Football Dataset of Countries

Team Members

Akash(1640202)Thejus(1640268)Jithesh(1640276)

Abstract

The aim of our project is to forecast the football performance of different countries based on their performance over the past twenty seven years. We will be assessing individual players and the summation of players from each country will be taken to conclude a country's standards. The graph will extend to the next year and the country with the highest rankings will likely be on top of the FIFA.

Introduction

The database is obtained from the official FIFA website which is a reliable source. The datasets contain multiple columns of which we will select only nine columns which our team believes affects a players performance. The top three most explanatory ones are Country, Age and Ball-Control. Age is the key factor when it comes to sports because as a person is beyond a certain age they are prone to injuries and their performance is likely to decrease.The final dataset which is used for the time series analysis has two columns(year,country name).The first column consists of the year from 1993-2019 and the second one has their respective rankings.

Import the data

The below procedure imports the required python libraries to plot the data. We use the pandas library to read the rankings of the top eight countries which is in csv.

In [196]:
```
from pandas import Series
import warnings
warnings.filterwarnings("ignore")
import matplotlib.pyplot as plt
plt.style.use('fivethirtyeight')
from statsmodels.tsa.seasonal import seasonal_decompose
from statsmodels.tsa.arima_model import ARIMA
from statsmodels.tsa.seasonal import seasonal_decompose
from sklearn.metrics import mean_squared_error
```

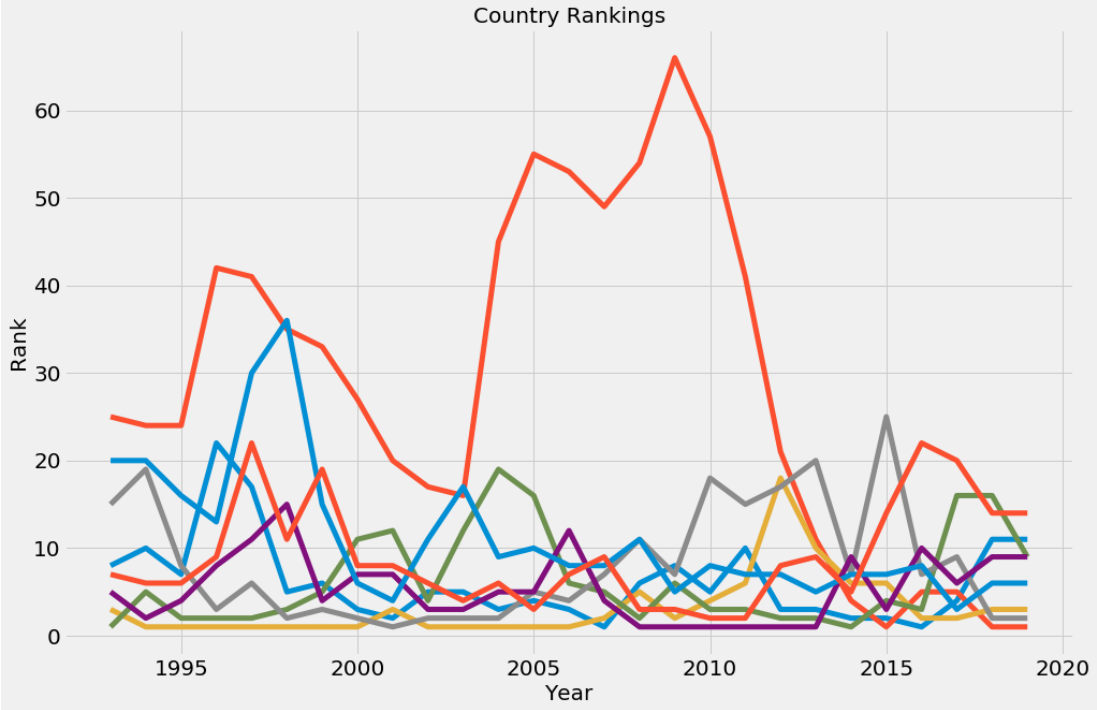
In [197]:
```
series=[]
series.append(Series.from_csv("Argentina.csv",header=0))
series.append(Series.from_csv("Belgium.csv",header=0))
series.append(Series.from_csv("Brazil.csv",header=0))
series.append(Series.from_csv("Germany.csv",header=0))
series.append(Series.from_csv("France.csv",header=0))
series.append(Series.from_csv("Spain.csv",header=0))
series.append(Series.from_csv("Portugal.csv",header=0))
series.append(Series.from_csv("Netherlands.csv",header=0))
```

Plot the data

All the countries rankings are ploted in the for loop.

In [202]:
```
for country_data in series:
    country_data.plot(linewidth=5,fontsize=20,figsize=(15,10),title='Countr
y Rankings',label='True')
plt.xlabel('Year',fontsize=20)
plt.ylabel('Rank',fontsize=20)
```

Out[202]:
```
Text(0,0.5,'Rank')
```



Decompose the data

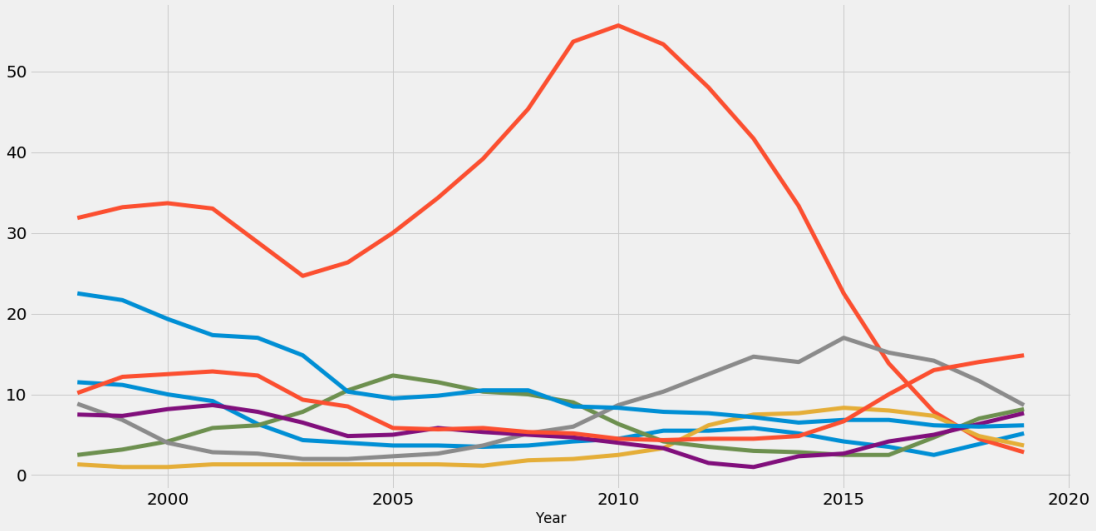
We use a library called seasonal_decompose to find out the trend,seasonal component, and residual error of the datasets.

In [70]:
```
result=[]
trend=[]
residuals=[]
observed=[]
seasonality=[]

for country_data in series:
    result.append(seasonal_decompose(country_data,model='additive'))

n=len(result)
for i in range(0,8):
    trend.append(result[i].trend)
    seasonality.append(result[i].seasonal)
    residuals.append(result[i].resid)
    observed.append(result[i].observed)
```

Country's trend

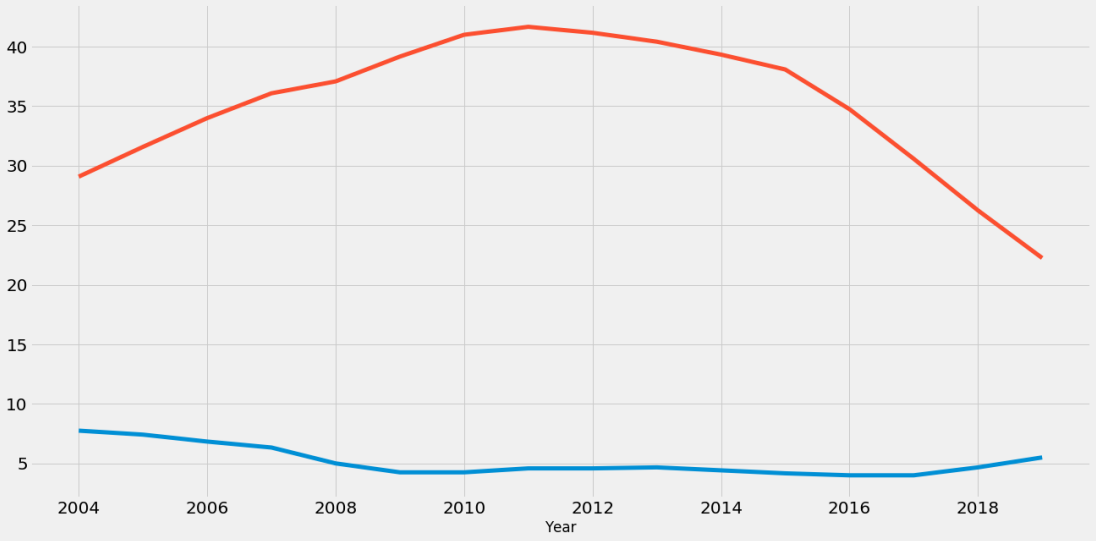
The trend shows how inconsistent a team's performance is. So from the plot we can see Belgium has the most variation while Brazil is the most constant performing country.

In [203]:
```
for i in range(0,8):
    series[i].rolling(6).mean().plot(figsize=(20,10), linewidth=5, fontsize
=20)#trend
```



In [204]:
```
series[0].rolling(12).mean().plot(figsize=(20,10), linewidth=5, fontsize=20
)
series[1].rolling(12).mean().plot(figsize=(20,10), linewidth=5, fontsize=20
)
```

Out[204]: <matplotlib.axes._subplots.AxesSubplot at 0x7f739186e470>

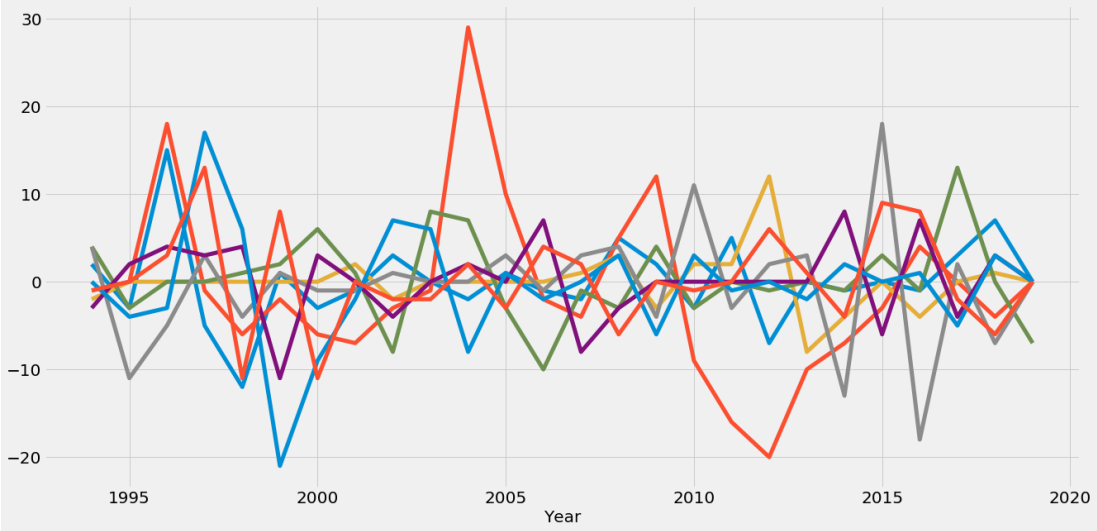


Country's Seasonality

We can see that the seasonal perfomance of some countries is a lot more than the other. Therefore the prediction will be affected based on an increase in seasonal performance.

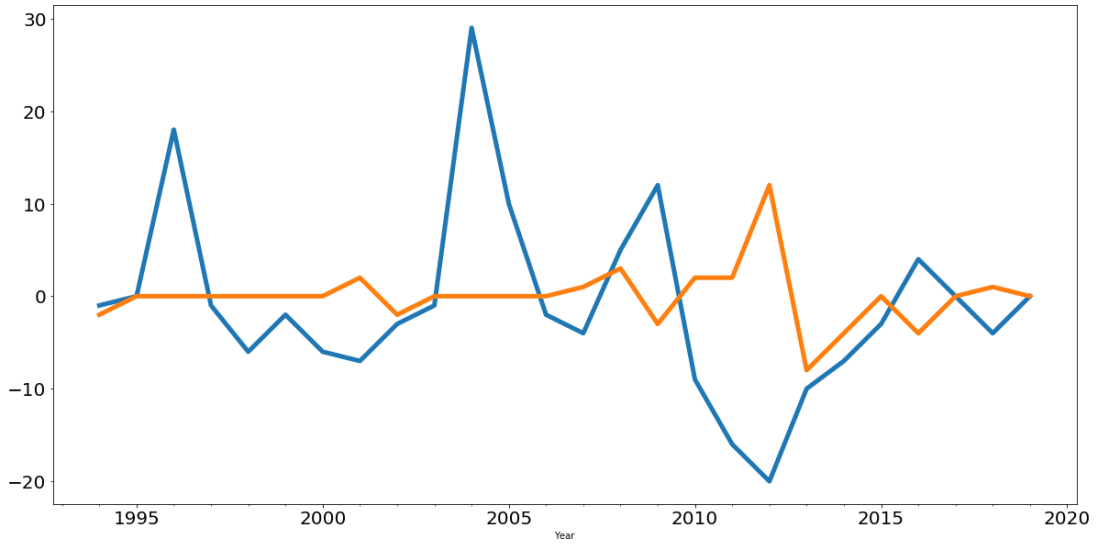
In [187]:
```
for i in range(0,8):
    series[i].diff().plot(figsize=(20,10), linewidth=5, fontsize=20)#season
ality
plt.xlabel('Year',fontsize=20)
```

Out[187]: Text(0.5,0,'Year')



In [89]:
```
series[1].diff().plot(figsize=(20,10), linewidth=5, fontsize=20)#seasonalit
y_Belgium
series[2].diff().plot(figsize=(20,10), linewidth=5, fontsize=20)#seasonalit
y_Brazil
```

Out[89]: <matplotlib.axes._subplots.AxesSubplot at 0x7f73996cca58>



Find ARIMA parameters

The below python function calculates multiple values for the ARIMA parameters(p,d,q). It returns the ones with the least root mean

square error which is calculated using libraries from scikit-learn.

In [114]:
```
def evaluate_arima_model(X, arima_order):
    # prepare training dataset
    train_size = int(len(X) * 0.66)
    train, test = X[0:train_size], X[train_size:]
    history = [x for x in train]
    # make predictions
    predictions = list()
    for t in range(len(test)):
        model = ARIMA(history, order=arima_order)
        model_fit = model.fit(disp=0)
        yhat = model_fit.forecast()[0]
        predictions.append(yhat)
        history.append(test[t])
    # calculate out of sample error
    error = mean_squared_error(test, predictions)
    return error
```

In [205]:
```
p=1
min=evaluate_arima_model(series[0],(1,0,0))
for i in range(1,5):
    if(min>evaluate_arima_model(series[0],(i,0,0))):
        p=i
        break
print(p)
```

```
1
```

In [152]:
```
evaluate_arima_model(series[0],(1,1,0))
```

Out[152]:
```
14.431384640078036
```

In [206]:
```
model=[]
model_fit=[]
model_fit2=[]
residuals=[]
forecast=[]
forecast2=[]

for i in range(0,8):
    model.append(ARIMA(series[i],order=(1,1,0)))#best-parameters

for i in range(0,8):
    model_fit.append(model[i].fit(disp=0))

for i in range(0,8):
    residuals.append(model_fit[i].resid)

for i in range(0,8):
    forecast.append(model_fit[i].forecast()[0])
```

This is a summary of the data for Argentina

In [193]:
```
model_fit[0].summary()#Argentina
```

Out[193]:

ARIMA Model Results

Dep. Variable:	D.Argentina	**No. Observations:**	26
Model:	ARIMA(1, 1, 0)	**Log Likelihood**	-77.383
Method:	css-mle	**S.D. of innovations**	4.743
Date:	Sat, 09 Mar 2019	**AIC**	160.766
Time:	11:30:33	**BIC**	164.540
Sample:	01-01-1994	**HQIC**	161.853
	- 01-01-2019		

	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
const	0.1043	0.786	0.133	0.895	-1.436	1.644
ar.L1.D.Argentina	-0.1912	0.189	-1.012	0.322	-0.562	0.179

Roots

	Real	**Imaginary**	**Modulus**	**Frequency**
AR.1	-5.2303	+0.0000j	5.2303	0.5000

Country with best ranking

Belgium will have the best ranking in 2020.

In [170]:
```
min=forecast[0]
for i in range(0,8):
    if(min>forecast[i]):
        country=series[i]
        min=forecast[i]
        val=i
```

In [166]:
```
country
```

Out[166]:
```
Year
1993-01-01    25
1994-01-01    24
1995-01-01    24
1996-01-01    42
1997-01-01    41
1998-01-01    35
1999-01-01    33
2000-01-01    27
2001-01-01    20
2002-01-01    17
2003-01-01    16
2004-01-01    45
2005-01-01    55
2006-01-01    53
2007-01-01    49
2008-01-01    54
2009-01-01    66
2010-01-01    57
2011-01-01    41
2012-01-01    21
2013-01-01    11
2014-01-01     4
2015-01-01     1
2016-01-01     5
2017-01-01     5
2018-01-01     1
2019-01-01     1
Name: Belgium, dtype: int64
```

In [194]:
```
model_fit[val].summary()
```

Out[194]:

ARIMA Model Results

Dep. Variable:	D.Belgium	**No. Observations:**	26
Model:	ARIMA(1, 1, 0)	**Log Likelihood**	-94.131
Method:	css-mle	**S.D. of innovations**	9.012
Date:	Sat, 09 Mar 2019	**AIC**	194.263
Time:	11:31:49	**BIC**	198.037
Sample:	01-01-1994	**HQIC**	195.350
	- 01-01-2019		

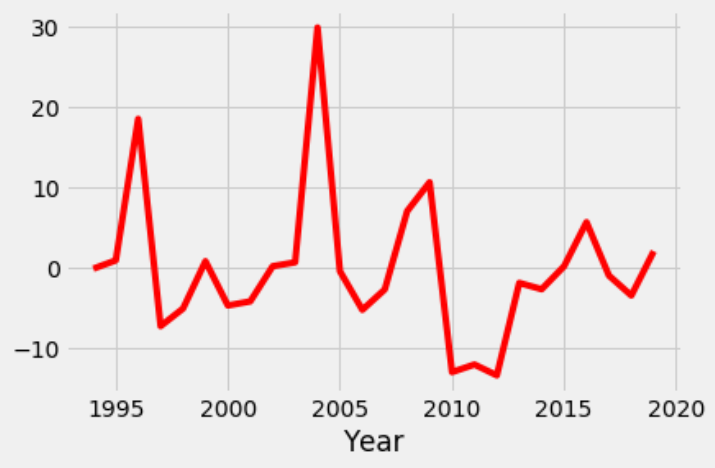
	coef	std err	z	P>|z|	[0.025	0.975]
const	-0.9041	2.779	-0.325	0.748	-6.351	4.542
ar.L1.D.Belgium	0.3784	0.176	2.149	0.042	0.033	0.723

Roots

	Real	Imaginary	Modulus	Frequency
AR.1	2.6428	+0.0000j	2.6428	0.0000

In [195]:
```
residuals[val].plot(color='r')
```

Out[195]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f7399117a90>
```



Conclusion

Therefore we can conclude that Belgium will have the best FIFA ranking in 2020 according to the ARIMA model(1,1,0).