The Global Game

An Analysis of Nationality in Professional European Football

Andrew Tremante

Introduction

In August 2021, the football world buzzed with discussions about the record-breaking transfer of English footballer Jack Grealish to Manchester City for a staggering £100 million (*ESPN,* "Manchester City Sign Jack Grealish"). This signing reignited debates over player valuations in the Premier League — the top football league in England — and raised questions about the influence of nationality on market prices. The concept of an "English Tax" resurfaced, highlighting perceived disparities in the valuation of English players compared to their international counterparts in the Premier League (*Over the Line Sports,* "A Look at the English Tax Issue").

The "English Tax" phenomenon provides a unique lens for soccer fans and analysts to understand how economic and strategic decisions influence their favorite clubs. This concept underscores how nationality impacts player valuations, with English clubs often paying a premium for domestic talent. From club officials to fans, understanding these market dynamics is significant. They shape transfer policies, team compositions, and even fan engagement, particularly in a league as popular as the Premier League. This deeper understanding of football economics enables clubs to make more informed transfer decisions and offers new insights for fans into the strategies of the clubs they support.

This project explores these issues by analyzing data from sources such as FBref and Kaggle to investigate the impact of nationality on players' market values and team affiliations across Europe's most prominent football leagues — the English Premier League, the German Bundesliga, the Spanish La Liga, the French Ligue 1, and the Italian Serie A. Through this investigation, the project seeks to provide a more nuanced understanding of the dynamics that shape professional football.

Data

This project leverages multiple data sources to analyze the impact of nationality on football player market values. It integrates datasets containing performance metrics from FBref, market values from Kaggle, and geographic coordinates from GitHub.

Performance data for the 2020-21 season was obtained from FBref, a prominent football statistics website. The data was scraped using the **`rvest`** package, which extracted the relevant HTML tables containing player statistics from the desired leagues. Subsequent cleaning adjusted column names and corrected abbreviations for player nationalities and competition names to ensure consistency. Only the most relevant columns were retained for the final dataset.

Market value data for the top 500 players in 2021 was sourced from Kaggle's *Most Expensive Footballers 2021* dataset. Initially downloaded as a **`csv`** file, the data was imported and column names were adjusted for compatibility. Next, the refined data was fully joined with the performance data scraped from FBref. The joining process matched players across the datasets based on their names, creating a new dataset containing both performance metrics and market values. This dataset was further processed by renaming and selecting relevant columns, removing **`NA`** values, adding a column for league countries, and converting appropriate columns to numeric values from their original character types.

Geographic data containing the longitude and latitude of countries was sourced from the **`countries_and_coordinates.csv`** dataset on GitHub. This data was essential for mapping players and leagues to their respective geographic locations in the Shiny app. The geographic data was transformed to align the Alpha-3 country codes with the country codes used in the player data, ensuring accurate joins. The player and geographic data were merged to include longitude and latitude for both player nationalities and the countries of the leagues in which they play. Lastly, the final dataset was modified by removing duplicates and once again ensuring all numerical data was correctly formatted.

The entire data wrangling process was documented in a Quarto markdown file, **`wrangling.qmd`**, providing a comprehensive record of the transformations and manipulations performed on the data. This documentation is crucial for reproducibility and offers a clear guide to the steps taken to prepare the dataset for analysis. The final dataset was saved as an RDS file, **`final_data.Rds`**, to maintain data integrity and facilitate easy access in the Shiny application. This RDS file is then loaded into the **`app.R`** file that powers the interactive elements of the Shiny app, allowing users to dynamically explore the data.

Methods

The Shiny application is structured into three pages, each designed to explore a unique impact of nationality on player market values and strategic decisions in football clubs.

Page 1: Nationality

The first page of the Shiny application assists the user in their analysis of football players based on their nationality. This page is centered around enabling users to filter players using sliders for market value (in millions of £), age, and minutes played. The interactivity of these sliders is powered by the **shiny** package, which allows for real-time interaction.

Code Implementation for Slider Inputs

The sliders are implemented in the UI (User Interface) part of the Shiny app using **sliderInput()** functions. Here is a snippet showing how these sliders are set up:

```
# Sliders for filtering players based on market value, age, and minutes played
# The sliderInput function creates a slider widget for each filter criteria
# The min, max, and default values are set based on the corresponding columns
# in the 'final_data' dataset

sliderInput("valueSlider", "Minimum Market Value:",
            min = 0,
            max = max(final_data$Value),
            value = 0),
sliderInput("ageSlider", "Minimum Age:",
            min = min(final_data$Age),
            max = max(final_data$Age),
            value = min(final_data$Age)),
sliderInput("mpSlider", "Minimum Minutes Played:",
            min = 0,
            max = max(final_data$Mins),
            value = 0)
```

Each slider filters the dataset based on the respective attribute, allowing users to dynamically adjust the visualization according to their specific interests.

Server Logic for Data Filtering

On the server side, the **dplyr** package is used to apply these filters to the dataset. The filtered data is then used to populate a geographic map and a data table. Here's a brief overview of how the filtering logic is implemented:

```
# Filter players based on user input from sliders
# The reactive expression 'filteredPlayers' updates automatically
# It filters the 'final_data' dataset based on
# the selected minimum market value, age, and minutes played

filteredPlayers <- reactive({
  final_data %>%
    filter(
      Value >= input$valueSlider,
      Age >= input$ageSlider,
      Mins >= input$mpSlider
    )
})
```

This reactive expression ensures that any changes in the slider inputs are automatically and efficiently propagated through the dataset.

Geographic Visualization with Leaflet

The **groupedPlayers** object is then created by first applying a function to **filteredPlayers** and then grouping the results by nationality, using the latitude and longitude coordinates. The data is then summarized into a single entry per group. Each entry combines the players' names, market values, ages, and minutes played into a formatted string for each nationality. The results of the filtering are visually represented on a geographic map using the **leaflet** package. Players' nationalities are marked with interactive markers, implemented using **addMarkers()** function, which provides popups with summarized data (*R-Bloggers,* "Using Leaflet in R").

```
# Creates a leaflet map with markers for each country
# The leaflet function initializes the map
# The addTiles function adds the base map tiles
# The addMarkers function adds markers for each country,
# with the nationality as the popup content and layer ID

leaflet(groupedPlayers) %>%
  addTiles() %>%
  addMarkers(
    lng = ~Nation_Long,
    lat = ~Nation_Lat,
    popup = ~paste0("<b>Country:</b> ", Nation),
    layerId = ~Nation
  )
```

Dynamic Data Table Updates

A key feature in the application is the use of **observeEvent** in the server logic. This Shiny function monitors user interactions with the map markers. When a marker is clicked, **observeEvent** triggers a reactive update to a detailed data table displayed below the map (*Shiny Development Center,* "observeEvent"). This table, rendered using the **DT** package, updates to show comprehensive information corresponding to the selected nationality or filter settings. This reactive behavior is crucial as it ensures that the data table only displays relevant player information matching the user's interaction with the map, thereby making the analysis more focused and user-friendly.

```
# Display player information when a marker is clicked
# The observeEvent function triggers an action when a marker is clicked
observeEvent(input$playerMap_marker_click, {
  # Get the ID of the clicked country from the marker click event
  clickedCountry <- input$playerMap_marker_click$id
  # Filter the player data based on the clicked country
  playerData <- filteredPlayers() %>%
    filter(Nation == clickedCountry) %>%
    select(Player, Value, Age, Mins)

  # Render the filtered player data in a table
  # The renderDataTable function renders an interactive table
  output$playerTable <- renderDataTable({
    datatable(playerData, options = list(pageLength = 5))
  })
})
```

This approach ensures that the displayed data in the table is always relevant to the user's selection on the map. The reactive behavior between the map and the data table through **observeEvent()** enriches the application by providing a seamless user experience, where map interactions directly influence the data displayed in real time.

Page 2: League

The League tab of the Shiny application showcases an advanced geographic visualization that illustrates the global recruitment strategies within each league. This tab allows users to select a football league from a dropdown menu, which activates a reactive function within the server logic to filter the dataset for players associated with the chosen league. This interactive capability is enabled through Shiny's input and reactive expressions, which monitor changes in user input and dynamically update the displayed data.

Code Implementation for League Selection

The league selection is handled via a **`selectInput()`** function in the UI, allowing users to choose from a list of leagues:

```
# The reactive expression 'filteredLeaguePlayers' updates automatically
# whenever the selected league changes
# It filters the 'final_data' dataset based on the selected league
# from the dropdown
filteredLeaguePlayers <- reactive({
  final_data %>% filter(League == input$leagueSelect)
})
```

Upon selection, the server logic reacts by filtering the dataset for the specified league using the **`dplyr`** filter function.

Geographic Visualization with Leaflet

The visualization on this tab is constructed using the **`leaflet`** package. A central marker is placed at the geographic center of the selected league, calculated by averaging the coordinates of all associated players. This central marker acts as an anchor point from which arrows (polylines) extend to each player's country of origin, revealing the distribution of foreign nationalities of players in the league.

Implementation of Polylines

Polylines were a new and advanced addition to this project, extending the visual representation capabilities within the **`renderLeaflet()`** function. The concept of polylines was incorporated to visually link players' nationalities with their league locations on the map (*Leaflet for R,* "Shapes").

Implementing polylines involved several steps, including reviewing documentation and examples to comprehend the mechanics of Leaflet's mapping functions, specifically **`addMarkers`** and **`addPolylines`** (*WRLD3D,* "Adding a Leaflet Polyline").

The code snippet provided demonstrates the use of a for loop to process each player's data iteratively. For each iteration, the loop performs the following tasks:

- **Unique ID Generation:** A unique layer ID is created for each nationality using **`paste0("nation", i)`**, which helps in associating markers with their corresponding polylines for interactive functionality.

- **Marker Placement:** The `addMarkers` function places a marker at the geographical coordinates (longitude and latitude) of the player's nationality with a popup for each marker displaying the country name.
- **Polyline Creation:** The `addPolylines` function draws a line from the league location to the player's nationality coordinates. These lines are color-coded and set with partial transparency to maintain clarity when overlapping occurs. The same layer ID used for markers is applied to polylines, ensuring that each line is connected to its corresponding marker.

```
# Add markers and lines for each player's nationality
# The for loop iterates over each player's nationality
# It generates a unique ID for each nationality layer
# The addMarkers function adds a marker for each nationality
# The addPolylines function adds a line connecting the league location to each
# player's nationality, with the same layer ID as the marker

for (i in 1:nrow(playerNationalities)) {
  natId <- paste0("nation", i)  # Generate a unique ID for the layer
  map <- map %>%
      addMarkers(
        lng = playerNationalities$Nation_Long[i],
        lat = playerNationalities$Nation_Lat[i],
        popup = playerNationalities$Nation[i],
        layerId = natId
      ) %>%
      addPolylines(
        lng = c(leagueLocation$Country_Long, playerNationalities$Nation_Long[i]),
        lat = c(leagueLocation$Country_Lat, playerNationalities$Nation_Lat[i]),
        color = "blue",
        opacity = 0.5,
        layerId = natId  # Use the same ID for the line
      )
}
```

Through trial and error and iterative testing, the integration of polylines was refined to ensure that the visual links were accurately drawn and that performance was optimized for user interactions. Challenges such as managing overlapping lines and ensuring that the map remained readable were addressed by adjusting line opacity and colors.

This coding approach not only facilitated the drawing of markers and lines but also ensured that each line was uniquely identified. The successful implementation of polylines significantly enhances the analytic capabilities of the project, allowing users to explore the complex geographic relationships between players' nationalities and their leagues (*Stack Overflow,* "How

to Add Interactive Maps to R Using Leaflet").

Interactive Elements and Data Display

The map allows users to interact with both the polylines and nationality markers. Clicking these elements triggers **`observeEvent()`** functions that display detailed information about players from the selected nation who are part of the league, rendered in a dynamically updated data table:

```
# Display player information when a marker or line is clicked on the league map
# The observeEvent function triggers an action when a marker or line is clicked
observeEvent(c(input$leagueMap_marker_click, input$leagueMap_shape_click), {
  # Get the ID of the clicked marker or line from the click event
  # The %||% operator returns the first non-null value
  clickedId <- input$leagueMap_marker_click$id
  %||% input$leagueMap_shape_click$id

  # Ensure a valid click ID is available before proceeding
  req(!is.null(clickedId))

  # Get the filtered players from the selected league
  playersFromLeague <- filteredLeaguePlayers()

  # Check if the clicked ID is not the league location marker
  if (clickedId != "leagueLocation") {
    # Extract the nationality index from the clicked ID
    nationIndex <- as.numeric(gsub("nation", "", clickedId))
    # Get the selected nationality based on the index
    selectedNation <- playerNationalitiesReactive()$Nation[nationIndex]
    # Filter the players from the selected league by the selected nationality
    playersFromNation <- playersFromLeague %>%
      filter(Nation == selectedNation) %>%
      select(Player, Nation, Age, Value, `G+A`)

    # Render the filtered player data in a DataTable
    # The renderDataTable function renders an interactive table
    output$playerTableLeague <- renderDataTable({
      datatable(playersFromNation, options = list(pageLength = 5))
    })
  }
})
```

This integration of interactive map elements with reactive data tables effectively illustrates the dynamic nature of international player transfers, emphasizing the global reach of football leagues.

Page 3: Clustering - Enhanced Unsupervised Learning

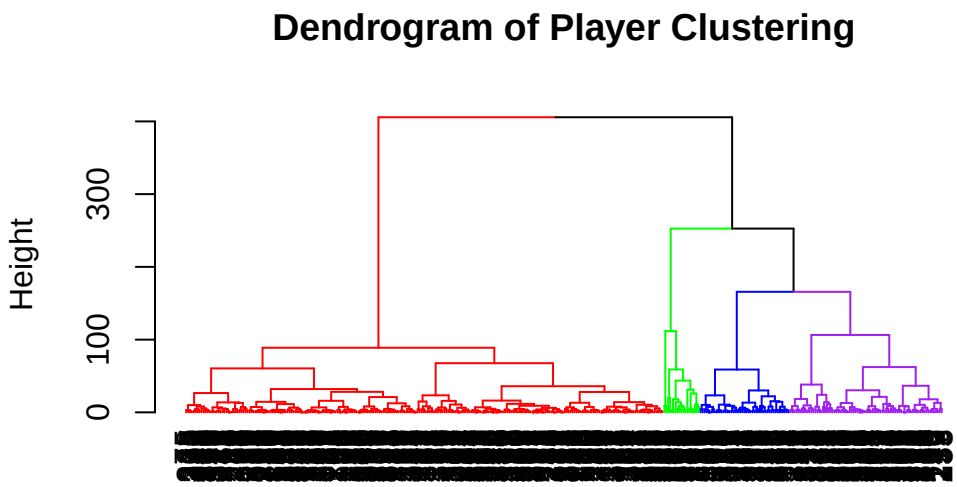
The Clutsering page of the Shiny application showcases an advanced approach to analyzing football player data using hierarchical clustering. This method *beyond* the scope of the course is adept at identifying nuanced patterns within complex datasets, which may not be detectable with simpler clustering techniques like K-means.

Hierarchical Clustering Methodology

Hierarchical clustering is an analytic method that provides a means of partitioning a dataset into clusters at different levels of similarity. This technique was new to this project, serving as a powerful tool for understanding complex relationships within the data.

The process begins by considering each individual data point as a separate cluster. It then iteratively merges the closest pairs of clusters based on a specific measure of similarity or distance, such as Euclidean distance or Manhattan distance. This merging continues until all points are unified into a single cluster, thereby creating a hierarchy of clusters that can be visualized through a dendrogram (*MathWorks,* "Hierarchical Clustering").

A dendrogram is a tree-like diagram that represents the nested groups of objects and their proximity in a visual form. Each merge is represented by a horizontal line, the height of which indicates the distance or dissimilarity between the merged clusters. This visualization allows for easy identification of the hierarchy and the number of clusters that best represents the data structure. For this project, four clusters, distinguished by different colors, were chosen given their clear distinction among different groups of players. While the dendrogram structure might suggest that five clusters could have been superior, this number led to less discernible results in the context of analyzing player data. Below is the dendrogram used for this project:



In the context of this project, agglomerative hierarchical clustering was implemented using the **hclust** function from the **stats** package in R using the Ward.D2 method. The Ward.D2 method aims to minimize the total within-cluster variance at each step of the clustering process. It uses Euclidean distance to calculate the distances between points. This method is advantageous because it tends to create more compact and equally sized clusters. However, alternative algorithms such as Single Linkage (Nearest-Point) or Complete Linkage (Farthest-Point) could also have been implemented (Johnson, "Understanding Hierarchical Clustering in Data Science").

```
# The reactive expression 'plot_data' updates automatically
plot_data <- reactive({
  # Include Age in the distance matrix calculation
  # The dist function computes the Euclidean distance matrix
  dist_mat <- dist(final_data[, c("G+A", "Value", "Age")])

  # Perform hierarchical clustering using the Ward's method
  # The hclust function performs hierarchical clustering using ward.D2 method
  hclust_res <- hclust(dist_mat, method = "ward.D2")
  # Cut the hierarchical clustering tree into 4 clusters
  # The cutree function assigns cluster labels to each observation (k = 4)
  clusters <- cutree(hclust_res, k = 4)
```

```
  # Add cluster labels and ensure numeric data types for plotting
  # Mutate function adds a new column "Cluster" with the assigned cluster labels
  # The Value, G+A, and Age columns are converted to numeric data type
  final_data %>%
    mutate(Cluster = as.factor(clusters),
           Value = as.numeric(Value),
           `G+A` = as.numeric(`G+A`),
           Age = as.numeric(Age))  # Ensure Age is numeric if not already
})
```

The use of hierarchical clustering in this project allowed for a deeper exploration into how different player attributes are interrelated within the dataset. By choosing an appropriate method like Ward.D2 and employing the **hclust** function, the data was successfully segmented into meaningful clusters, thereby supporting more sophisticated analysis based on the patterns revealed through the clustering process (*Statistics How To,* "Hierarchical Clustering").

Application and Visualization

The Clustering tab categorizes players using three key metrics: goals and assists (G+A), market value, and age. These metrics are selected due to their relevance in evaluating a player's performance and market standing. The results of the hierarchical clustering are visualized interactively using the **plotly** package, which created a dynamic scatter plot:

```
# The renderPlotly function renders an interactive scatter plot
output$clusterPlot <- renderPlotly({
  # Get the prepared data for plotting
  plot_data <- plot_data()
  # Create a scatter plot using the plot_ly function
  # The x-axis represents the "G+A" (Goals + Assists) values
  # The y-axis represents the "Value" (Market Value) values
  # The text argument specifies the hover text for each point
  # The color argument assigns different colors to each cluster
  plot_ly(plot_data, x = ~`G+A`, y = ~Value,
          text = ~paste("Name:", Player, "<br>Age:", Age,
                        "<br>G+A:", `G+A`, "<br>Value:", Value),
        mode = "markers", color = ~Cluster,
        colors = RColorBrewer::brewer.pal(4, "Dark2")) %>%
    # Customize the plot layout
    # The title argument sets the title of the plot
    # The xaxis and yaxis arguments set the labels for the x-axis and y-axis
    # The hovermode argument sets the behavior to "closest" point
```

```
    layout(title = "Hierarchical Clustering of Players by G+A, Value, and Age",
           xaxis = list(title = "Goals + Assists (G+A)"),
           yaxis = list(title = "Value, in millions of pounds(£)"),
           hovermode = "closest")
})
```

This visualization not only provides a clear representation of how player attributes are grouped but also allows users to interact with the data points. By clicking on individual markers, users can access detailed information about each player, such as name, age, and specific performance metrics.

User Interaction and Data Table

The interactive plot lets users explore and visually interpret complex patterns in player data. Below the plot, a data table provides further exploration, featuring a dropdown menu that allows users to select a cluster, displaying players in that cluster in the table.

The cluster selection dropdown is updated dynamically with available clusters using an observer that monitors changes to **`plot_data`**. The **`updateSelectInput`** function refreshes the dropdown options with unique cluster labels.

The table is implemented using the **`renderDT`** function. Before rendering, the **`req`** function ensures a cluster is selected, and the table is filtered accordingly using **`filter`** and **`select`** functions. The **`datatable`** function then creates an interactive table, displaying key columns (Player, Age, G+A, Value, Cluster), and the table is paginated, showing 10 rows at a time with search highlighting disabled.

```
# Update the cluster selection dropdown based on the available clusters
# The observe function is used to update the choices of the cluster selection
# It is triggered whenever the plot_data reactive expression updates
observe({
  # Update the choices of the "clusterSelect" dropdown input
  # The levels function extracts the unique cluster labels
  # from the "Cluster" column of the plot_data
  # The updateSelectInput function updates the choices of the dropdown
  # input with the cluster labels
  updateSelectInput(session, "clusterSelect", choices = levels(plot_data()$Cluster))
})

# Render the cluster table
# The renderDT function renders an interactive table using the DT package
output$clusterTable <- renderDT({
```

```
  # Ensure that a cluster is selected before rendering the table
  req(input$clusterSelect)
  # Filter the plot_data based on the selected cluster
  # The filter function selects only the rows corresponding
  # to the selected cluster
  # The select function chooses the columns to be displayed
  # in the table (Player, Age, G+A, Value, Cluster)
  filtered_data <- plot_data() %>%
    filter(Cluster == input$clusterSelect) %>%
    select(Player, Age, `G+A`, Value, Cluster)
  # Create an interactive table using the datatable function
  # The options argument sets the number of rows per page to 10
  # and disables search highlighting
  datatable(filtered_data, options = list(pageLength = 10, searchHighlight = FALSE))
})
```

By integrating hierarchical clustering with interactive visualizations, this tab offers users a robust tool to analyze the football market dynamically and in-depth. This methodological advancement adds another dimension to the analysis of football player data, providing valuable insights into the structure and dynamics of the football market.

Results

Nationality Analysis

An analysis of the distribution of player market values by nationality reveals a strong concentration of high-value players in Europe, using £70 million as a benchmark value to denote expensive players. This pattern reflects Europe's dominant role in the global football market as both a hub for producing top talent and the primary location for the world's leading leagues and teams. England, in particular, stands out with a concentration of high-value players more than double that of any other country, providing empirical support to the concept of the "English Tax." This suggests that English players are often priced higher than their counterparts from other nations, likely due to the premium placed on local talent in the Premier League.

In terms of age distribution, South America emerges as a prominent source of younger high-value talent. This could indicate that South American players are recognized for their potential at younger ages, often due to their creative technical skills and early exposure to professional play in domestic leagues. Football is incredibly popular in South America, which contributes to a vibrant culture that nurtures talent from a young age. This trend is exemplified by players like Rodrygo and Federico Valverde who have garnered attention and significant market valuations at relatively young ages. Rodrygo, from Brazil, made a high-profile move to Real

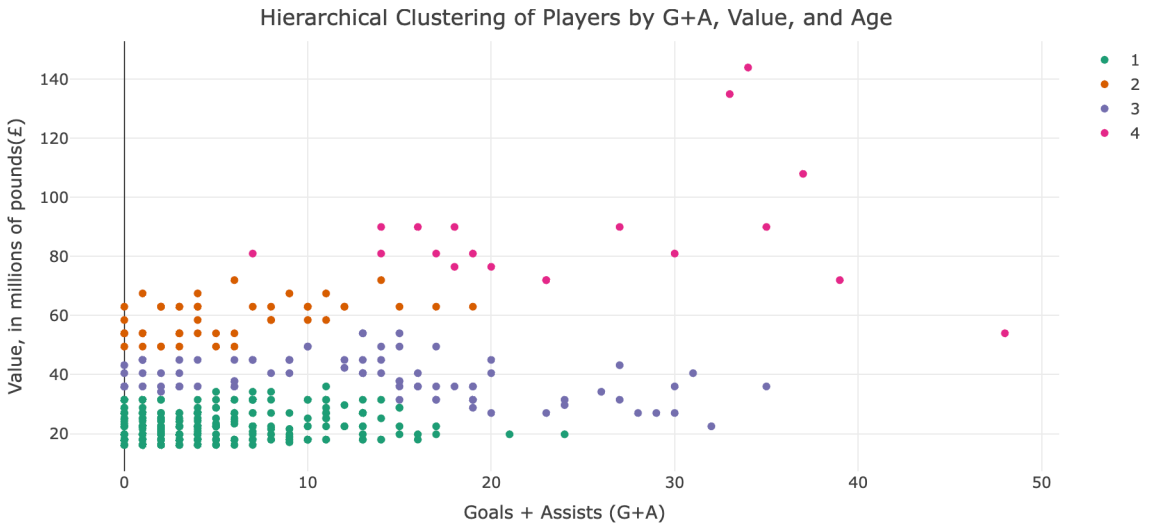
Madrid in 2019 at the age of 22 for a reported fee of around £38 million. In 2024, he is valued at £85 million and is viewed as one of the most prolific forwards in the world (*Bleacher Report*, "Rodrygo Goes Completes €54 Million Transfer"). Similarly, Valverde moved from Uruguayan club CA Peñarol at a young age to Real Madrid and is now valued over £100 million as one of the most dynamic midfielders today (*Managing Madrid*, "Federico Valverde: The Generational Swiss Army Knife"). These examples demonstrate that South American players are often recognized for their potential at younger ages, which not only boosts their individual market values but also indicates a broader trend of South American talent being pivotal in shaping the strategies of top European clubs.

League Analysis

The league-wise breakdown of player nationalities offers insights into recruitment strategies across different European leagues. The German Bundesliga and French Ligue 1 are noted for their wide distribution of player nationalities, including talents from less traditional football continents such as North America and Asia, and less recruitment from their own countries compartively. This indicates a more global scouting approach, likely aimed at uncovering undervalued talent and broadening fan base appeal internationally. It is also indicative of a less distinct style of play in these leagues, which allows foreign-born players to better adapt to the style of play in Germany and France.

By contrast, the English Premier League and Spanish La Liga show a stronger inclination towards nurturing and retaining local talent. This trend is influenced by tradition, style of play, and league policies, such as the Premier League homegrown rule that limits the number of foreign players allowed on a roster. The Premier League is known for having a more direct, fast-paced, physical sytle of play that can make it difficult for foreign players to adapt (*Medium*, "Let's Get Physical"). The Premier League, in particular, features a high number of English players, which could also be reflective of the aforementioned "English Tax," where local players command higher market values, making them significant assets both on and off the field. Likewise, La Liga has a very distinct style of play, where players are required to have exceptional technique and game intelligence, which may cause challenges for players without the proper training or experience. The Italian Serie A emerges in-between, with more diversity than the Premier League and La Liga, but less than the Bundesliga and Ligue 1. This suggests the existence of a distinct Italian style that emphasizes defensive structure and efficiency. However, the substantial presence of foreign players also suggests that this style may be more adaptable for internationals than the styles in England or Spain.

Clustering Analysis



The clustering analysis, visualized through the scatter plot, provides a structured overview of how goals and assists (G+A), market value, and age interrelate across the data:

1. **Cluster 1 (Green):** This cluster mainly consists of players with lower market values and modest performance stats. It likely includes younger or developing players from smaller clubs. Their lower market values could be reflective of their current impact on the field, which may be limited compared to more established stars.
2. **Cluster 2 (Orange):** Featuring a mix of ages and positions, this cluster has players with a moderate performance record and variable market values. This diversity suggests that players in this cluster are valued for more than just their current on-field contributions, such as their potential, versatility, or marketability.
3. **Cluster 3 (Blue):** Dominated by younger players with high market values but lower G+A scores, this cluster underscores the market's investment in potential. Clubs are likely willing to pay premium prices for these players in anticipation of future returns as these talents mature and develop their skills.
4. **Cluster 4 (Pink):** The standout cluster for high market values, primarily consisting of midfielders and forwards with substantial G+A figures. This cluster highlights the premium placed on players who directly contribute to scoring, underlining the market's valuation of offensive players. Notably, the absence of defenders and goalkeepers in this cluster might indicate a lower valuation for defensive roles in the current market.

The analysis underscores the significant impact of nationality on player market values and illustrates how individual leagues' strategic choices shape their team compositions and transfer activities. The clustering further outlines the attributes valued in the football market,

emphasizing the premium on youth and offensive capabilities. Together, these findings offer a nuanced understanding of the dynamics shaping player valuation and recruitment in global football.

Conclusion

Limitations

While the analysis presented offers significant insights into player valuation and the influence of nationality and league on football markets, several limitations must be acknowledged. Firstly, the scope of data is restricted to top European leagues and high-value players, potentially overlooking lower leagues and less prominent footballing nations where market dynamics could vary. Furthermore, the reliance on publicly available market values and transfer fees may not accurately capture the true economic worth of players due to undisclosed financial details and factors unrelated to performance, such as marketing potential.

In addition, the correlations identified between player attributes and their market values should not be construed as causal relationships. Factors such as club financial health, investment strategies, and broader economic conditions could also significantly influence these dynamics but were not explored in this study. Furthermore, the findings are primarily applicable to high-level professional football and may not generalize across different levels of play or other sports.

Future Research

To enhance the applicability of future research, several improvements could be made. Expanding the dataset to include more leagues globally and incorporating lower divisions would offer a more comprehensive view of the football market. A longitudinal analysis could provide insights into how player values and performances evolve over time, offering a perspective on career progression and valuation. Additionally, researchers could develop methods to implement polylines more efficiently, potentially without using a for loop, to streamline the analysis and improve the runtime of the code.

Incorporating qualitative data, such as player interviews, expert opinions, and scouting reports, could also improve the analysis by adding context to numerical findings, especially in understanding intangible factors affecting player valuations. Finally, studying the economic impact of player transfers on leagues and clubs could offer a holistic view of the financial implications of market dynamics. By addressing these limitations and considering these possibilities for future research, new studies could offer even more nuanced understandings of the football market.

References

Johnson, Sarah. "Understanding Hierarchical Clustering in Data Science." *Towards Data Science*, Medium, 1 Jan. 2023, https://towardsdatascience.com/understanding-hierarchical-clustering-d3068627cc75.

"A Look at the English Tax Issue." *Over the Line Sports*, Over the Line Sports, no publication date, https://overthelinesports.ca/soccer/f/a-look-at-the-english-tax-issue?blogcategory=Soccer.

"Adding a Leaflet Polyline." *WRLD3D*, WRLD, https://www.wrld3d.com/wrld.js/latest/docs/examples/adding-a-leaflet-polyline/.

"Federico Valverde: The Generational Swiss Army Knife." *Managing Madrid*, Vox Media, 3 Feb. 2024, https://www.managingmadrid.com/2024/2/3/24058138/federico-valverde-the-generational-swiss-army-knife.

"Hierarchical Clustering." *Statistics and Machine Learning Toolbox - MATLAB & Simulink*, MathWorks, https://www.mathworks.com/help/stats/hierarchical-clustering.html.

"Hierarchical Clustering." *Statistics How To*, Dotdash Meredith, 2022, https://www.statisticshowto.com/hierarchical-clustering/.

"How to Add Interactive Maps to R Using Leaflet." *Stack Overflow*, 12 March 2022, https://stackoverflow.com/questions/6703185/how-to-add-interactive-maps-to-r-using-leaflet.

"Let's Get Physical: Comparing the Big 5 European Football Leagues." *Medium*, Skillcorner, 22 Apr. 2020, https://medium.com/skillcorner/lets-get-physical-comparing-the-big-5-european-football-leagues-bcb04ebb835c.

"Manchester City Sign Jack Grealish from Aston Villa for British Record Fee." *ESPN*, ESPN Internet Ventures, 5 Aug. 2021, https://www.espn.com/soccer/story/_/id/37619538/manchester-city-sign-jack-grealish-aston-villa-british-record-fee.

"observeEvent." *Shiny Developer Center*, Posit, https://shiny.posit.co/r/reference/shiny/0.11/observeevent.

"Rodrygo Goes Completes €54 Million Transfer from Santos to Real Madrid." *Bleacher Report*, Turner Broadcasting System, 15 June 2018, https://bleacherreport.com/articles/2781233-rodrygo-goes-completes-eur54-million-transfer-from-santos-to-real-madrid.

"Shapes." *Leaflet for R*, RStudio, https://rstudio.github.io/leaflet/articles/shapes.html.

"Using Leaflet in R." *R-Bloggers*, 18 June 2021, https://www.r-bloggers.com/using-leaflet-in-r/.

Appendix

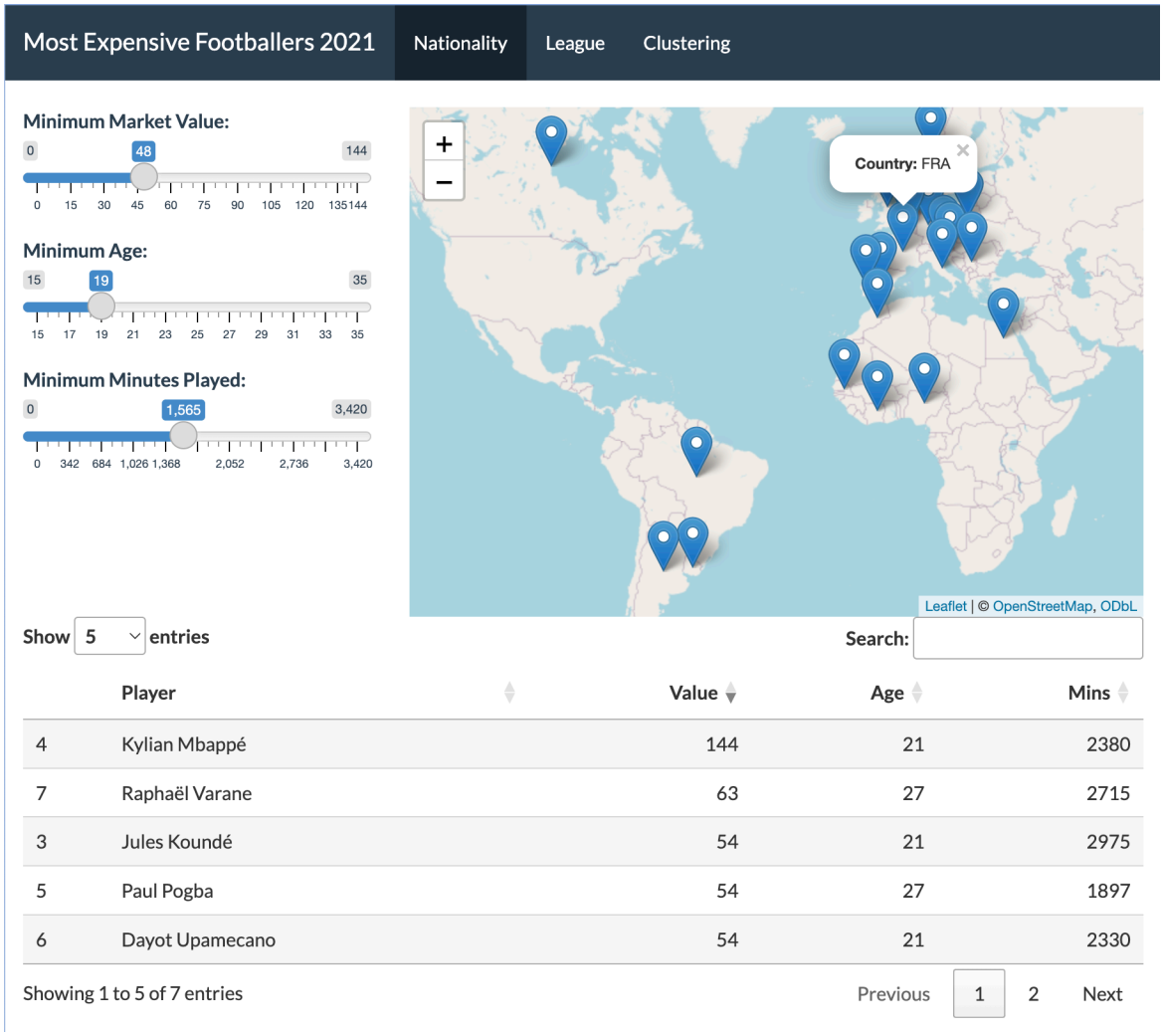
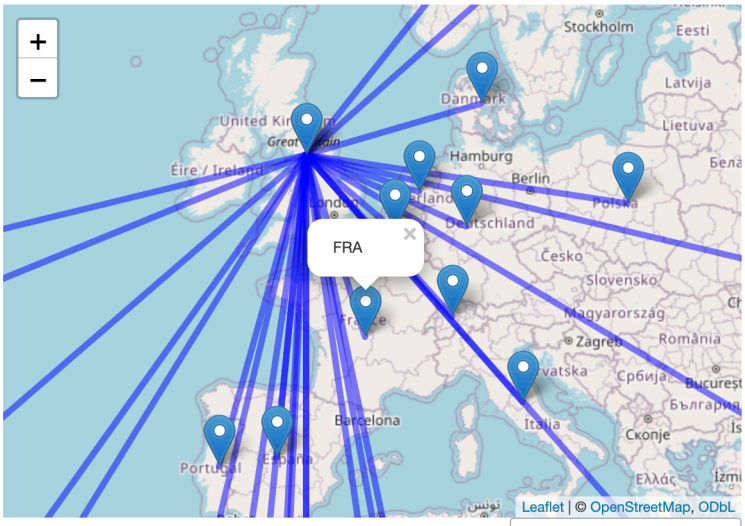


Figure 1: Page 1 - Nationality

Most Expensive Footballers 2021 | Nationality | League | Clustering

Choose a league:

Premier League



Show 5 entries

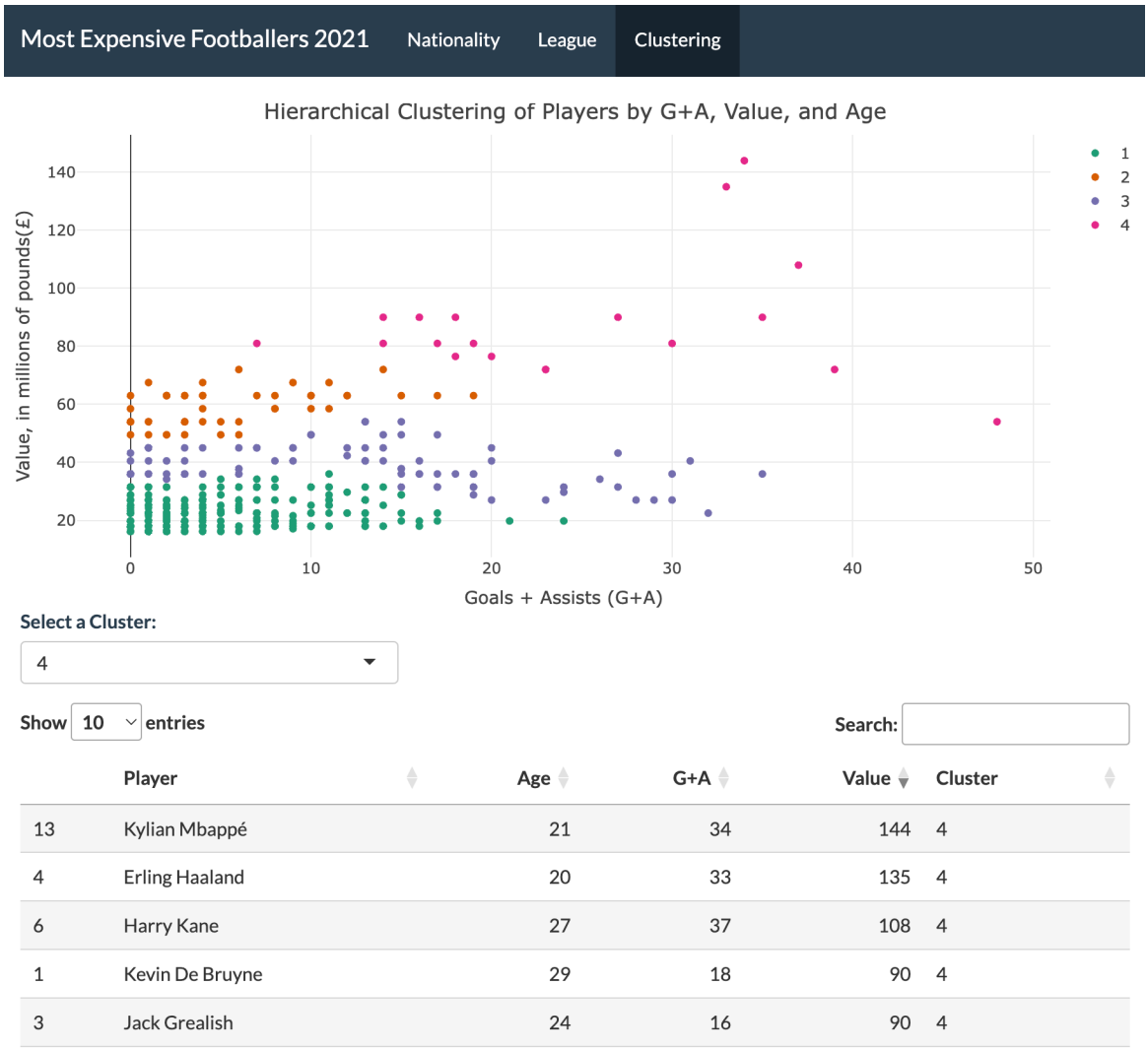
Search:

	Player	Nation	Age	Value	G+A
1	Lucas Digne	FRA	27	31.5	7
2	Wesley Fofana	FRA	19	36	0
3	N'Golo Kanté	FRA	29	49.5	2
4	Alexandre Lacazette	FRA	29	19.8	15
5	Anthony Martial	FRA	24	31.5	7

Showing 1 to 5 of 10 entries

Previous 1 2 Next

Figure 2: Page 2 - League



	Player	Age	G+A	Value	Cluster
13	Kylian Mbappé	21	34	144	4
4	Erling Haaland	20	33	135	4
6	Harry Kane	27	37	108	4
1	Kevin De Bruyne	29	18	90	4
3	Jack Grealish	24	16	90	4

Figure 3: Page 3 - Clustering

Clustering Dendrogram Code

```
final_data <-
  readRDS("~/Documents/GitHub/stat231Tremante/Projects/final-project/
          data:wrangling/final_data.RDS")

# Calculate the Euclidean distance matrix
dist_mat <- dist(final_data[, c("G+A", "Value", "Age")])

# Perform hierarchical clustering using the Ward's method
hclust_res <- hclust(dist_mat, method = "ward.D2")

# Cut the hierarchical tree into 4 clusters
clusters <- cutree(hclust_res, k = 4)

# Convert the hclust object to a dendrogram object
dend <- as.dendrogram(hclust_res)

# Assign colors to the branches based on cluster assignments
dend <- color_branches(dend, k = 4, col = c("red", "green", "blue", "purple"))

# Plot the colored dendrogram
plot(dend, main = "Dendrogram of Player Clustering", sub = "",
     xlab = "",
     ylab = "Height")
```