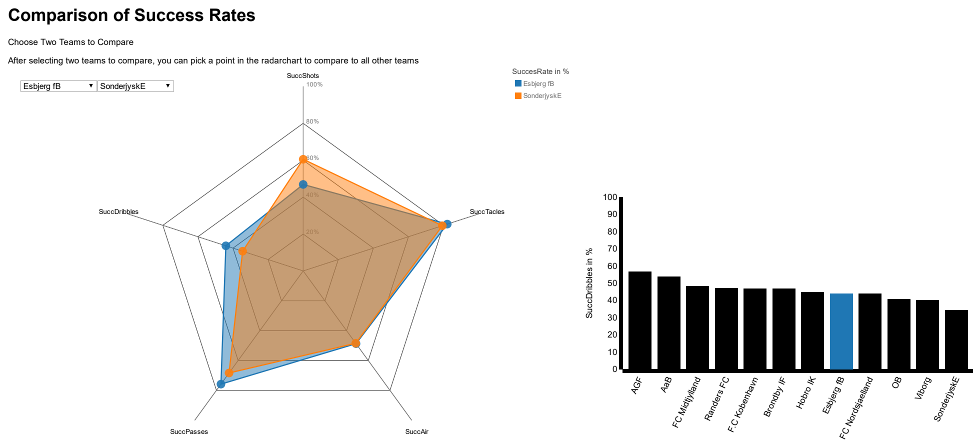

Figure 9: Idiom showing the comparison of the success rates using a radar chart for comparing two teams and bar chart for comparing with all other teams in the league

5.2 Success Rates

5.2.1 What-why-how

The visualisation shows the comparison of two teams' success rate on several points of measure in a radar chart combined with a bar chart, this is shown in Figure 9. Concerning magnitude channels we position the marks on a commons scale and also use the area of the radar chart for comparison. For categorising we use color hue.

The action of the idiom is to compare two teams to find differences/similarities, maybe in relation to the placement in the league table, and to analyse where the teams differ. The target is to find extremes and outliers.

This is done by faceting into two views. In the left view we have a radar chart and in the right view we have a bar chart. The user can select two teams to compare in the radar chart. The teams' success rates in relation to shots, tackles, air challenges, passes and dribbles, is shown on the radar chart. When pressing one of the team's value in one of the measurement points the bar chart for all the teams in the league shown comparing this measurement. The views are linked together by synchronizing the highlighting the chosen team in the bar chart. The highlight in the bar chart creates pop out by having this bar coloured and all others black. The raw data is reduced filtering and by aggregating. Only some variables are chosen and calculating the values for each team from the individual player's values. The radar chart is manipulated first off by selecting two teams to compare. If the user hovers over the values, which highlights all the team's values together with the area of the chart. Also if clicking a value the bar chart is updated to show compare this measurement for all teams.

5.2.2 Code

The R code is quite simple. In broad strokes it does the following: Read the data file, convert columns to be numeric, sum the data by team (before player wise), add and calculate the success columns, remove the unnecessary columns, restructure the data to a format that fits the radar chart and write three files, one containing the names of the teams,one containing the data in one format and one containing it in another format. The data is exported in two files to suit both the radar chart and the bar chart.

In JavaScript we create the selectors, where the user chooses the teams. If two teams are chosen the radar chart is drawn. The radar chart is based [d3n13] and modified to our needs. One of the modifications is that we draw a bar chart if the user presses a value being a circle. The function which draws the chart takes the data variable to visualise as an argument together with the selected team. The data is then loaded and drawn making sure that the data is sorted for better comparison, and that the right team is highlighted.

5.3 Nationality of players

5.3.1 What-why-how

This visualisation shows the nationality of the players on the different teams in the league. It also shows the distribution of the foreign players among the other countries. This is done by a sankey chart, where the teams are sources, the targets are either Denmark and Foreign. On the second level the source is Foreign and the target is the other countries. The idiom is shown in Figure 10.

The actions of this idiom is to summarise the players nationality, to explore and compare the different teams to find similarities/differences between the teams in comparison to the standing in the table.

This is done by an interactive sankey chart. The view is manipulated by highlighting when hovering over a link, where we also get the percentage of players in this connection. The raw data is reduced by only focusing on the nationality of the players. The players are categorised into the different bins by using hue and region.

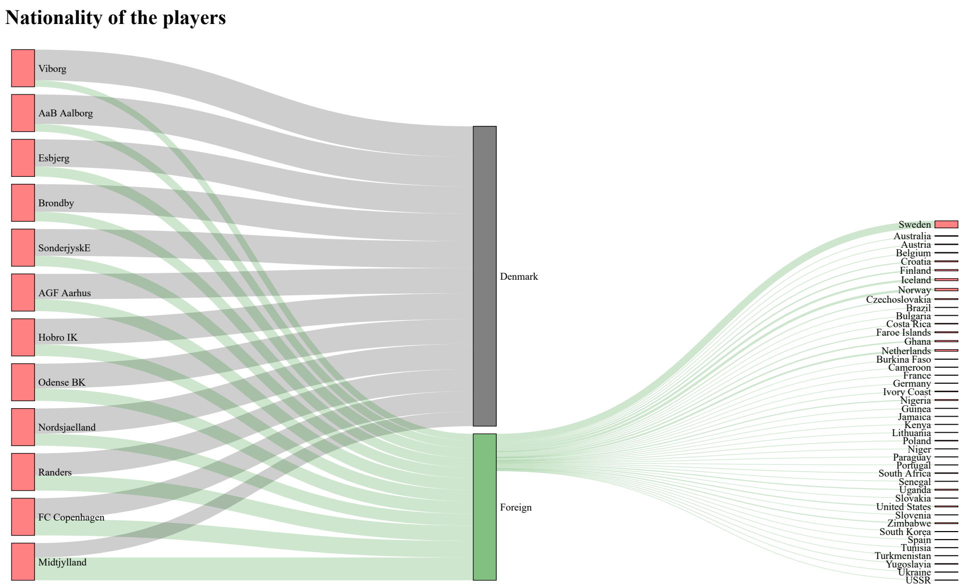

Figure 10: Sankey-Chart showing the distribution of danish vs. foreign players on the different teams

5.3.2 Code

We load in the data file, whcih now is a json file. We then find the data we want to look in the json file, and convert it into a dataframe. We then sum the nationalities of each club. We then create a copy of the frame. In the original frame we all the sources to be foreign where the target is not Denmark. In the second frame we set the target to be Foreign where the target is not Denmark, and then remove all rows where the target is Denmark. In both frames we sum the values for each source-target pair. We then combine the two frames and exchange the value being the number of players to now being the percentage of players. Finally we write the data to a .csv file.

The sankey chart itself is based on [Bre16] and modified to our use. We modify the placement of the nodes to have multiple levels, and clearer color scheme with clear differences between the links.

6 Discussion

7 Conclusion

8 Usability

9 Bibliography

References

[Bre16] Nadia Bremer. D3.js - radar chart or spider chart - adjusted from radar-chart-d3, 2016.

[d3n13] d3noob. Sankey diagram with horizontal and vertical node movement, 2013.

[Guo12] Philip Jia Guo. *Data Science Workflow: Overview and Challenges.* PhD thesis, Stanford University, 2012.

[MM15] Tamara Munzner and Eamonn Maguire. *Visualization analysis and design.* CRC Press, Boca Raton, FL, 2015.

[Smi02] Lindsay Smith. A tutorial on principal components analysis, 2002.

10 Appendix

11 Process Evaluation