Utilizando machine learning para predecir resultados del mundial de Rugby 2023

Radoslav Yuras

El Mundial

Francia 2023

- 8 sept - 28 oct
- 20 Equipos divididos en 4 grupos
- 40 partidos de fases de grupos,
  7 partidos de eliminación directa,
  1 por el tercer puesto
- Primer mundial al que clasifica Chile

El plan

Predicción mediante machine learning

1. Armar la base de datos
2. Preparar y limpiar los datos
3. Incluir un sistema tipo ELO basado en los rankings mundiales
4. Entrenar y testear el modelo
5. Realizar una predicción final con el modelo

Armar la base de datos

Qué datos necesitamos?

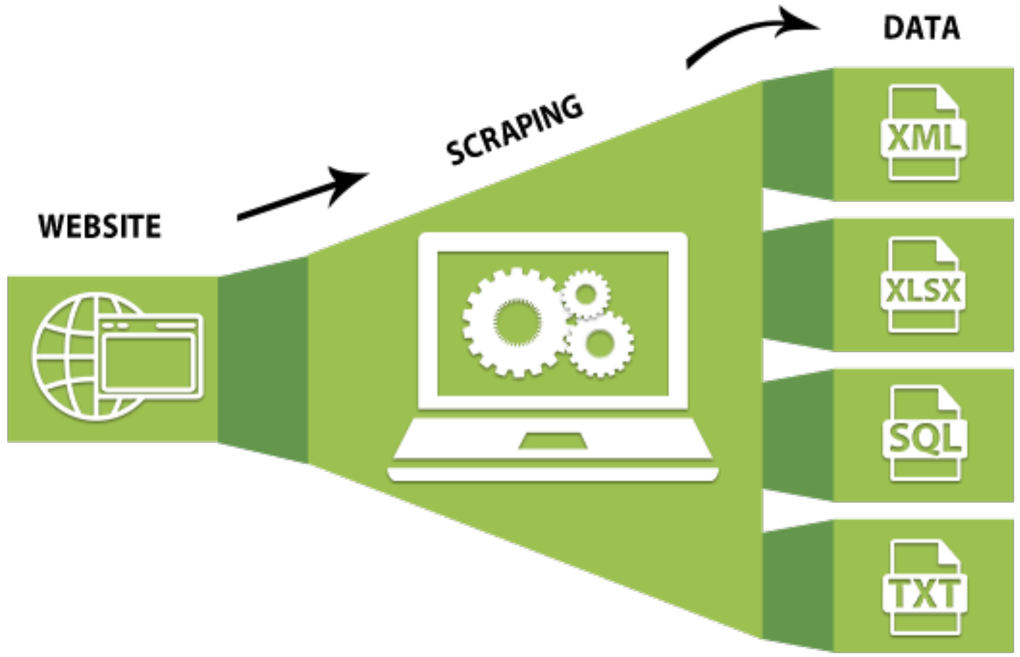
- La idea es utilizar una base de datos compuesta por todos los partidos intencionales de rugby, desde el año 2003 (1500-2000 partidos aprox.), en este periodo han ocurrido 5 mundiales y además es el año en el que se comenzó a implementar el ranking mundial, lo que usaremos más adelante

- De cada partido necesitamos por lo menos a los equipos participantes, fecha, y resultado

Armar la base de datos

Web Scraping

- Lamentablemente no existe una base de datos accesible ya completa, por lo que debemos recurrir al web scraping
- Para esto podemos recurrir a librerías como BeautifulSoup, que nos permiten extraer datos desde páginas HTML



Armar la base de datos

Fuente de datos

- http://www.lassen.co.nz/pickandgo.php

ALL -vs- ALL (01/01/2003 to 31/12/2023)

bottom

Date	Tourn	Rnd	Match	Score	Tries	Pnts	Venue	Neut.
Sat, 15 Feb 2003	6N	1	ITA v WAL	30-22	3:3	4-0	Stadio Flaminio, Rome	
Sat, 15 Feb 2003	6N	1	ENG v FRA	25-17	1:3	4-0	Twickenham, London	
Sun, 16 Feb 2003	6N	1	SCO v IRE	6-36	0:3	0-4	Murrayfield, Edinburgh	
Sat, 22 Feb 2003	6N	2	ITA v IRE	13-37	1:5	0-5	Stadio Flaminio, Rome	
Sat, 22 Feb 2003	6N	2	WAL v ENG	9-26	0:2	0-4	Millennium Stadium, Cardiff	
Sun, 23 Feb 2003	6N	2	FRA v SCO	38-3	4:0	5-0	Stade de France, Paris	
Sat, 08 Mar 2003	6N	3	IRE v FRA	15-12	0:0	4-1	Lansdowne Rd, Dublin	
Sat, 08 Mar 2003	6N	3	SCO v WAL	30-22	3:3	4-0	Murrayfield, Edinburgh	

Armar la base de datos

Código

```python
import csv
from datetime import datetime
from bs4 import BeautifulSoup
from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.common.by import By
from selenium.webdriver.chrome.options import Options


# Set path to chromedriver executable
chrome_driver_path = '/usr/local/bin/chromedriver'

# Create a Chrome WebDriver instance
chrome_options = Options()
chrome_options.add_argument('--headless')  # Run Chrome in headless mode
driver = webdriver.Chrome(service=Service(chrome_driver_path), options=chrome_options)

# Navigate to the website
driver.get('http://www.lassen.co.nz/pickandgo.php')

# Find the form fields and submit button
fyear_input = driver.find_element(By.NAME, 'txtfyear')
tyear_input = driver.find_element(By.NAME, 'txttyear')
submit_button = driver.find_element(By.NAME, 'Submit')

# Enter the desired values in the form fields
fyear_input.send_keys('2003')
tyear_input.send_keys('2023')

# Submit the form
submit_button.click()

# Wait for the page to load (add appropriate wait time if necessary)
driver.implicitly_wait(10)

# Get the resulting page source
page_source = driver.page_source

# Create a BeautifulSoup object to parse the page source
soup = BeautifulSoup(page_source, 'html.parser')
```

```python
# Find all table rows excluding the header row
table = soup.find("table", attrs={"border": "0", "cellpadding": "2", "style": "border: 1px solid #000;"})

# Create a list to store the extracted data
data = []

# Process each row
for row in table.find_all("tr", bgcolor=lambda value: value != "#FFFFCC"):
    # Extract the required data using BeautifulSoup methods
    columns = row.find_all("td")

    # Extract the date and match information
    date = columns[0].text.strip()
    tournament = columns[1].text.strip()
    round_num = columns[2].text.strip()

    # Date to datetime object
    date = datetime.strptime(date, "%a, %d %b %Y").date()

    # Split the match into home and away teams
    match = columns[3].text.strip()
    home_team, away_team = match.split(" v ")

    # Extract the score, tries, and points
    score = columns[4].text.strip()
    tries = columns[5].text.strip()
    points = columns[6].text.strip()

    # Split the score into home and away values
    home_score, away_score = score.split("-")

    # Split the tries into home and away values
    home_tries, away_tries = tries.split(":")

    # Split the points into home and away values
    home_points, away_points = points.split("-")

    # Extract the venue and neutral information
    venue = columns[7].text.strip()
    neutral = 1 if columns[8].text.strip() == "Y" else 0
```

Armar la base de datos

Código

```python
    # Append the extracted data as a dictionary to the data list
    data.append({
        "Date": date,
        "Tournament": tournament,
        "Round": round_num,
        "Home Team": home_team,
        "Away Team": away_team,
        "Home Score": home_score,
        "Away Score": away_score,
        "Home Tries": home_tries,
        "Away Tries": away_tries,
        "Home Points": home_points,
        "Away Points": away_points,
        "Venue": venue,
        "Neutral": neutral
    })

# Define the path of the CSV file
csv_file = "rugby_data.csv"

# Write the data to the CSV file
with open(csv_file, "w", newline="") as file:
    writer = csv.DictWriter(file, fieldnames=data[0].keys())
    writer.writeheader()
    writer.writerows(data)

print("Data saved to:", csv_file)
```

```
Date,Tournament,Round,Home Team,Away Team,Home Score,Away Score,Home Tries,Away Tries,Home Points,Away Points,Venue,Neutral
2003-02-15,6N,1,ITA,WAL,30,22,3,3,4,0,"Stadio Flaminio, Rome",0
2003-02-15,6N,1,ENG,FRA,25,17,1,3,4,0,"Twickenham, London",0
2003-02-16,6N,1,SCO,IRE,6,36,0,3,0,4,"Murrayfield, Edinburgh",0
2003-02-22,6N,2,ITA,IRE,13,37,1,5,0,5,"Stadio Flaminio, Rome",0
2003-02-22,6N,2,WAL,ENG,9,26,0,2,0,4,"Millennium Stadium, Cardiff",0
2003-02-23,6N,2,FRA,SCO,38,3,4,0,5,0,"Stade de France, Paris",0
2003-03-08,6N,3,IRE,FRA,15,12,0,0,4,1,"Lansdowne Rd, Dublin",0
2003-03-08,6N,3,SCO,WAL,30,22,3,3,4,0,"Murrayfield, Edinburgh",0
2003-03-09,6N,3,ENG,ITA,40,5,6,1,5,0,"Twickenham, London",0
2003-03-22,"6N,CA",4,ENG,SCO,40,9,4,0,5,0,"Twickenham, London",0
2003-03-22,6N,4,WAL,IRE,24,25,3,2,1,4,"Millennium Stadium, Cardiff",0
2003-03-23,6N,4,ITA,FRA,27,53,4,7,1,5,"Stadio Flaminio, Rome",0
2003-03-29,6N,5,SCO,ITA,33,25,4,3,5,0,"Murrayfield, Edinburgh",0
2003-03-29,6N,5,FRA,WAL,33,5,3,1,4,0,"Stade de France, Paris",0
2003-03-30,6N,5,IRE,ENG,6,42,0,5,0,5,"Lansdowne Rd, Dublin",0
2003-06-07,ST,,SAF,SCO,29,25,2,3,4,1,"Kings Park Stadium, Durban",0
2003-06-07,ST,,AUS,IRE,45,16,6,1,5,0,"Subiaco Oval, Perth_AUS",0
2003-06-14,ST,,ARG,FRA,10,6,1,0,4,1,"Velez Sarsfield, Buenos Aires",0
2003-06-14,ST,,SAF,SCO,28,19,1,1,4,0,"Ellis Park, Johannesburg",0
2003-06-14,ST,,AUS,WAL,30,10,5,1,5,0,"Stadium Australia, Sydney",0
2003-06-14,ST,,NZL,ENG,13,15,1,0,1,4,"Westpac Trust, Wellington",0
2003-06-14,ST,,TON,IRE,19,40,2,6,0,5,Nuku A'lofa,0
2003-06-20,ST,,ARG,FRA,33,32,2,2,4,1,"Velez Sarsfield, Buenos Aires",0
2003-06-20,ST,,SAM,IRE,14,40,2,3,0,4,Apia,0
2003-06-21,"ST,CC",e,AUS,ENG,14,25,1,3,0,4,"Colonial Stadium, Melbourne",0
2003-06-21,ST,,NZL,WAL,55,3,8,0,5,0,"Waikato Stadium, Hamilton",0
2003-06-28,"ST,DG",,NZL,FRA,31,23,3,2,4,0,"Jade Stadium, Christchurch",0
```

Rankings y ELO

Datos adicionales

- A cada partido quiero también asociarle el ranking mundial en el que estaba cada equipo en esa fecha

- Además utilizando esto quiero utilizar esto para crear mi propio sistema estilo ELO, que vaya aprendiendo dependiendo de los resultados y rankings, para obtener un orden de nivel de cada equipo aún más real

Rankings y ELO

Código

```python
import requests
import csv
from bs4 import BeautifulSoup

# Send a GET request to the URL
url = 'https://commons.wikimedia.org/wiki/Data:Men%27s_World_Rugby_rankings.tab'
response = requests.get(url)

# Parse the HTML content using BeautifulSoup
soup = BeautifulSoup(response.content, 'html.parser')

# Find the table element
table = soup.find('table', class_='mw-tabular')

# Extract table headers
headers = [th.get_text(strip=True) for th in table.select('thead th')]
headers = headers[-31:]

# Extract table rows
rows = []
for tr in table.select('tbody tr'):
    row = [td.get_text(strip=True) for td in tr.select('td')]
    rows.append(row)

new_headers = ['Date','NZL','AUS','SAF','FRA','ENG','IRE','WAL','SCO','ITA','SAM','FIJ','TON','ARG','USA',

# Save data to CSV file
with open('rugby_rankings.csv', 'w', newline='') as csvfile:
    writer = csv.writer(csvfile)
    writer.writerow(new_headers)
    writer.writerows(rows)

print('CSV file saved successfully.')
```

```
Date,NZL,AUS,SAF,FRA,ENG,IRE,WAL,SCO,ITA,SAM,FIJ,TON,ARG,USA,CAN,ROM,GEO,JAP,URU,POR,NAM,RUS,SPA,POL,CHL,HKG,BRA,BEL,NED,SWI
2003-10-06,2,4,6,5,1,3,10,9,13,8,11,12,7,14,16,15,17,18,19,20,25,23,30,32,24,28,35,53,40,38
2003-10-13,2,3,6,5,1,4,8,10,13,9,11,12,7,14,16,15,17,18,19,20,25,23,30,32,24,28,34,54,41,40
2003-10-20,2,3,6,5,1,4,8,10,12,9,11,15,7,14,16,13,17,18,19,20,25,23,30,32,24,28,34,53,40,39
2003-10-27,2,4,6,5,1,3,8,10,12,9,11,14,7,15,16,13,17,18,19,20,25,23,30,33,24,28,34,53,40,37
2003-11-03,2,3,6,4,1,5,8,9,11,10,12,17,7,15,13,14,19,20,16,18,25,23,30,33,24,28,34,53,40,37
2003-11-10,1,4,5,3,2,6,8,9,11,10,12,17,7,15,13,14,19,20,16,18,25,23,30,33,24,28,34,45,40,37
2003-11-17,3,2,5,4,1,6,8,9,11,10,12,17,7,15,13,14,19,20,16,18,25,23,30,33,24,28,34,45,41,37
2003-11-24,2,3,5,4,1,6,8,9,11,10,12,17,7,15,13,14,19,20,16,18,25,23,30,33,24,28,34,45,41,37
2003-12-01,2,3,5,4,1,6,8,9,11,10,12,17,7,15,13,14,19,20,16,18,25,23,30,33,24,28,34,45,41,37
2003-12-08,2,3,5,4,1,6,8,9,11,10,12,17,7,15,13,14,19,20,16,18,25,23,30,33,24,28,34,45,41,37
2003-12-15,2,3,5,4,1,6,8,9,11,10,12,17,7,15,13,14,19,20,16,18,25,23,30,33,24,28,34,45,41,37
2003-12-22,2,3,5,4,1,6,8,9,11,10,12,17,7,15,13,14,19,20,16,18,25,23,30,33,24,28,34,45,41,37
2003-12-29,2,3,5,4,1,6,8,9,11,10,12,17,7,15,13,14,19,20,16,18,25,23,30,33,24,28,34,45,41,37
2004-01-05,2,3,5,4,1,6,8,9,11,10,12,17,7,15,13,14,19,20,16,18,25,23,30,33,24,28,34,45,41,37
2004-01-12,2,3,5,4,1,6,8,9,11,10,12,17,7,15,13,14,19,20,16,18,25,23,30,33,24,28,34,45,41,37
2004-01-19,2,3,5,4,1,6,8,9,11,10,12,17,7,15,13,14,19,20,16,18,25,23,30,33,24,28,34,45,41,37
2004-01-26,2,3,5,4,1,6,8,9,11,10,12,17,7,15,13,14,19,20,16,18,25,23,30,33,24,28,34,45,41,37
2004-02-02,2,3,5,4,1,6,8,9,11,10,12,17,7,15,13,14,19,20,16,18,25,23,30,33,24,28,34,45,41,37
2004-02-09,2,3,5,4,1,6,8,9,11,10,12,17,7,15,13,14,19,20,16,18,25,23,30,33,24,28,34,45,41,37
2004-02-16,2,3,5,4,1,6,8,9,11,10,12,18,7,15,13,14,23,19,16,17,25,22,30,33,24,28,34,45,41,37
2004-02-23,2,3,5,4,1,6,8,9,11,10,12,17,7,15,13,14,23,19,16,18,25,22,30,33,24,28,34,45,41,37
2004-03-01,2,3,5,4,1,6,8,9,11,10,12,17,7,15,13,14,23,20,16,18,26,22,30,33,24,28,34,45,41,37
2004-03-08,2,3,6,4,1,5,8,9,11,10,12,18,7,15,13,14,22,20,16,17,26,24,30,33,23,28,34,45,41,37
2004-03-15,2,3,6,4,1,5,8,9,11,10,12,18,7,15,13,14,22,20,16,17,26,24,30,33,23,28,34,45,41,37
2004-03-22,2,3,6,4,1,5,8,9,11,10,12,18,7,14,13,15,22,20,16,17,26,21,31,33,24,28,34,49,39,37
2004-03-29,2,3,6,4,1,5,8,9,11,10,12,18,7,14,13,15,21,20,16,17,26,22,31,33,24,28,35,54,37,34
2004-04-05,2,3,6,4,1,5,8,9,11,10,12,18,7,14,13,15,21,20,16,17,26,22,31,33,24,28,35,54,37,34
2004-04-12,2,3,6,4,1,5,8,9,11,10,12,18,7,14,13,15,21,20,16,17,26,22,30,33,24,28,36,55,34,35
2004-04-19,2,3,6,4,1,5,8,9,11,10,12,18,7,14,13,15,21,20,16,17,26,22,30,33,24,28,36,55,34,35
2004-04-26,2,3,6,4,1,5,8,9,11,10,12,18,7,14,13,15,21,20,16,17,26,22,31,36,24,28,35,51,32,34
2004-05-03,2,3,6,4,1,5,8,9,11,10,12,18,7,14,13,15,21,20,16,17,26,22,31,36,24,28,35,51,32,34
2004-05-10,2,3,6,4,1,5,8,9,11,10,12,18,7,14,13,15,21,19,16,17,26,22,32,36,24,28,35,50,30,34
2004-05-17,2,3,6,4,1,5,8,9,11,10,12,18,7,14,13,15,22,19,16,17,26,23,32,36,25,28,35,50,31,34
```

Rankings y ELO

Código

```python
import csv
from datetime import datetime

K_FACTOR = 32  # The constant K factor determines how much the ratings change after each match

# Read rugby_data.csv
with open('rugby_data.csv', 'r') as data_file:
    data_reader = csv.DictReader(data_file)
    data = list(data_reader)

# Read rugby_rankings.csv to obtain initial Elo ratings
with open('rugby_rankings.csv', 'r') as rankings_file:
    rankings_reader = csv.DictReader(rankings_file)
    rankings = list(rankings_reader)

# Initialize team ratings based on the closest world rugby ranking
team_ratings = {}  # Dictionary to store team ratings

for row in data:
    match_date = datetime.strptime(row['Date'], '%Y-%m-%d')

    closest_ranking = None
    closest_ranking_date_diff = None

    for ranking_row in rankings:
        ranking_date = datetime.strptime(ranking_row['Date'], '%Y-%m-%d')
        date_diff = abs((ranking_date - match_date).days)

        if closest_ranking is None or date_diff < closest_ranking_date_diff:
            closest_ranking = ranking_row
            closest_ranking_date_diff = date_diff

    home_team = row['Home Team']
    away_team = row['Away Team']
```

```python
# Check if home team is in the ranking dictionary
    if home_team in closest_ranking:
        home_rank = int(closest_ranking[home_team])
    else:
        home_rank = 0  # Assign a default rank if missing in rankings

    # Check if away team is in the ranking dictionary
    if away_team in closest_ranking:
        away_rank = int(closest_ranking[away_team])
    else:
        away_rank = 0  # Assign a default rank if missing in rankings

    # Calculate initial Elo ratings based on world rugby ranking
    home_rating = 2000 - 10 * home_rank
    away_rating = 2000 - 10 * away_rank

    # Add the Elo ratings to the row
    row['Home Rank'] = home_rank
    row['Away Rank'] = away_rank
    row['Home Elo'] = home_rating
    row['Away Elo'] = away_rating


    # Update team ratings based on match result using the Elo update formula
    home_score = int(row['Home Score'])
    away_score = int(row['Away Score'])
    actual_home_score = 1 if home_score > away_score else 0
    actual_away_score = 1 if away_score > home_score else 0

    expected_home_score = 1 / (1 + 10 ** ((away_rating - home_rating) / 400))
    expected_away_score = 1 - expected_home_score

    home_new_rating = home_rating + K_FACTOR * (actual_home_score - expected_home_score)
    away_new_rating = away_rating + K_FACTOR * (actual_away_score - expected_away_score)

    # Store the updated ratings for the teams
    team_ratings[home_team] = home_new_rating
    team_ratings[away_team] = away_new_rating
```

```python
# Write the updated data to a new file
output_file = 'rugby_data_with_elo.csv'
fieldnames = data[0].keys()

with open(output_file, 'w', newline='') as outfile:
    writer = csv.DictWriter(outfile, fieldnames=fieldnames)
    writer.writeheader()
    writer.writerows(data)

print(f"The data with Elo ratings has been written to {output_file} successfully.")
```

Rankings y ELO

Database Final

```
Date,Tournament,Round,Home Team,Away Team,Home Score,Away Score,Home Tries,Away Tries,Home Points,Away Points,Venue,Neutral,Home Rank,Away Rank,Home Elo,Away Elo
2003-02-15,6N,1,ITA,WAL,30,22,3,3,4,0,"Stadio Flaminio, Rome",0,13,10,1870,1900
2003-02-15,6N,1,ENG,FRA,25,17,1,3,4,0,"Twickenham, London",0,1,5,1990,1950
2003-02-16,6N,1,SCO,IRE,6,36,0,3,0,4,"Murrayfield, Edinburgh",0,9,3,1910,1970
2003-02-22,6N,2,ITA,IRE,13,37,1,5,0,5,"Stadio Flaminio, Rome",0,13,3,1870,1970
2003-02-22,6N,2,WAL,ENG,9,26,0,2,0,4,"Millennium Stadium, Cardiff",0,10,1,1900,1990
2003-02-23,6N,2,FRA,SCO,38,3,4,0,5,0,"Stade de France, Paris",0,5,9,1950,1910
2003-03-08,6N,3,IRE,FRA,15,12,0,0,4,1,"Lansdowne Rd, Dublin",0,3,5,1970,1950
2003-03-08,6N,3,SCO,WAL,30,22,3,3,4,0,"Murrayfield, Edinburgh",0,9,10,1910,1900
2003-03-09,6N,3,ENG,ITA,40,5,6,1,5,0,"Twickenham, London",0,1,13,1990,1870
2003-03-22,"6N,CA",4,ENG,SCO,40,9,4,0,5,0,"Twickenham, London",0,1,9,1990,1910
2003-03-22,6N,4,WAL,IRE,24,25,3,2,1,4,"Millennium Stadium, Cardiff",0,10,3,1900,1970
2003-03-23,6N,4,ITA,FRA,27,53,4,7,1,5,"Stadio Flaminio, Rome",0,13,5,1870,1950
2003-03-29,6N,5,SCO,ITA,33,25,4,3,5,0,"Murrayfield, Edinburgh",0,9,13,1910,1870
2003-03-29,6N,5,FRA,WAL,33,5,3,1,4,0,"Stade de France, Paris",0,5,10,1950,1900
2003-03-30,6N,5,IRE,ENG,6,42,0,5,0,5,"Lansdowne Rd, Dublin",0,3,1,1970,1990
2003-06-07,ST,,SAF,SCO,29,25,2,3,4,1,"Kings Park Stadium, Durban",0,6,9,1940,1910
2003-06-07,ST,,AUS,IRE,45,16,6,1,5,0,"Subiaco Oval, Perth_AUS",0,4,3,1960,1970
2003-06-14,ST,,ARG,FRA,10,6,1,0,4,1,"Velez Sarsfield, Buenos Aires",0,7,5,1930,1950
2003-06-14,ST,,SAF,SCO,28,19,1,1,4,0,"Ellis Park, Johannesburg",0,6,9,1940,1910
2003-06-14,ST,,AUS,WAL,30,10,5,1,5,0,"Stadium Australia, Sydney",0,4,10,1960,1900
2003-06-14,ST,,NZL,ENG,13,15,1,0,1,4,"Westpac Trust, Wellington",0,2,1,1980,1990
2003-06-14,ST,,TON,IRE,19,40,2,6,0,5,Nuku A'lofa,0,12,3,1880,1970
2003-06-20,ST,,ARG,FRA,33,32,2,2,4,1,"Velez Sarsfield, Buenos Aires",0,7,5,1930,1950
2003-06-20,ST,,SAM,IRE,14,40,2,3,0,4,Apia,0,8,3,1920,1970
2003-06-21,"ST,CC",e,AUS,ENG,14,25,1,3,0,4,"Colonial Stadium, Melbourne",0,4,1,1960,1990
2003-06-21,ST,,NZL,WAL,55,3,8,0,5,0,"Waikato Stadium, Hamilton",0,2,10,1980,1900
2003-06-28,"ST,DG",,NZL,FRA,31,23,3,2,4,0,"Jade Stadium, Christchurch",0,2,5,1980,1950
2003-06-28,,,SAF,ARG,26,25,2,3,4,1,"Boet Erasmus, Pt Elizabeth",0,6,7,1940,1930
2003-07-12,3N,,SAF,AUS,26,22,2,3,4,1,"Newlands, Cape Town",0,6,4,1940,1960
2003-07-19,3N,,SAF,NZL,16,52,1,7,0,5,"Loftus Versfeld, Pretoria",0,6,2,1940,1980
2003-07-26,"3N,BC",a,AUS,NZL,21,50,3,7,0,5,"Stadium Australia, Sydney",0,4,2,1960,1980
2003-08-02,3N,,AUS,SAF,29,9,2,0,4,0,"Suncorp Stadium, Brisbane",0,4,6,1960,1940
2003-08-09,3N,,NZL,SAF,19,11,1,1,4,0,"Carisbrook, Dunedin",0,2,6,1980,1940
2003-08-16,WU,,IRE,WAL,35,12,5,2,5,0,"Lansdowne Rd, Dublin",0,3,10,1970,1900
2003-08-16,"3N,BC",n,NZL,AUS,21,17,2,1,4,1,"Eden Park, Auckland",0,2,4,1980,1960
2003-08-18,WU,,ARG,FIJ,49,30,7,3,5,0,Cordoba,0,7,11,1930,1890
```

El modelo

Train y Test

- Mi idea es trabajar con la librería Keras y desarrollar un modelo secuencial construyendo redes neuronales, experimentando con el número de capas y número de neuronas por capa, hasta obtener el mejor modelo
- Con el modelo buscaría reducir el error cuadrático medio

Live DEMO

Conclusiones

Sobre las predicciones

- A simple vista parecen predicciones bastante acertadas
- F por chilito
- Ojo con el Francia - Nueva Zelanda

```
Predictions for match: FRA vs NZL
Predicted Home Score: 21.335169
Predicted Away Score: 20.919676

Predictions for match: ITA vs NAM
Predicted Home Score: 38.417206
Predicted Away Score: 13.416033

Predictions for match: IRE vs ROM
Predicted Home Score: 52.404224
Predicted Away Score: 10.455403

Predictions for match: AUS vs GEO
Predicted Home Score: 30.046772
Predicted Away Score: 14.119124

Predictions for match: ENG vs ARG
Predicted Home Score: 30.760674
Predicted Away Score: 14.464723

Predictions for match: JAP vs CHL
Predicted Home Score: 56.30631
Predicted Away Score: 10.935213

Predictions for match: SAF vs SCO
Predicted Home Score: 30.836292
Predicted Away Score: 16.115639

Predictions for match: WAL vs FIJ
Predicted Home Score: 31.122255
Predicted Away Score: 13.504702
```

Conclusiones

¿Cómo mejorar?

- Agregar más variables al modelo, como por ejemplo si el partido fue amistoso o por algún torneo, cantidad de tries, etc.
- Revisar bien la base de datos, da la impresión de que faltan algunos partidos, un indicio de esto es el bajo ELO de países como Argentina y Escocia
- Probar con otro tipo de modelos

Utilizando machine learning para predecir resultados del mundial de Rugby 2023

Radoslav Yuras