Statistical Analysis Top 5 Leagues

November 19, 2024

1 Top 5 Leagues Historical Goals: Statistical Insight

1.1 Top 5 Leagues Historical Goals: France recent increase in goals

1.1.1 Import relevant libraries

[327]:
```
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
import seaborn as sns

from scipy import stats
from scipy.stats import linregress
from scipy.stats import norm
```

1.1.2 Load the CSV file with the data

[328]:
```
# Read Top 5 Leagues 25 Countries CSV file
df = pd.read_csv(r"C:\Users\jprey\OneDrive\Escritorio\JP\KUL\5th Semester␣
↪(Polimi)\MIT IDSS\Practice Projects\Top 5␣
↪Leagues\top_5_leagues_25_countries_cumulative.csv")
df.head()
```

[328]:
```
  Countries  1963-1964  1964-1965  1965-1966  1966-1967  1967-1968  1968-1969  \
0   Germany        830        760        953        845        913        776
1   England       1132       1061        953        895        982        890
2    France        721        701        967        801        771        585
3     Spain        495        495        499        578        576        491
4     Italy        373        457        466        468        383        415

   1969-1970  1970-1971  1971-1972  …  2015-2016  2016-2017  2017-2018  \
0        881        869        940  …        360        360        390
1        836        736        783  …        297        291        275
2        699        759        788  …        552        603        515
3        516        474        596  …        573        647        584
4        396        423        427  …        439        472        431
```

```
   2018-2019  2019-2020  2020-2021  2021-2022  2022-2023  2023-2024    sum
0        384        364        351        365        419        440  39164
1        284        366        372        358        387        459  37995
2        614        493        614        700        681        568  37713
3        608        600        613        618        503        546  34409
4        417        424        374        392        310        321  27814

[5 rows x 63 columns]
```

[329]:

```python
# Create a sub-dataframe that includes only the 5 nations with the most overall␣
↪goals between 1994-2004
thirty_years_ago = df.iloc[:5, [0,32,33,34,35,36,37,38,39,40,41]]
# Add a column with its mean, use np.floor to avoid fractional goals.
thirty_years_ago['Mean'] = np.floor(thirty_years_ago.iloc[:, 1:].mean(axis=1))
# Make it an int.
thirty_years_ago["Mean"] = thirty_years_ago["Mean"].astype(int)

# Create a sub-dataframe that includes only the 5 nations with the most overall␣
↪goals between 2004-2014
previous_ten_years = df.iloc[:5, [0,42,43,44,45,46,47,48,49,50,51]]
# Add a column with its mean, use np.floor to avoid fractional goals.
previous_ten_years['Mean'] = np.floor(previous_ten_years.iloc[:, 1:].
↪mean(axis=1))
# Make it an int.
previous_ten_years["Mean"] = previous_ten_years["Mean"].astype(int)

# Create a sub-dataframe that includes only the 5 nations with the most overall␣
↪goals between 2014-2024
these_ten_years = df.iloc[:5,[0,52,53,54,55,56,57,58,59,60,61]]
# Add a column with its mean, use np.floor to avoid fractional goals.
these_ten_years['Mean'] = np.floor(these_ten_years.iloc[:, 1:].mean(axis=1))
# Make it an int.
these_ten_years["Mean"] = these_ten_years["Mean"].astype(int)
```

[330]:

```python
thirty_years_ago
```

[330]:

```
  Countries  1994-1995  1995-1996  1996-1997  1997-1998  1998-1999  1999-2000  \
0   Germany        695        593        572        563        508        445
1   England        779        604        541        489        437        433
2    France        670        616        671        568        565        558
3     Spain        558        734        638        402        469        559
4     Italy        487        544        568        611        489        463

   2000-2001  2001-2002  2002-2003  2003-2004  Mean
0        403        361        269        302   471
1        422        397        361        346   480
2        525        555        577        570   587
```

```
3          617          560          564          552    565
4          503          518          538          518    523
```

[331]:
```python
previous_ten_years
```

[331]:
```
   Countries  2004-2005  2005-2006  2006-2007  2007-2008  2008-2009  2009-2010  \
0    Germany        317        327        302        292        295        312
1    England        367        355        323        327        310        342
2     France        482        438        443        519        503        457
3      Spain        513        491        457        500        587        574
4      Italy        635        626        673        664        616        569

   2010-2011  2011-2012  2012-2013  2013-2014  Mean
0        348        365        437        390   338
1        320        323        290        320   327
2        467        507        584        531   493
3        513        546        626        619   542
4        522        475        488        474   574
```

[332]:
```python
these_ten_years
```

[332]:
```
   Countries  2014-2015  2015-2016  2016-2017  2017-2018  2018-2019  2019-2020  \
0    Germany        370        360        360        390        384        364
1    England        297        297        291        275        284        366
2     France        514        552        603        515        614        493
3      Spain        556        573        647        584        608        600
4      Italy        433        439        472        431        417        424

   2020-2021  2021-2022  2022-2023  2023-2024  Mean
0        351        365        419        440   380
1        372        358        387        459   338
2        614        700        681        568   585
3        613        618        503        546   584
4        374        392        310        321   401
```

1.1.3 Extract the data relevant to France

[333]:
```python
# Extract the France values for the seasons 2004-2014 in a dataframe
france_previous_ten_years = previous_ten_years.iloc[2:3,␣
↪[1,2,3,4,5,6,7,8,9,10]].iloc[0]
# Extract the France values for the seasons 2014-2024 in a dataframe
france_these_ten_years = these_ten_years.iloc[2:3, [1,2,3,4,5,6,7,8,9,10]].
↪iloc[0]
```

1.1.4 T-test for two samples

We want to perform a t-test for two samples to check whether the mean goals of France have increased in these last 10 years compared to the previous last 10 years (2004-2014). To do so, we have to make sure that: - The samples are independent between each other and random - This is true because each year has different goals that do not depend on historical data. Moreover, the goals scored per season can be approximated to random since they depend on the players' performance year by year. - The samples are assumed to be normally distributed - A Shapiro-Wilk test will be performed to determine this. - The variances are equal. - Levene's test can be performed to determine this.

We will consider a level of significance of 0.01

[334]:
```
# Set alpha as a global variable.
alpha = 0.01
```

Assess for normality: Shapiro-Wilk test For each sample: if the p-value is higher than alpha, then the null hypothesis stating that the sample follows a normal distribution cannot be rejected, i.e., the sample is normally distributed if p-value > alpha.

[335]:
```
# Shapiro test parameters for the previous last ten years (2004-2014)
shapiro_france_previous_ten_years = stats.shapiro(france_previous_ten_years)
p_value_previous_ten_years = shapiro_france_previous_ten_years.pvalue
print("Shapiro-Wilk Test for France's goals from 2004 to 2014: W =",␣
 ↪shapiro_france_previous_ten_years.statistic, ", p-value =",␣
 ↪p_value_previous_ten_years)

# Shapiro test parameters for this last ten years (2014-2024)
shapiro_france_these_ten_years = stats.shapiro(france_these_ten_years)
p_value_these_ten_years = shapiro_france_these_ten_years.pvalue
print("Shapiro-Wilk Test for France's goals from 2014 to 2024: W =",␣
 ↪shapiro_france_these_ten_years.statistic, ", p-value =",␣
 ↪p_value_these_ten_years)

# Conclusion
print("Both samples are normally distributed?", p_value_previous_ten_years >␣
 ↪alpha and p_value_these_ten_years > alpha)
```

```
Shapiro-Wilk Test for France's goals from 2004 to 2014: W = 0.9478259459792676 ,
p-value = 0.6428542654489622
Shapiro-Wilk Test for France's goals from 2014 to 2024: W = 0.9403216198196231 ,
p-value = 0.556597672066893
Both samples are normally distributed? True
```

Therefore, the samples are normally distributed.

Assess for equal variances: Levene's test We execute the test simultaneously for both samples: if the p-value is higher than alpha, then the null hypothesis stating that the variances of the samples are equal cannot be rejected, i.e., the samples' variances are equal if p-value > alpha.

[336]:
```
# We execute Levene's test using our two datasets.
levene_test = stats.levene(france_previous_ten_years, france_these_ten_years)
p_value_levene_test = levene_test.pvalue
print("Levene's Test for both samples: Statistic =", levene_test.statistic, ",␣
↪p-value =", p_value_levene_test)

# Conclusion
print("Both samples' variances are equal?", p_value_levene_test > alpha)
```

```
Levene's Test for both samples: Statistic = 2.358587346422444 , p-value =
0.14198606565081504
Both samples' variances are equal? True
```

Therefore, the samples' variances are equal.

The samples are independent, randomly selected, uniformly distributed and have equal variances.

1.1.5 Let's perform the t-test for two samples

- **Null Hypothesis (H0)**: previous >= this (the mean of the goals scored by French players in the top 5 European Football Leagues between the seasons 2003-2024 and 2013-2024 is greater than or equal to the mean of the seasons 2013-2014 to 2023-2024)
- **Alternative Hypothesis (Ha)**: previous < this (the mean of the goals scored by French players in the top 5 European Football Leagues between the seasons 2003-2024 and 2013-2024 is less than the mean of the seasons 2013-2014 to 2023-2024)

[337]:
```
# Using scipy module stats, a statistical test is perfroemd.
t_statistic, p_value_France_Germany = stats.
↪ttest_ind(france_previous_ten_years, france_these_ten_years,␣
↪alternative="less")
print("p-value =", p_value_France_Germany)
print("Reject the null hypothesis in favor of the alternative hypothesis?
↪",p_value_France_Germany < alpha)
```

```
p-value = 0.001274605152448159
Reject the null hypothesis in favor of the alternative hypothesis? True
```

1.1.6 We are Sure with a 99% Confidence Level that the mean of goals scored of French players in Europe Top 5 Football leagues has increased in these last 10 years with resepct to the previous last 10 years (2004-2014)

1.2 Top 5 Leagues Historical Goals: Who will be at the top in the next decade

1.2.1 Let's interpret our findings regarding France.

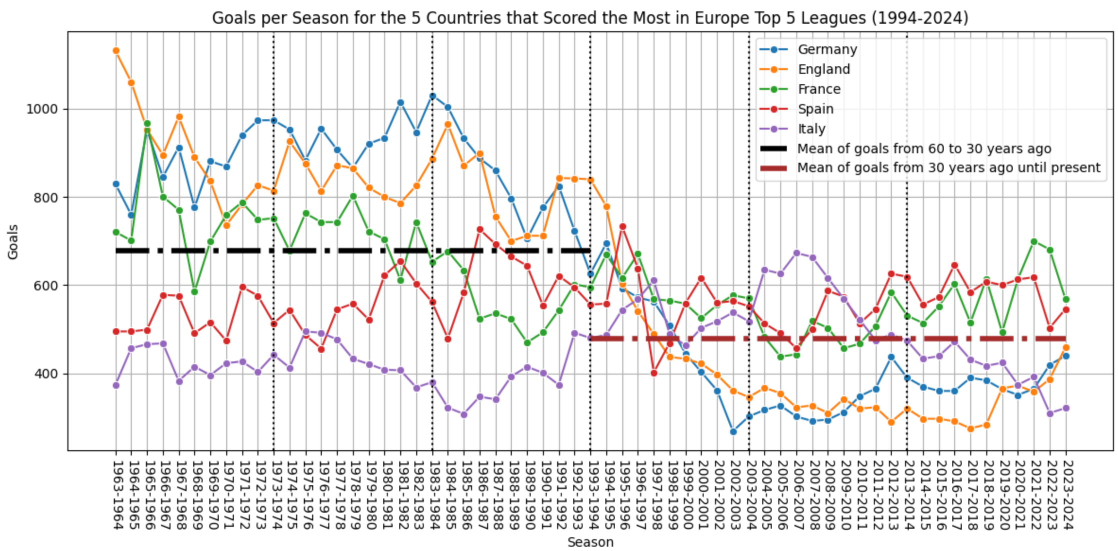
France players have score more goals this last decade with respect to the previous one… does that mean that this trend will continue?

- Although computers and AI can do a lot for the Data Science domain, there is always the need for human insight. For example, it is true that France players have scored more this last decade, but does this mean that there will be a new increase the next decade? This is why it is very important to consider as much data as possible. In fact, this is why there is a different test for distributions of more than 30 samples.
- If we had only consider the previous result, we would have assumed an increase for the next decade, or we could have thought that it is reasonable to consider the average of the lsat decade only. However, this is a bad approximation. Goals scored by players per nation naturally vary throughout time.
- Modern football (the last 30 years) has become a more defensive game, leading to less goals. Moreover, globalization of the sport has brought great players from different countries, reducing the total sum of this historically frequent scoring countries. ##### This will be shown in the next graph

[338]:
```
# Reshape the DataFrame using melt
df_melted = df.iloc[:5, :-1].melt(id_vars='Countries', var_name='Season',␣
↪value_name='Goals')

# Calculate the average for the first 30 years of the database and these last␣
↪30 years
average_60_to_30_years_ago_top_countries = df.iloc[0:5, 1:32].mean().mean()
average_30_to_present = df.iloc[0:5, 32:-1].mean().mean()
print("From 1963-1993, on average, the top 5 countries scored", int(np.
↪floor(average_60_to_30_years_ago_top_countries-average_30_to_present)),␣
↪"goals than in the past 30 years.")

# Create the line plot using Countries as hue
plt.figure(figsize=(12, 6))
sns.lineplot(data=df_melted, x='Season', y='Goals', hue='Countries', marker='o')

# Title and labels of the plot
plt.title('Goals per Season for the 5 Countries that Scored the Most in Europe␣
↪Top 5 Leagues (1994-2024)')
plt.xticks(rotation=-90)
plt.ylabel('Goals')
plt.xlabel('Season')

# Horizontal lines to show the average in their respective timeframe
plt.hlines(y=average_60_to_30_years_ago_top_countries, xmin="1963-1964",␣
↪xmax="1993-1994", label="Mean of goals from 60 to 30 years ago",␣
↪color='black', linestyle='-.', linewidth=4)
plt.hlines(y=average_30_to_present, xmin="1993-1994", xmax="2023-2024",␣
↪label="Mean of goals from 30 years ago until present", color='brown',␣
↪linestyle='-.', linewidth=4)

# Vertical dotted lines to show decade division
plt.axvline(x="1973-1974", color="black", linestyle='dotted')
```

```
plt.axvline(x="1983-1984", color="black", linestyle='dotted')
plt.axvline(x="1993-1994", color="black", linestyle='dotted')
plt.axvline(x="2003-2004", color="black", linestyle='dotted')
plt.axvline(x="2013-2014", color = "black", linestyle='dotted')

# Visualization
plt.grid()
plt.legend()
plt.tight_layout()
plt.show()
```

```
From 1963-1993, on average, the top 5 countries scored 198 goals than in the
past 30 years.
```



As shown in the graph, there is a clear difference in these last 30 years compare to the previous ones.

- Moreover, the scoring trends in the last 30 years seems to have stabilized.

Which nation will be on top by the next decade?

- It would be interesting to consider the cumulative goals in the top 5 leagues since the birth of the most recent league (Bundesliga).

[339]:
```
# Cumulative goals since 1963-1964. Table sorted by total sum
df.iloc[:, [0,62]].sort_values("sum", ascending = False).head()
```

[339]:

	Countries	sum
0	Germany	39164

```
1   England  37995
2    France  37713
3     Spain  34409
4     Italy  27814
```

Germany is on top currently, but how likely is it for them to stay on the top for another decade? Let's dive deeper into the average goals scored per decade these last years.

[340]:
```python
# Concatenate all the means per decade to visualize it more easily.
means_table = pd.concat([thirty_years_ago.iloc[:, [0,-1]], previous_ten_years.
 ↪iloc[:, [-1]], these_ten_years.iloc[:, [-1]]], axis=1)
# Create an overall mean as int
means_table["Mean Modern Football"] = np.floor(means_table.iloc[:, 1:].
 ↪mean(axis=1))
means_table["Mean Modern Football"] = means_table["Mean Modern Football"].
 ↪astype(int)
# Add the total goals sum per country
means_table = pd.concat([means_table, df.iloc[0:5, [-1]]], axis=1)
# Rename the columns
means_table.columns = ["Countries", "Mean 30-20 years ago", "Mean 20-10 years␣
 ↪ago", "Mean 10 years ago to present", "Overall Mean Past 30 years", "Total␣
 ↪goals sum"]
# Show the table
means_table.reset_index(drop=True, inplace=True)
means_table
```

[340]:
```
  Countries  Mean 30-20 years ago  Mean 20-10 years ago  \
0   Germany                   471                   338
1   England                   480                   327
2    France                   587                   493
3     Spain                   565                   542
4     Italy                   523                   574

   Mean 10 years ago to present  Overall Mean Past 30 years  Total goals sum
0                           380                         396            39164
1                           338                         381            37995
2                           585                         555            37713
3                           584                         563            34409
4                           401                         499            27814
```

Who could dethrone Germany in the next decade? Let's analyze the Overall Mean of goals per season in the past 30 years and the historical sum of goals. - England has had less goals on average this past 30 years per season than Germany. They cannot beat Germany. - Italy is more than 12000 goals away, they cannot dethrone Germany. - Spain and France have scored on average the same amount of goals per season (± 8 goals) in these last 30 years. Therefore, the closest one to Germany is the best candidate. This is France

1.2.2 France vs Germany: race for gold

Let's perform a Normal Distribution problem where we provide the population standard deviations of means for each country. We will do a Linear Combination of random independent variables: - Var(1F - 1G) = 12xVar(F) + (-1)2xVar(F) - E(1F - 1G) = 1x F - 1x G

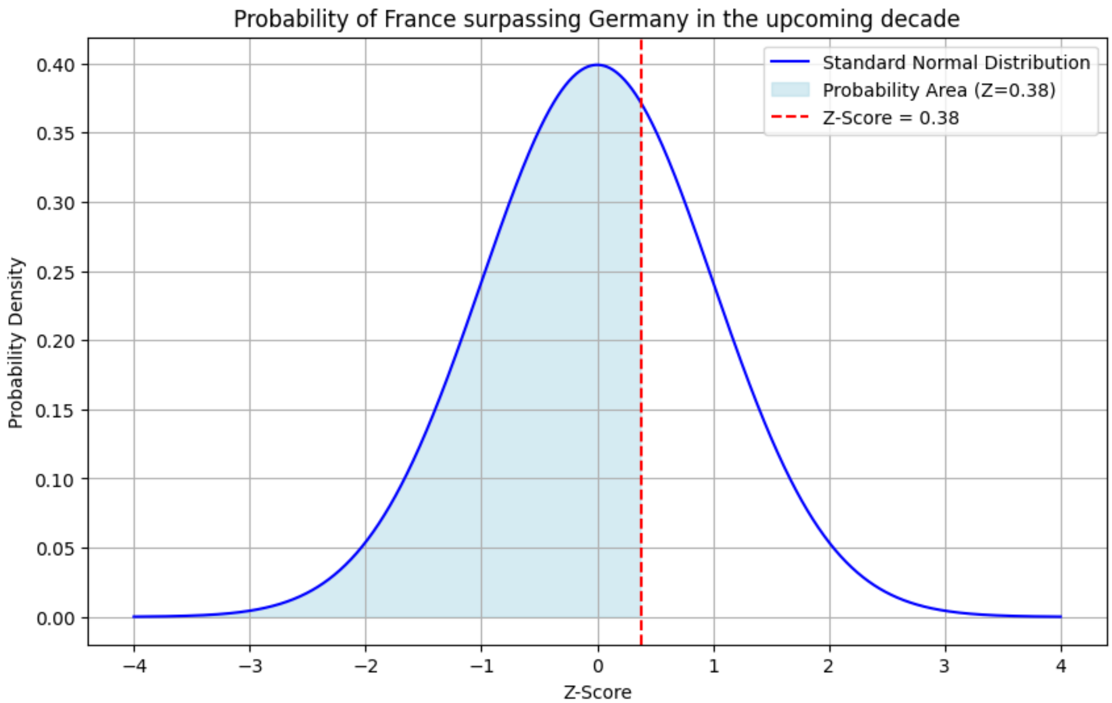
We are evaluating over 30 different seasons, so z-tests can be performed

[341]:
```
# Historical averages and population standard deviations. Also the current lead␣
↪in accumulated goals is calculated
avg_france_goals_last_30 = means_table.iloc[2,4]
avg_germany_goals_last_30 = means_table.iloc[0,4]
std_france_goals_last_30 = df.iloc[2, 32:-1].std()
std_germany_goals_last_30 = df.iloc[4, 32:-1].std()
current_lead_germany = means_table.iloc[0,5] - means_table.iloc[2,5]

# Number of seasons to project
n_seasons = 10

# Linear combination of continuous random independent variables
mean_difference = avg_france_goals_last_30 - avg_germany_goals_last_30
std_france_and_germany = np.sqrt(std_france_goals_last_30**2 +␣
↪std_germany_goals_last_30**2)

# Account for 10 years (it is NOT a linear combination, the variance gets␣
↪multiplied by 10 and not 10 squared.)
mean_diff_10_years = mean_difference * n_seasons  # Cumulative mean difference␣
↪over 10 seasons
std_diff_10_years = std_france_and_germany * np.sqrt(n_seasons)  # Cumulative␣
↪standard deviation over 10 seasons

# Calculate the Z-score for the current lead of Germany
z_score = (mean_diff_10_years - current_lead_germany) / std_diff_10_years
probability = norm.cdf(z_score)
# Calculate the probability using the standard normal distribution
p_value_france_germany = round(1 - probability,4)
print(p_value_france_germany)

# Output the result
print(f"Probability that France will surpass Germany in {n_seasons} seasons:␣
↪{probability*100}%")
print(f"Reject null hypothesis (France surpasses Germany)?",␣
↪p_value_france_germany < alpha )
```

```
0.3538
Probability that France will surpass Germany in 10 seasons: 64.62122179514941%
Reject null hypothesis (France surpasses Germany)? False
```

We cannot conclude that France will surpass Germany with a 95% confidence level. The probability that France does surpass Germany is 64.62% ### Let's visualize the findings

[342]:
```
# Create an array of x values from -4 to 4
x = np.linspace(-4, 4, 1000)
# Calculate the y values for the normal distribution
y = norm.pdf(x)

# Create the plot
plt.figure(figsize=(10, 6))
plt.plot(x, y, label='Standard Normal Distribution', color='blue')

# Shade the area under the curve to the right of the Z-score
plt.fill_between(x, y, where=(x <= z_score), color='lightblue', alpha=0.5,␣
 ↪label=f'Probability Area (Z={z_score:.2f})')

# Vertical line for the Z-score
plt.axvline(z_score, color='red', linestyle='--', label=f'Z-Score = {z_score:.
 ↪2f}')

# Add labels and title
plt.title('Probability of France surpassing Germany in the upcoming decade')
plt.xlabel('Z-Score')
plt.ylabel('Probability Density')
plt.legend()
plt.grid()

# Show the plot
plt.show()
```



1.2.3 Findings

- The whole point of this code is to show that it is easy to jump into wrong conclusions when not all the data is present.
- The increase of goals scored per season by France in the top 5 leagues in this decade with respect to the previous one is a fact (the null hypothesis was indeed rejected). However, this doesn't mean that the goals on next decade will also increase. Moreover, it doesn't even mean that the average will stay the same. It is necessary to account for as much data as possible.
- In this case, it was found that France has been rather monotonous this past 30 years in terms of goals scored per season in Europe's top 5 Football leagues.
- Furthermore, we were interested in predicting when is Germany going to be dethroned by most likely France. It was found that this is rather likely (64.62%) by the next decade. However, if the statistics follow the assumptions made, France will surpass Germany in 13 years and 7 months.