Athens University of Economics and Business

School of Business

Department of Management Science & Technology

Master of Science in Business Analytics

Program:	Full-time
Quarter:	1st (Fall Quarter)
Course:	Data Management and Business Intelligence
Assignment №:	2
Students (Registration №):	Gkouma Konstantina (f2822208), Souflas Eleftherios-Efthymios (f2822217)

Table of Contents

Data Management & Business Intelligence
Assignment #2

Students: Gkouma Konstantina, Souflas Eleftherios-Efthymios

Introduction & Business Goal

People should not gamble but earn their money from their regular jobs. This is not an ethical or philosophical statement but reality and personal experience. There is no person that I know, including us, that have earned more from betting than the amount that they lost from it. On the other hand, there are a lot of people that have lost their fortunes on bets, mainly due to the factor that they became addictive to it. Predicting sports results and making a bet on the outcome is known as sports betting. The frequency and types of sports betting varies according to culture and country of residence, with most bets being placed on amateur and professional levels of football/soccer, American football, basketball, baseball, hockey, track cycling, car racing, mixed martial arts, and boxing.

Owning or investing on a betting/gambling company can be very profitable. Online betting and sports gambling apps are now widely used, so that users can bet from any of their personal devices (PC, laptop, phone, tablet or even TV), contributing to global expansion to $52 Billion in 2018 (Symphony Solutions, 2019). In the year 2020, the COVID-19 outbreak impacted the sector greatly. Sports betting were forced to shut their businesses for most of the year due to the ongoing coronavirus restrictions and lockdowns which postponed many sports and tournaments globally. During the year 2020, the market value of sports betting in the Europe region saw a de-growth of 11%. The sports betting market in Europe region saw marginal decline of 1% in 2021 as the sector witnessed a recovery in demand after easing the restrictions which were placed on account of the pandemic. The market size of the sports betting sector in Europe region reached a value of $23.63 billion in 2021 (GlobalData Plc, 2022).

So, if sport betting companies gain large profits it is due to the existence of betters who bet their money on their odds and of course they lose their money. And sometimes an enormous amount of it. One solution to this problem and the most efficient one is to persuade betters not to bet again in their life. Due to the aspect of “easy money” and the fun of being lucky (rarely) it is a rather difficult case for someone to make the most efficient solution feasible. So, we should search for an alternative one.

Someone could take advantage of big data that are flooded everywhere in our web universe, collect them, load them in a database and extract some derivatives making use of descriptive, predictive, or even prescriptive statistics. In that case, he/she would not avoid losing some of his money but could minimize that amount as close to zero as possible.

Dataset

To support this case study, we found a dataset in https://www.kaggle.com/ called “European Soccer Database” from Hugo Mathien (https://www.kaggle.com/datasets/hugomathien/soccer) and downloaded it. The dataset is an SQLite file (.sqlite) containing 25979 matches from 11 European Countries with their lead Championship of seasons 2008/2009 to 2015/2016. It also contains 11060 players and 299 teams with players’ and teams’ attributes sourced from EA Sports' FIFA video game series, including the weekly updates. For every match, betting odds from up to 10 providers are included together with their outcome.

ETL Procedure

Because our dataset was not a simple CSV file, we downloaded the appropriate drivers for the SQL Server Integration Services to communicate to the SQLite database file and we followed the instructions posted in https://rainydaycode.wordpress.com/2013/05/27/connecting-to-sqlite-through-ssis/. After the creation of the SQLite Connection (Figure 1), we created the appropriate tables in the

SQL Server Database named “DMBI”. We loaded only the columns that we decided that are useful to us to reduce the duration of the “Data Flow” task of SSIS to the absolutely needed one. Because there were many tables to be loaded data to, we created a separate data flow task with ADO.NET Source and an SQL Server Destination for each table and we entitled them in a “Sequence Container”.

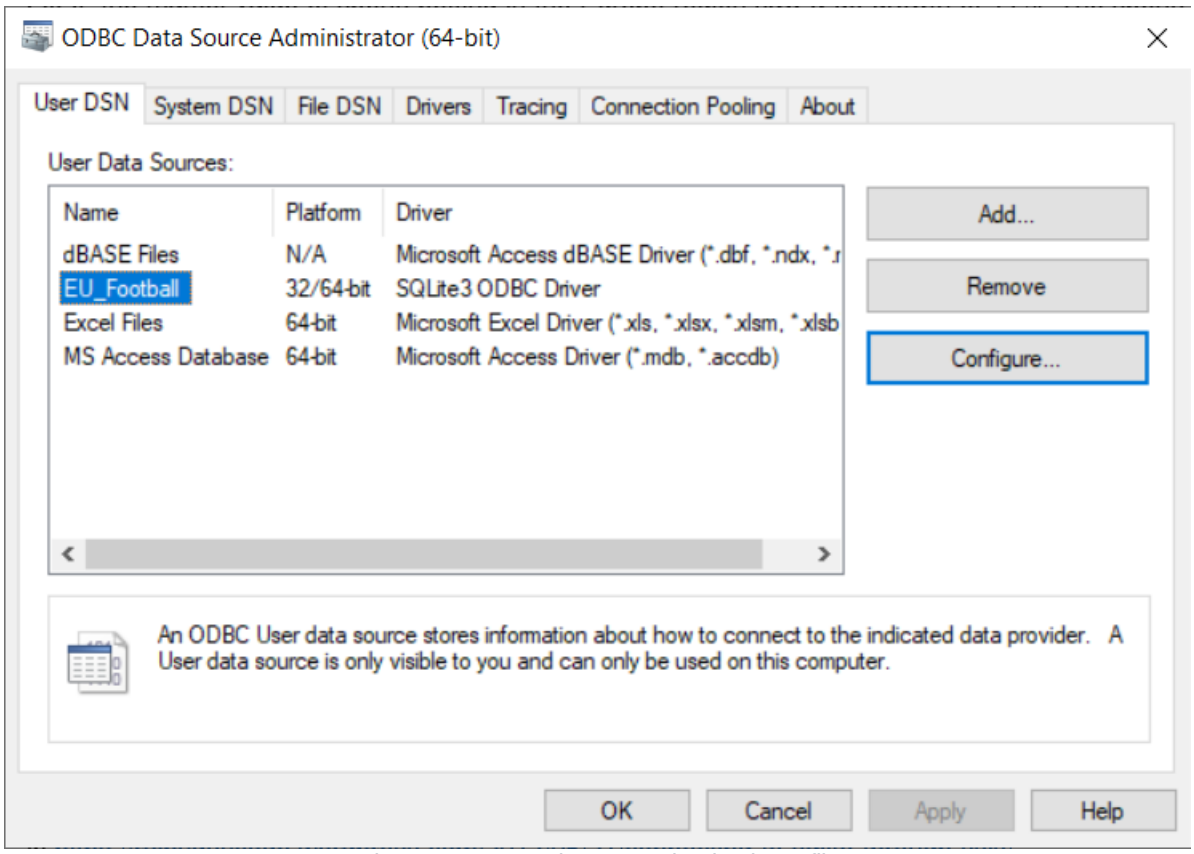


Figure 1 - Connection to SQLite database file

We then converted all NVARCHAR(MAX) columns to VARCHAR because SQL Server treats the first case as TEXT values and cannot be added as UNIQUE KEYS. We wanted some columns to be unique to exclude the case where a next load of data adds an already existing row to a Dimension Table. We also added an Identity Primary Key to Dimensions. Also, there were some columns that we wanted to be Dimensions and a separate table with an id did not exist. So, we took advantage of the Analytic Function “ROW_NUMBER() OVER(ORDER BY column)” in order to tackle with this problem. We also created a separate Date Dimension, after we converted all varchar-like dates to date data-types and we included foreign keys to all date columns pointing to the primary key of the Date Dimension.

We also did several “cleaning data” jobs like:

- We discovered that in the Attributes of the Players, columns “Attacking Work Rate” and “Defensive Work Rate”, were given only three certain values by FIFA (FIFAUTeam, 2021). So, we set it all other values, including NULL values, to N/A.
- We realized that in the Team table there were some duplicate values in the full name column of a team but with different id. Thus, we searched for the games that were recorded in the database and we realized that were completely different teams. One of the teams (ŁKS Łódź) was recorded with the name of its city archrival.
- We found a full duplicate Team Attribute with different id and we deleted the second record.

Match table of the SQLite Database include, among others, 30 columns with the betting odds for the three possible results (Win, Draw, Home) of 10 betting companies (BET365, BETSSON, BWIN,

Interwetten, William Hill, among others). It was more convenient for the data handling of the Analysis Services and in general, to create a Fact Table with different rows for the betting odds of each match, betting company and result of match. So, we took advantage of the UNPIVOT relational operator to rotate columns of the SQLite Match table into column values in the Bets Fact table discarding the rows that contained null values for all columns. Thus, we created a Bets Fact table of 575910 rows.

After the creation of all tables, we added all primary, unique, and foreign keys to the tables in order to create the relations between the tables. The Entity-Relationship Diagram of the Data Warehouse is as being showed in Figure 2. It contains three Fact Tables: Bets, Team Attributes and Player Attributes.

- The Bets Fact Table is related to the Bet Result Dimension, containing the IDs for the outcomes of which the odd is assigned to, the Bet Company Dimension containing the 10 Bet Companies and the Match Dimension containing all details for each match which is then related to the League Dimension (which is related to the Country Dimension), the Season Dimension, the Player Dimension, the Team Dimension and the Date Dimension.
- The Team Attributes Fact Table is related to several Dimension containing details about the tactics each team used for each year, including Defense, Build Up (Midfielder) and Chances (Attacking) Attributes, and Team and Date Dimensions.
- The Player Attributes Fact Table is related to Defense and Attack Work Rate Dimensions, a Preferred Foot and a Player and Date Dimensions.

We then created an SSIS package to handle future updates of our Data Warehouse, as it contains data till 2015/2016 season. Since our data are not going to be updated, as they contain past events and attributes, we need only insert statements. After we created an “Execute SQL” task to delete and create the intermediate (staging) tables, we created a “Sequence Container” to include all “Data Flow” tasks, as forementioned. After the above are done a series of “Execute SQL” Tasks are being executed in parallel (where possible) to clean the data that were needed to be cleaned and update Dimension and Fact Tables. The “update” of the tables is actually a “MERGE INTO USING ON WHEN NOT MATCHED THEN INSERT VALUES” statement in order only rows with new id to be inserted. We discovered that, when executing parallel queries, some of them were inserting or updating on the same table at the same time and giving back an error output. This was done because while inserting or updating a table from the first query, the table was locked, and the second execution was trying to do the same and failing. We found those queries and made them executed the one after the other (Figure 3 and Figure 4).

Analysis Services (Cube and Reporting)

We then followed a straightforward procedure of creating our OLAP Cube on top of our Data Warehouse with the SQL Server Analysis Services of Microsoft. If the Data Warehouse is correctly designed there is no chance of getting an error when creating the OLAP Cube. We decided to create one OLAP Cube for the entire Data Warehouse instead of one OLAP Cube per Fact Table following the recommendations and instructions found here (https://www.youtube.com/watch?v=TnyRsO4NJPc). The main problem is the handling of duplicate dimension tables, because one dimension table of one fact table is a dimension for other fact tables also, so if we wanted to create a report (as we do) that includes several Fact Tables, the Dimension Table that is linked to one Fact Table will be different from the same Dimension Table linked to the other Fact Table. It was a business decision that we chose to follow.

There are several reporting capabilities that SSAS provide to its users. One of them is to create a calculated measure. In our case, only counting the betting odds, that the OLAP Cube provides when created, does not mean anything or to declare it better does not provide any insight on the data that we have. We would like to have the average betting odds that all ten betting companies provided for each match’s outcome. In Figure 6, we can see an example of the creation of a calculated measure on our OLAP Cube. With the addition of the above calculated measure we can create a report (Figure 7), just by using drag & drops, of the average betting odd per match of English Premier League of the 2015/2016 season (the season where Leicester City surprised the world and became the champion). Unclicking design mode we can view the real MDX language used (Figure 8).

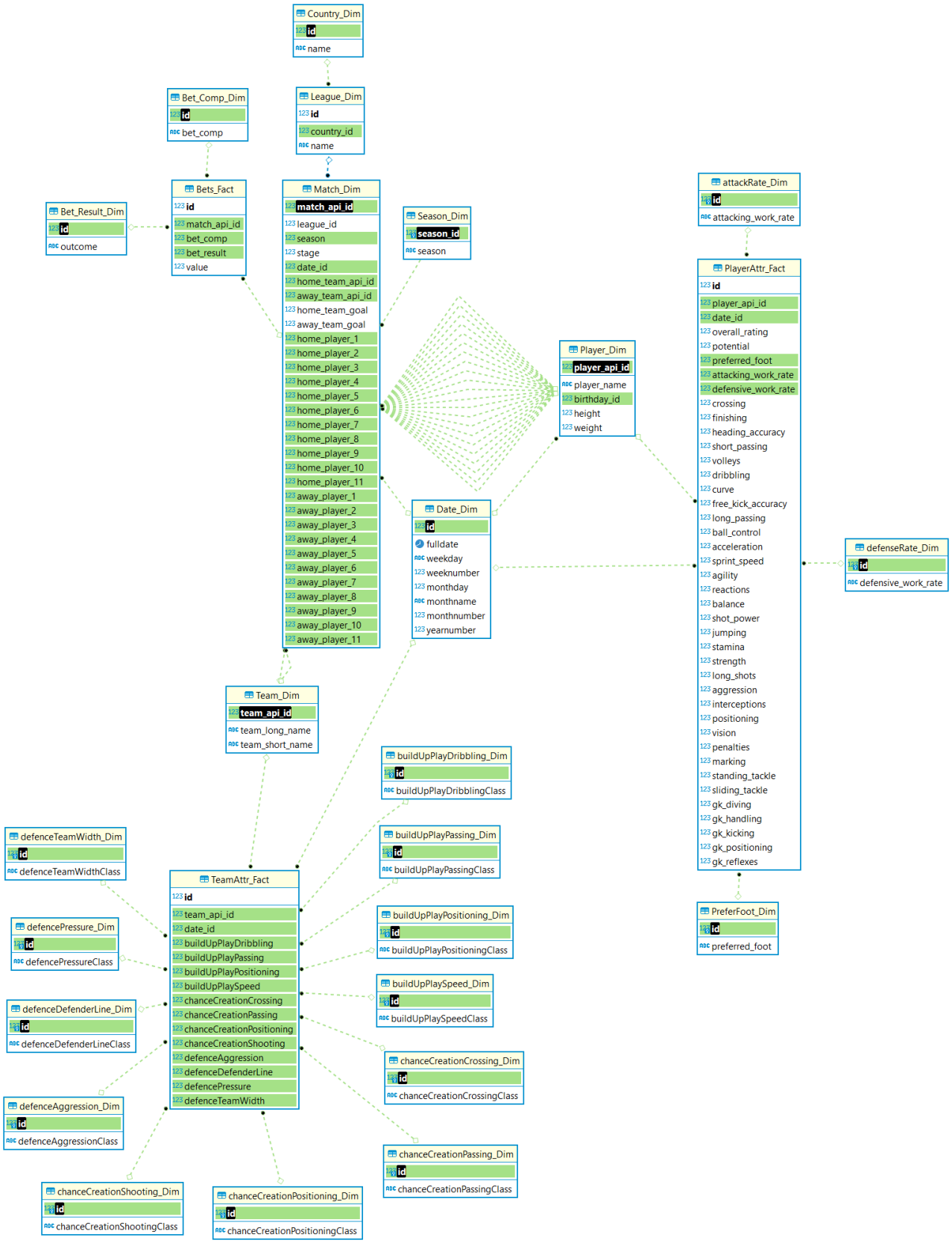


Figure 2 - ER Diagram of Data Warehouse

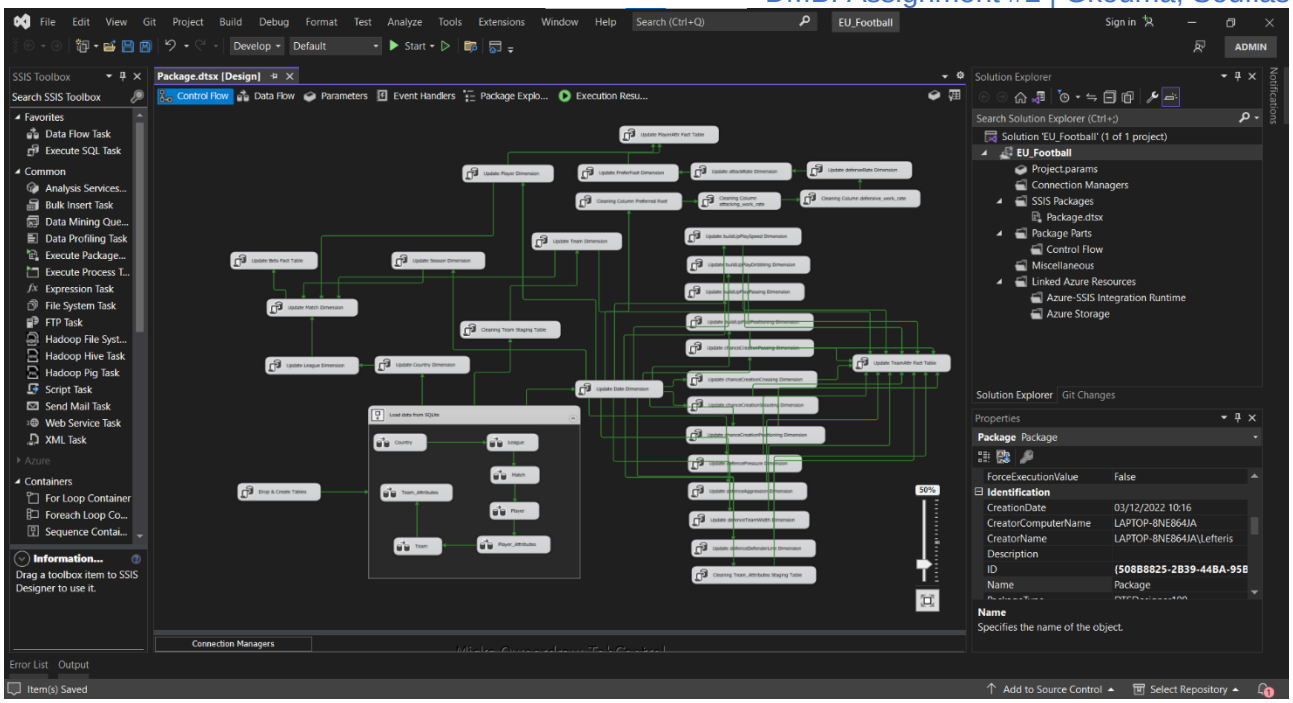


Figure 3 - SSIS Package Control Flow

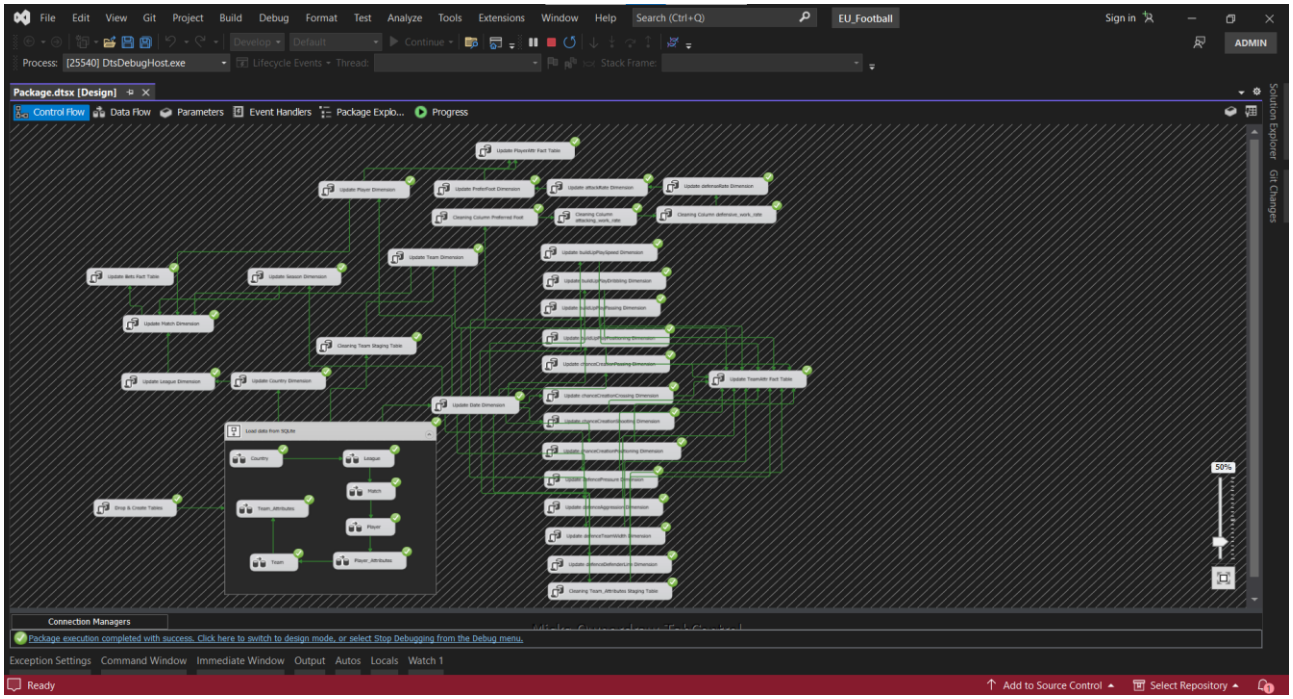


Figure 4 - SSIS Package execution success

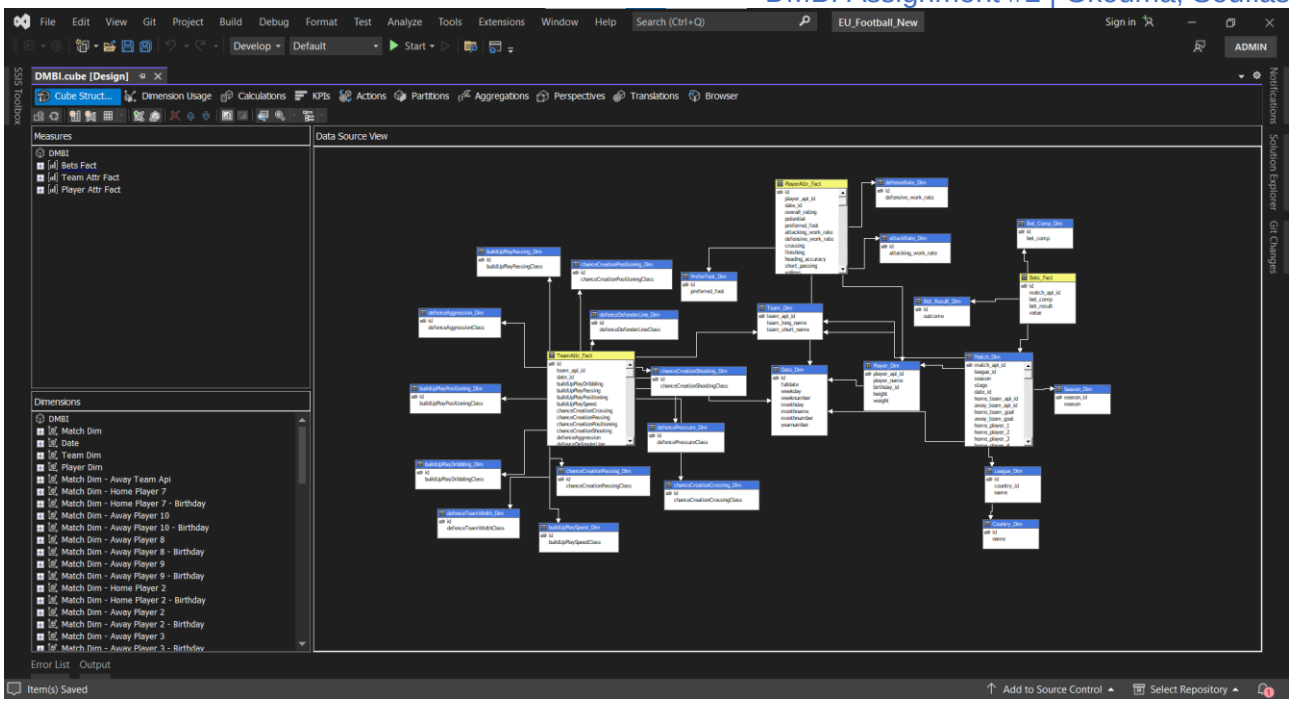


Figure 5 - Cube Structure

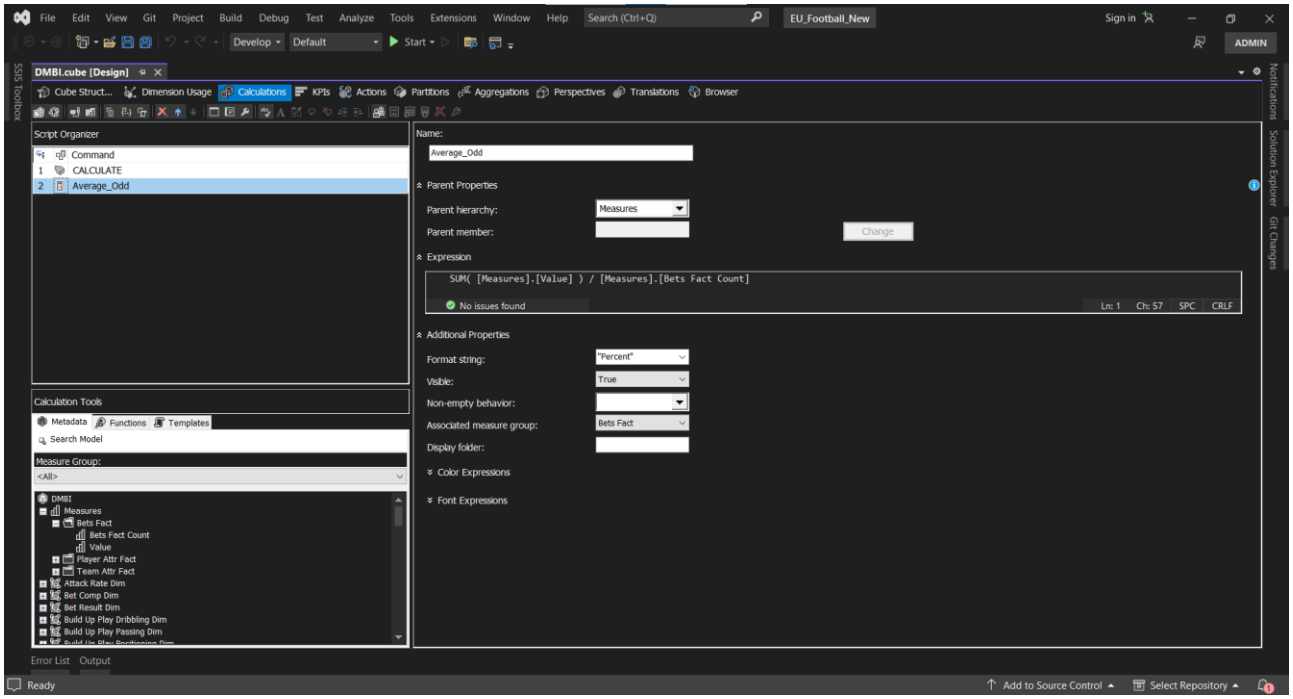


Figure 6 - Create calculated measure Average Odd

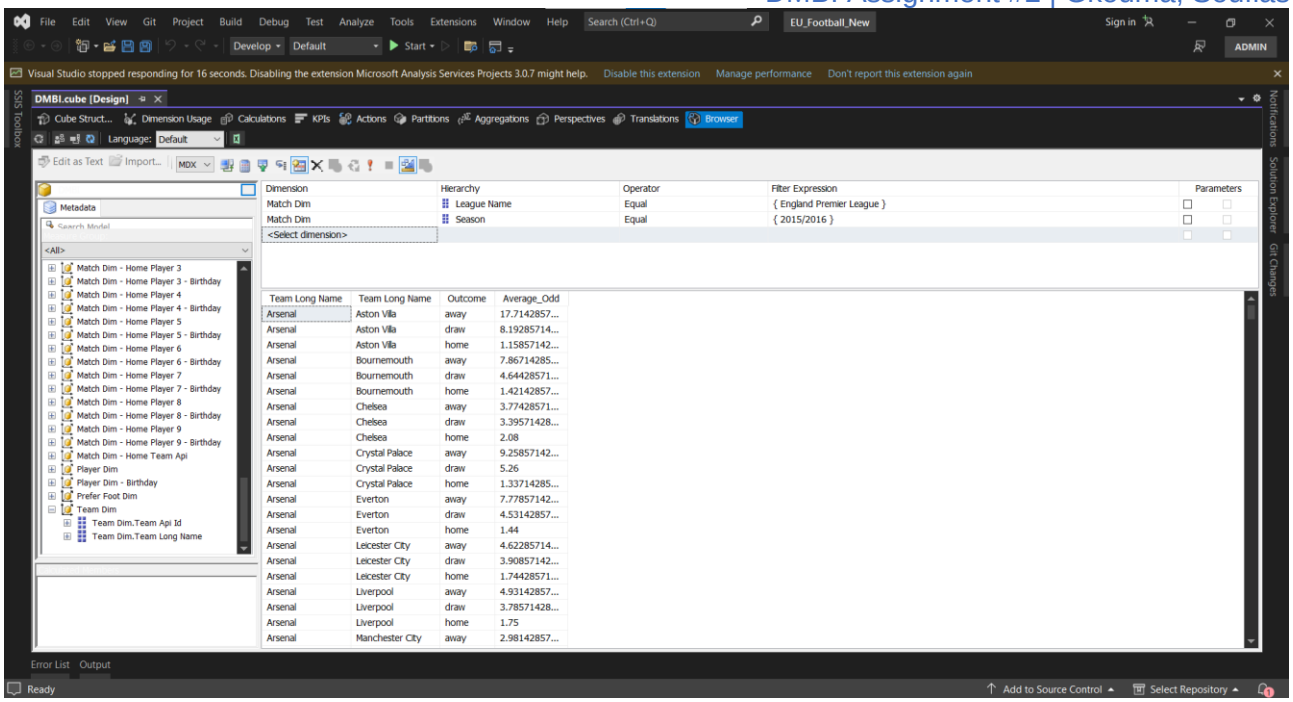


Figure 7 - Report of the average English Premier League 2015/2016 season betting odds

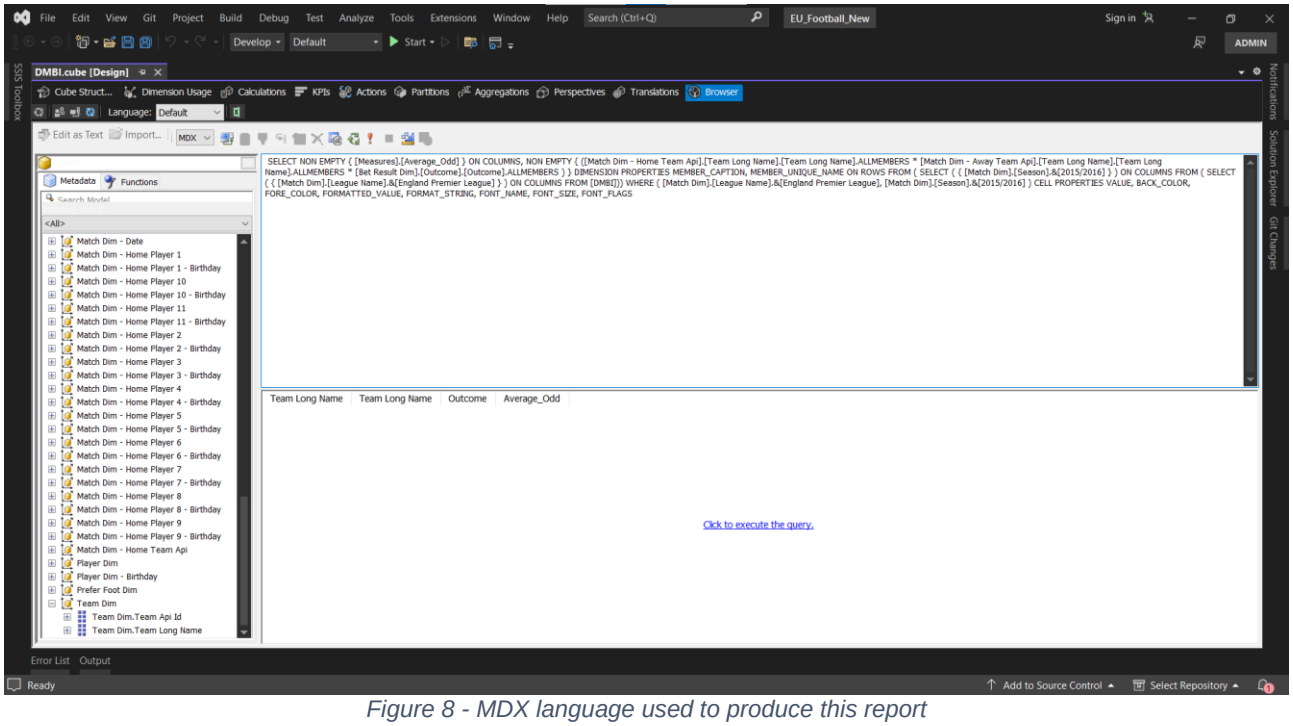


Figure 8 - MDX language used to produce this report

One other reporting capability of SSAS is that someone can create a drill-through action. Dimensioning fact data by a fact dimension without correctly filtering the data that the query returns can cause slow query performance. To avoid this, you can define a drill-through action that restricts the total number of rows that are returned. This will significantly improve query performance. Suppose we wanted to know what tactics Leicester City, of our previous example, in the season of 2015/2016 used. We can create a drill-through action named “Drill Through Team” and add all drill-through columns with their respective names (Figure 9). After deploying the solution, we can click on browser, reconnect and then “Analyze in Excel”. When the Excel Window pops up under the tab PivotTable Fields we can add the team name, filter selecting Leicester City and after adding the Team Attributes Fact Count, we can select the Quick Explore option and add one by one the fields we want to drill through (an example is shown on the Figure 10).

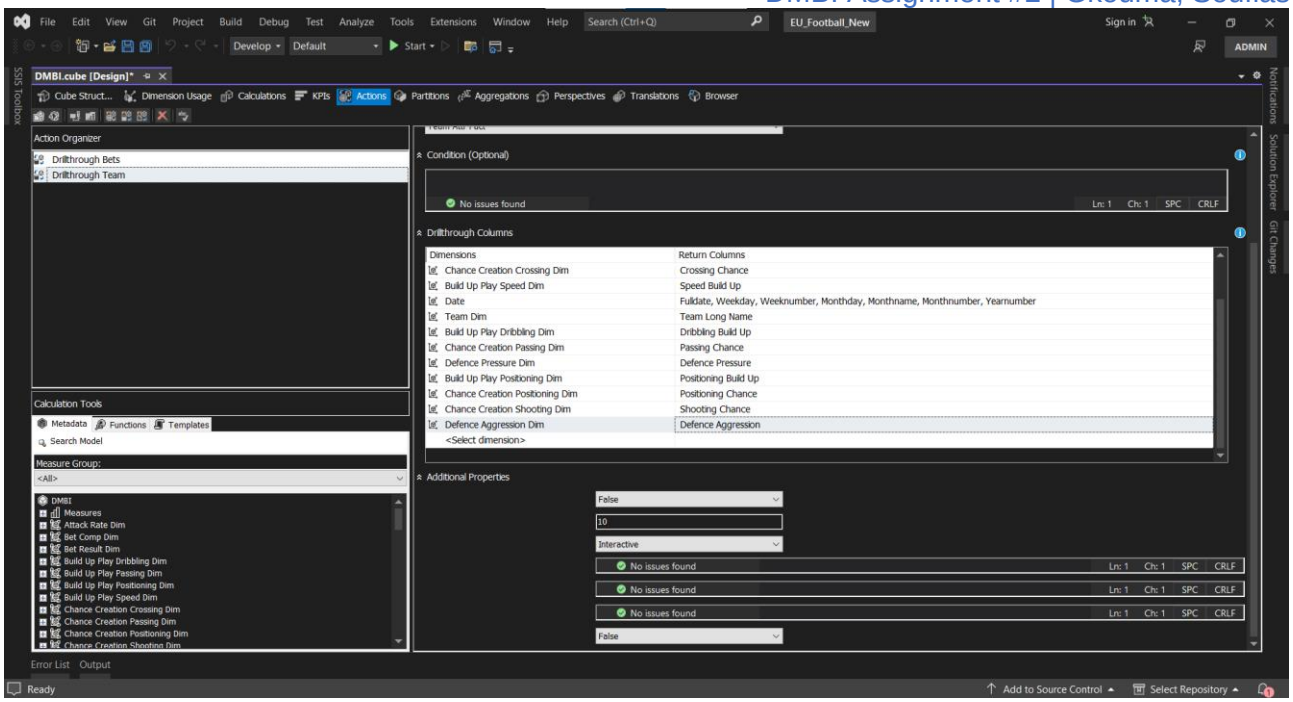


Figure 9 - Create Drill Through Action

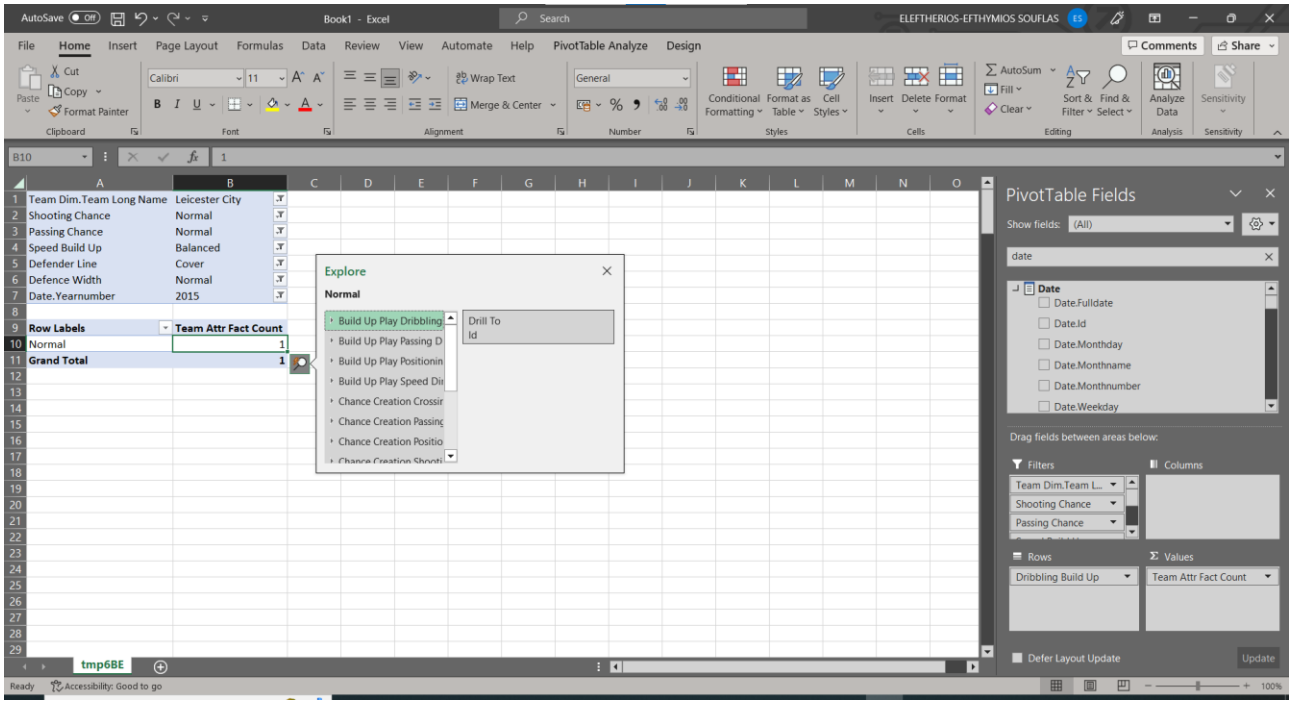


Figure 10 - Excel PivotTable Analyze

Visualizations

The best part and the most interesting one, as someone can derive a large number of conclusions, is the visualization of the data. Visualizations can produce a better understanding of how data points are distributed, how scattered they are, if they follow some patterns and maybe forecast the future.

For this part of the assignment, we have decided to use Tableau to produce some insights for betters that insist betting, in order to help them not to lose all their fortune. Betters should be able to know which Championships are the most predictable ones and thus bet for the lowest betting odd and which Championships betting companies drastically fail to predict outcomes and so head for the highest betting odd. The better should combine a set of high odds together with low odds in order to increase their profit.

We created a connection with the Data Warehouse and on the Data Source tab we double clicked "New Custom SQL". On the pop-up window "Edit Custom SQL" we pasted our SQL Queries. We chose the radio button "Extract" to include all data the query produces.

For our first visualization we pasted the following Query:

```
select c.name as league, d.season, 100*round(cast(a.outcomes_predicted as
float)/cast(b.total_matches as float),2) as percent_outcomes_predicted
from (
select [league_id], season, sum(cnt) as outcomes_predicted
from (
select c.[league_id], c.[season], case
           when a.[bet_result] = 1 and b.result < 0 then 1
           when a.[bet_result] = 2 and b.result = 0 then 1
           when a.[bet_result] = 3 and b.result > 0 then 1
           else 0
     end as cnt
from (select [match_api_id], [bet_result] from
(select a.[match_api_id], RANK() OVER (PARTITION BY [match_api_id] ORDER
BY avg_bets) as avg_rank, a.[bet_result]
from
(select [match_api_id], [bet_result], avg(value) as avg_bets
from [dbo].[Bets_Fact]
group by [match_api_id], [bet_result]) a) b
where avg_rank = 1) a, (select [match_api_id], [home_team_goal]-
[away_team_goal] as result
from [dbo].[Match_Dim]) b, [dbo].[Match_Dim] c
where a.[match_api_id] = b.[match_api_id]
and a.[match_api_id] = c.[match_api_id]
) bets_outcomes
group by [league_id], season
) a,
(
select [league_id], [season], count([match_api_id]) as total_matches
from [dbo].[Match_Dim]
group by [league_id], [season]
) b,
[dbo].[League_Dim] c,
[dbo].[Season_Dim] d
where a.[league_id] = b.[league_id] and a.season = b.season and
a.[league_id] = c.id and d.season_id = a.season
```

Using the stacked bars, it produced the graph of Figure 11. After adding Average Line from the Analytics tab, we can securely draw conclusions. Each bar depicts a Championship and the colors from the season legend mention the amount of the percentage of games predicted per season. We should exclude the Belgium Jupiler League as we have no data for the season 2013/2014. So the most predicted leagues are the Spanish Liga BBVA, the Dutch Eredivisie and the Portuguese Liga ZON Sagres. On the other hand, the leagues that fail to predict the outcome are the French Ligue 1, the Scottish Premier League and the German Bundesliga.

So, someone could say that, although with the previous graph we can see the most predicted league regarding the match outcome, we cannot say the same for the least predicted outcome. A league having high game prediction, predicts that a match ends with a certain outcome (win, draw or loss), but for the least predicted leagues we can say that if the lowest betting odd is on home win, it can

end as draw or loss for the home team. It is also interesting to mention that the average line is set on the 419.11, which divided by the amount of leagues played (8) we have on average a prediction of 52.11 %. So those leagues being below the average, approximately the half of their league games will be predicted.

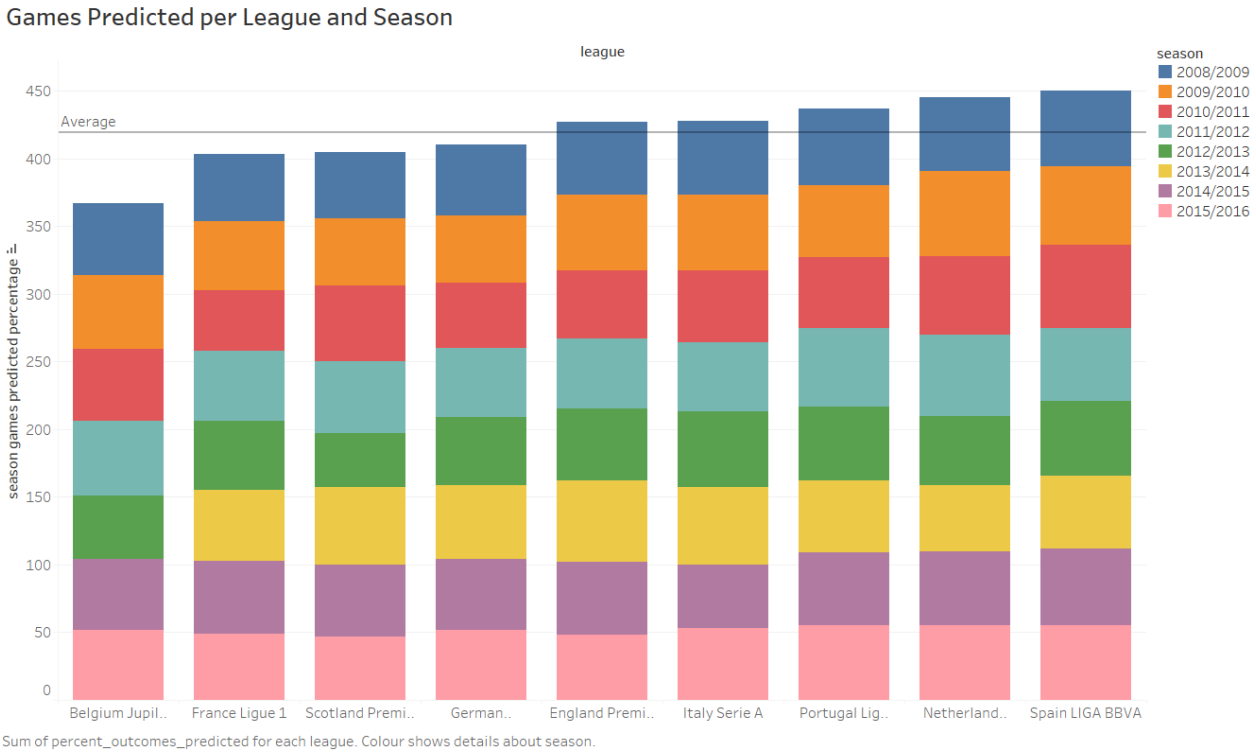


Figure 11 - Games predicted per League and Season

So, someone could ask for taking advantage of the knowledge the data produce, regarding the totally unexpected outcomes per league and season. Those games that the highest betting odd (the least probable outcome) came true.

For this second visualization we pasted on Custom SQL Query of Tableau the following SQL Script:

```
select c.name as league, d.season, 100*round(cast(a.outcomes_predicted as
float)/cast(b.total_matches as float),2) as percent_unexpected_outcomes
from (
select [league_id], season, sum(cnt) as outcomes_predicted
from (
select c.[league_id], c.[season], case
           when a.[bet_result] = 1 and b.result < 0 then 1 --away
           when a.[bet_result] = 2 and b.result = 0 then 1 --draw
           when a.[bet_result] = 3 and b.result > 0 then 1 --home
           else 0
     end as cnt
from (select [match_api_id], [bet_result] from
(select a.[match_api_id], RANK() OVER (PARTITION BY [match_api_id] ORDER
BY avg_bets) as avg_rank, a.[bet_result]
from
(select [match_api_id], [bet_result], avg(value) as avg_bets
from [dbo].[Bets_Fact]
group by [match_api_id], [bet_result]) a) b
where avg_rank = 3) a, (select [match_api_id], [home_team_goal]-
[away_team_goal] as result
```

```
from [dbo].[Match_Dim]) b, [dbo].[Match_Dim] c
where a.[match_api_id] = b.[match_api_id]
and a.[match_api_id] = c.[match_api_id]
) bets_outcomes
group by [league_id], season
) a,
(
select [league_id], [season], count([match_api_id]) as total_matches
from [dbo].[Match_Dim]
group by [league_id], [season]
) b,
[dbo].[League_Dim] c,
[dbo].[Season_Dim] d
where a.[league_id] = b.[league_id] and a.season = b.season and
a.[league_id] = c.id and d.season_id = a.season
```

Using the stacked bars of Tableau again and the average line as previously we created Figure 12.

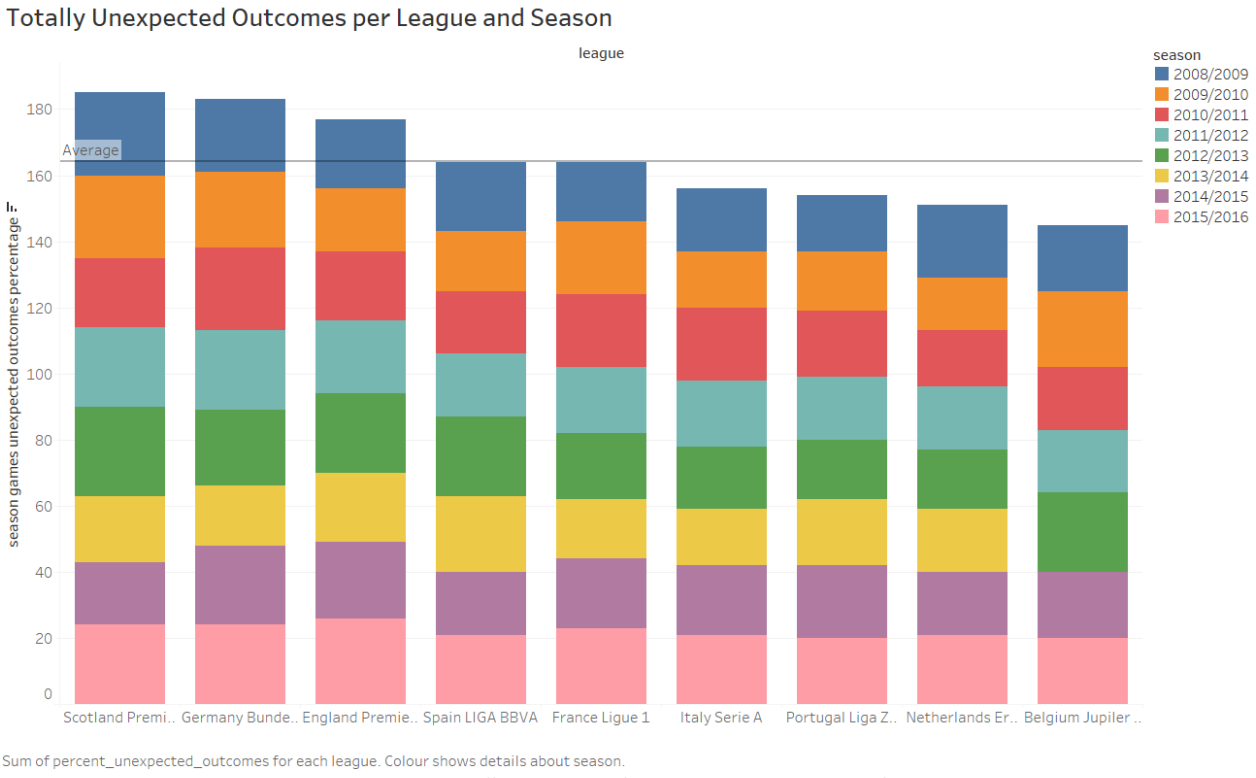


Figure 12 - Totally Unexpected Outcomes per League and Season

From the above figure, we can securely mention that in the Scottish Premier League, the German Bundesliga and the English Premier League, above 1 out of 4 games result to a totally unexpected outcome. But be cautious, 1 out of 4 is a small amount for someone to bet to, but it should be taken into consideration. Also, English Premier League should rather not be count in the three foretold leagues as the 2015/2016 season (with pink colour) was the season that Leicester City won the Championship.

So far, we are looking for the leagues rather than the games, but most of the bets are on the games (there exist also bets for teams, like who is going to win the championship etc.). It is well-known that one of the best teams in Europe and Worldwide are Real Madrid CF, FC Barcelona, Manchester United, Juventus etc. So, someone betting on them to win would be a reasonable case. Well, let's

look at the data.

We again pasted another Custom SQL Query and with the help of Tableau's stacked bars and the Median with Quartiles summarization, we got the following graph ():

```
with match_predicted as (
select case
            when a.[bet_result] = 1 and b.home_team_goal-b.away_team_goal
< 0 then b.home_team_api_id
            when a.[bet_result] = 2 and b.home_team_goal-b.away_team_goal
= 0 then b.home_team_api_id
            when a.[bet_result] = 3 and b.home_team_goal-b.away_team_goal
> 0 then b.home_team_api_id
      end as home_team, case
            when a.[bet_result] = 1 and b.home_team_goal-b.away_team_goal
< 0 then b.away_team_api_id
            when a.[bet_result] = 2 and b.home_team_goal-b.away_team_goal
= 0 then b.away_team_api_id
            when a.[bet_result] = 3 and b.home_team_goal-b.away_team_goal
> 0 then b.away_team_api_id
      end as away_team,
      b.[league_id]
from (select [match_api_id], [bet_result] from
(select a.[match_api_id], RANK() OVER (PARTITION BY [match_api_id] ORDER
BY avg_bets) as avg_rank, a.[bet_result]
from
(select [match_api_id], [bet_result], avg(value) as avg_bets
from [dbo].[Bets_Fact]
group by [match_api_id], [bet_result]) a) b
where avg_rank = 1) a, [dbo].[Match_Dim] b
where a.[match_api_id] = b.[match_api_id]
), match_played as (
select [home_team_api_id], sum(matches_played) as matches_played from (
select [home_team_api_id], count(*) as matches_played from
[dbo].[Match_Dim] group by [home_team_api_id]
union all
select [away_team_api_id], count(*) from [dbo].[Match_Dim] group by
[away_team_api_id]
) a group by [home_team_api_id]
)
select team, league, round(cast(matches_predicted as
float)/cast(matches_played as float),2)*100 as percent_predicted from (
select b.team_api_id, b.team_long_name as team, c.name as league,
count(*) as matches_predicted from (
select home_team, league_id from match_predicted
where home_team is not null
union all
select away_team, league_id from match_predicted
where away_team is not null) a, [dbo].[Team_Dim] b, [dbo].[League_Dim] c
where a.home_team = b.team_api_id and c.id = a.league_id
group by b.team_api_id, b.team_long_name, c.name) b, match_played d
where d.[home_team_api_id] = b.team_api_id
```

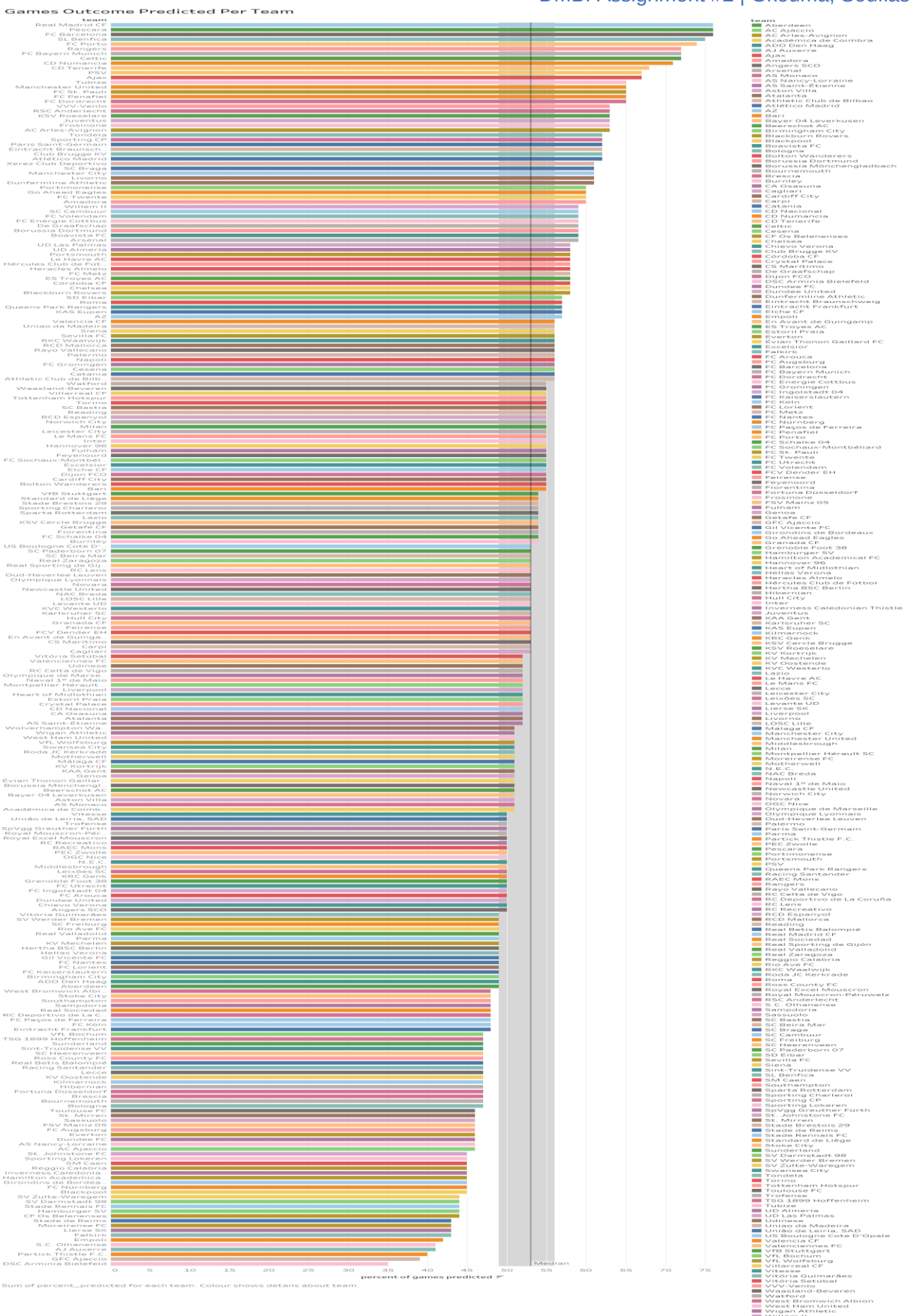


Figure 13 - Games Outcome Predicted per Team

From the above graph we can securely mention the teams that are more likely to have a predicted outcome are those that exceed the median and upper quartile and the teams that are least likely to get a correct prediction are those that lie below the lower quartile. We should mention that of course we found the best Teams on the top, like Real Madrid and Barcelona holding an enormous amount of 76 percent of outcome predicted that means that every four matched, they lose only one prediction. But someone could find out that on the top of predictions are teams like Pescara, Numancia and Tenerife. Those teams get also high amount of predictions, but not for the preferred for them outcome (losing).

Now that we have the most or less predictable leagues and teams, someone would ask about the goals. What is the most likelihood number of goals in a match. Is it biased towards the day the match is being held? To create this visualization and derive conclusions from it, we used a Custom SQL Query, a scatter plot and median with quartiles for both variables (amount of games played, average goals). We should mention that if a match ends with two or less goals the match is considered to be an "Under", whereas if a match ends with three or more goals then it is considered as an "Over". The SQL Query used is written below and it produces Figure 14:

```
select c.name as league, b.weekday, count([match_api_id]) as
games_played, round(cast(sum([home_team_goal] + [away_team_goal]) as
float) / cast(count([match_api_id]) as float), 2) as avg_goals
from [dbo].[Match_Dim] a, [dbo].[Date_Dim] b, [dbo].[League_Dim] c
where a.date_id = b.id and a.[league_id] = c.id
group by c.name, b.weekday
```

From the Figure produced we can derive some important conclusions. Firstly, the x axis is the line of trust, where above the upper quartile means that we can rely on the results as they belong to the top 25% of the distribution. There exists a match on the German Bundesliga that the average goals scored are 8. Of course, we could not say that if another match on German Bundesliga was being held on Monday, it would end with 8 goals, because the number of matches played on that occasion is one and only. The median is 2.69 goals, the upper quartile is 2.9 goals and lower quartile is 2.5. So, we can see that most of our data points lie between the 2.5 and 2.9. This explains why the limit that the bet changes is the numbers two and three. Of course, we can denote a tendency to be on the top limit (it takes values from 2.5 to 2.9 most of the cases). So, in the Leagues that we study is better to bet on the "Over" than on the "Under". Of course, this is not true for other leagues. Different data, different outcomes and explanations. We are studying the top leagues of Europe, thus being natural the teams to have good goal-scorers.

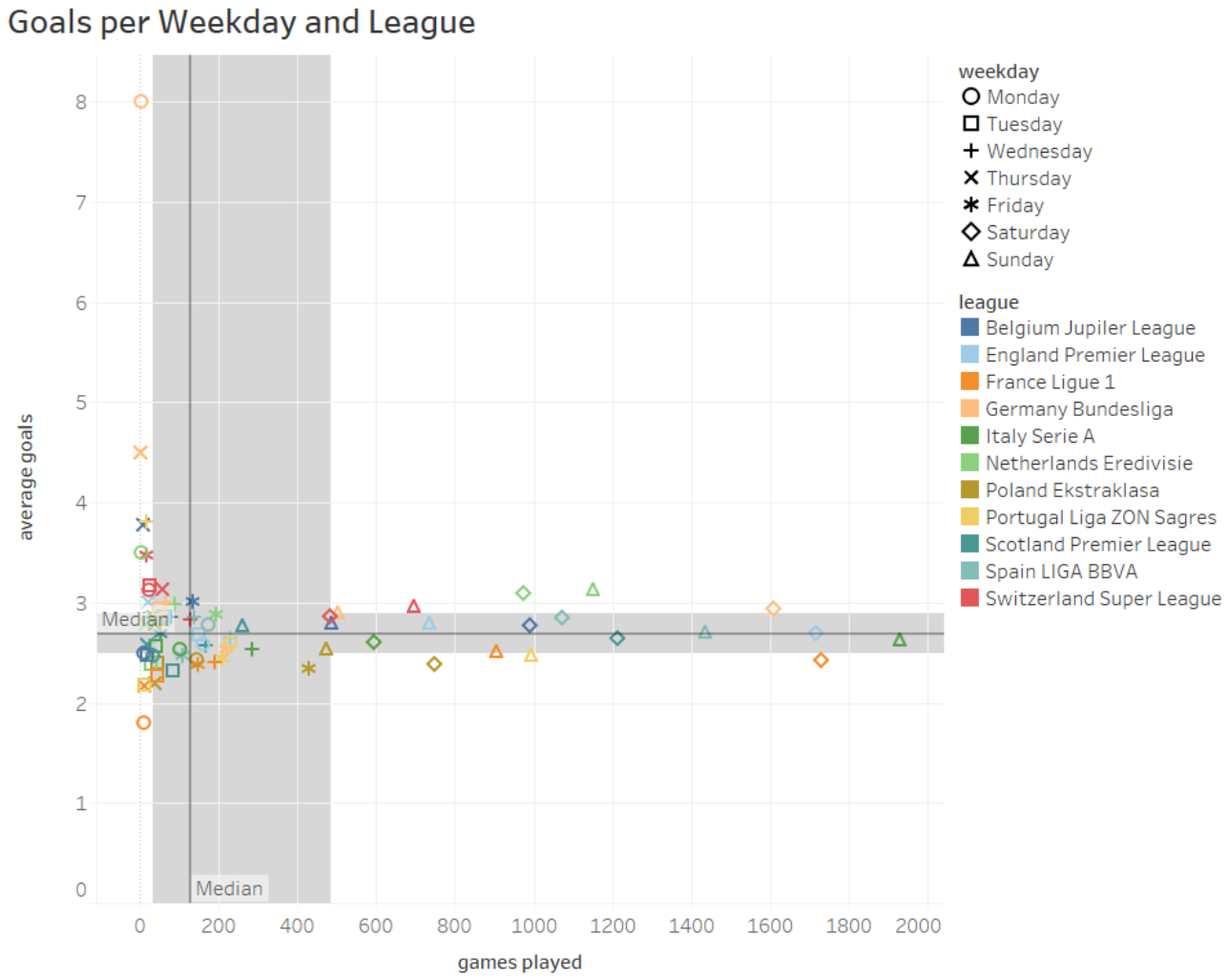
Examining Figure 14, we can extract the following important conclusions:

- Polish Ekstraklasa matches played on Saturday tend to have an "Under" conclusion.
- Portuguese Liga ZON Sagres matches played on Sunday tend to have an "Under" conclusion.
- French League 1 Saturday matches tend to have also an "Under" conclusion.
- German Bundesliga Sunday matches tend to lead to an "Over" conclusion.
- Swiss Super League Sunday matches tend to lead to an "Over" conclusion.
- German Bundesliga Saturday matches tend to lead to an "Over" conclusion.

The matches that we can securely derive that on average they end as "Over" are:

- The Dutch Eredivisie Saturday matches with an average of 3.1.
- The Dutch Eredivisie Sunday matches with an average of 3.13.

To conclude, we are again going to mention that we discourage people from betting. In many cases, people have become addictive, and it is then difficult to get rid from the addiction, with several catastrophic, not only economic but also psychological, effects for themselves and their families. In case people are not following the advice given, at least they should take into account the results of this case study, in order to eliminate huge losses of their fotune.



Sum of games_played vs. sum of avg_goals. Colour shows details about league (Custom SQL Query3). Shape shows details about weekday.

Figure 14 - Goals per Weekday and League

References

FIFAUTeam. (2021, Nov 22). *Work Rates Guide*. Retreived from https://fifauteam.com/: https://fifauteam.com/work-rates-fifa-22/

GlobalData Plc. (2022). *market-size-of-sports-betting-in-the-europe-region*. Retreived from https://www.globaldata.com/.

Symphony Solutions. (2019). *coral-case-study*. Retreived from symphony-solutions.eu: https://www.symphony-solutions.eu/coral-case-study/