```
import datetime
import os
import json
import pickle
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.impute import SimpleImputer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from sklearn.impute import SimpleImputer
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
import tensorflow as tf
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, mean_absolute_error
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
from datetime import datetime
import matplotlib.pyplot as plt
import seaborn as sns
import matplotlib.ticker as ticker
import matplotlib.ticker as plticker
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.datasets import make_classification
```

World Cup 2023 Winnier Prediction

```
world_cup = pd.read_csv('/content/World Cup 2023 Dataset.csv')
results = pd.read_csv('/content/results.csv')
```

```
worldcup_teams = ['England', ' South Africa', 'Netherlands',
            'Pakistan', 'New Zealand', 'Sri Lanka', 'Afghanistan',
            'Australia', 'Bangladesh', 'India']
df_teams_1 = results[results['Team_1'].isin(worldcup_teams)]
df_teams_2 = results[results['Team_2'].isin(worldcup_teams)]
df_teams = pd.concat((df_teams_1, df_teams_2))
df_teams.drop_duplicates()
df_teams.count()
```

```
Unnamed: 0     590
Team_1        6040
Team_2        6040
Winner        6040
Margin        5783
Ground        6040
Match Date    6040
dtype: int64
```

```
df_teams.head()
```

	Unnamed: 0	Team_1	Team_2	Winner	Margin	Ground	Match Date
0	0.0	Australia	Pakistan	Australia	92 runs	Brisbane	Jan 13, 2017
1	1.0	Australia	Pakistan	Pakistan	6 wickets	Melbourne	Jan 15, 2017
2	2.0	India	England	India	3 wickets	Pune	Jan 15, 2017
3	3.0	Australia	Pakistan	Australia	7 wickets	Perth	Jan 19, 2017
4	4.0	India	England	India	15 runs	Cuttack	Jan 19, 2017

```

df_teams_2023 = df_teams.drop(['Match Date','Margin', 'Ground'], axis=1)
df_teams_2023.head()
```

	Unnamed: 0	Team_1	Team_2	Winner
0	0.0	Australia	Pakistan	Australia

```
df_teams_2023= df_teams_2023.reset_index(drop=True)
df_teams_2023.loc[df_teams_2023.Winner == df_teams_2023.Team_1,'winning_team']=1
df_teams_2023.loc[df_teams_2023.Winner == df_teams_2023.Team_2, 'winning_team']=2
df_teams_2023 = df_teams_2023.drop(['winning_team'], axis=1)
df_teams_2023.head()
```

	Unnamed: 0	Team_1	Team_2	Winner
0	0.0	Australia	Pakistan	Australia
1	1.0	Australia	Pakistan	Pakistan
2	2.0	India	England	India
3	3.0	Australia	Pakistan	Australia
4	4.0	India	England	India

```

final = pd.get_dummies(df_teams_2023, prefix=['Team_1', 'Team_2'], columns=['Team_1', 'Team_2'])

X = final.drop(['Winner'], axis=1)
y = final["Winner"]

```

```

imputer = SimpleImputer(strategy="mean")
X = imputer.fit_transform(X)


X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=42)


logistic_model = LogisticRegression()


logistic_model.fit(X_train, y_train)

y_pred = logistic_model.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy * 100:.2f}%")

```

```
Accuracy: 54.83%
/usr/local/lib/python3.10/dist-packages/sklearn/linear_model/_logistic.py:458: ConvergenceWarning: lbfgs failed to conve
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

```


imputer = SimpleImputer(strategy='mean')


X_imputed = imputer.fit_transform(X)

X_train, X_test, y_train, y_test = train_test_split(X_imputed, y, test_size=0.30, random_state=42)

rf = RandomForestClassifier(n_estimators=100, max_depth=20, random_state=0)
rf.fit(X_train, y_train)


score = rf.score(X_train, y_train)
score2 = rf.score(X_test, y_test)

print("Training set accuracy: ", '%.3f' % (score))
print("Test set accuracy: ", '%.3f' % (score2))

```

```
Training set accuracy:  0.666
Test set accuracy:  0.614
```

```
ranking = pd.read_csv('/content/icc_rankings.csv')
fixtures = pd.read_csv('/content/fixtures.csv')
pred_set = []
```

```
ranking.head()
```

	Position	Country	Points
0	1	Pakistan	3061
1	2	Australia	3061
2	3	India	4516
3	4	England	2790
4	5	New Zealand	3057

```
fixtures.head()
```

	Round Number	Date	Location	Team_1	Team_2	Group	Result
0	1	05/10/2023	Narendra Modi Stadium, Ahmedabad	ENGLAND	NEW ZEALAND	Group A	NaN
1	1	06/10/2023	Rajiv Gandhi International Stadium, Hyderabad	PAKISTAN	NETHERLANDS	Group A	NaN
2	1	07/10/2023	HPCA Stadium, Dharamsala	BANGLADESH	AFGHANISTAN	Group A	NaN
3	1	07/10/2023	Arun Jaitley Stadium, Delhi	SOUTH AFRICA	SRI LANKA	Group A	NaN
4	1	08/10/2023	MA Chidambaram Stadium, Chennai	INDIA	AUSTRALIA	Group A	NaN

```
fixtures['Team_1'] = fixtures['Team_1'].str.title()
fixtures['Team_2'] = fixtures['Team_2'].str.title()


fixtures['first_position'] = fixtures['Team_1'].map(ranking.set_index('Country')['Position'])
fixtures['second_position'] = fixtures['Team_2'].map(ranking.set_index('Country')['Position'])

fixtures.tail()
```

	Round Number	Date	Location	Team_1	Team_2	Group	Result	fi
43	1	11/11/2023	Eden Gardens, Kolkata	England	Pakistan	Group A	NaN	
44	1	12/11/2023	M. Chinnaswamy Stadium, Bengaluru	India	Netherlands	Group A	NaN	
45	1	15/11/2023	Wankhede Stadium, Mumbai	1St Place	4Th Place	Group A	NaN	
46	1	16/11/2023	Eden Gardens, Kolkata	2Nd Place	3Rd Place	Group A	NaN	
47	1	19/11/2023	Narendra Modi Stadium, Ahmedabad	Winner Of Semi-Final 1 Men	Winner Of Semi-Final 2 Men	Group A	NaN	

```

```

```
for index, row in fixtures.iterrows():
    if row['first_position'] < row['second_position']:
        pred_set.append({'Team_1': row['Team_1'], 'Team_2': row['Team_2'], 'winning_team': None})
    else:
        pred_set.append({'Team_1': row['Team_2'], 'Team_2': row['Team_1'], 'winning_team': None})

pred_set = pd.DataFrame(pred_set)
backup_pred_set = pred_set
pred_set.head()
```

	Team_1	Team_2	winning_team
0	England	New Zealand	None
1	Pakistan	Netherlands	None

```
pred_set = pd.get_dummies(pred_set, prefix=['Team_1', 'Team_2'], columns=['Team_1', 'Team_2'])

missing_cols = set(final.columns) - set(pred_set.columns)
for c in missing_cols:
    pred_set[c] = 0
pred_set = pred_set[final.columns]


pred_set = pred_set.drop(['Winner'], axis=1)
pred_set.head()
```

	Unnamed: 0	Team_1_Afghanistan	Team_1_Australia	Team_1_Bangladesh	Team_1_Bermuda	Team_1_Canada	Team_1_East Africa	Team_1_E
0	0	0	0	0	0	0	0	
1	0	0	0	0	0	0	0	
2	0	0	0	1	0	0	0	
3	0	0	0	0	0	0	0	
4	0	0	1	0	0	0	0	

5 rows × 46 columns

```

predictions = rf.predict(pred_set)
match_winners = []

for i in range(fixtures.shape[0]):
    match_detail = f"{backup_pred_set.iloc[i, 1]} VS {backup_pred_set.iloc[i, 0]}"
    winner = backup_pred_set.iloc[i, 1] if predictions[i] == 1 else backup_pred_set.iloc[i, 0]

    match_winners.append({
        'Match': match_detail,
        'Winner': winner
    })

    print(match_detail)
    print(f"Winner: {winner}")
    print("")


```

```
Afghanistan VS South Africa
Winner: South Africa

Bangladesh VS Australia
Winner: Australia

England VS Pakistan
Winner: Pakistan

Netherlands VS India
Winner: India

1St Place VS 4Th Place
Winner: 4Th Place

2Nd Place VS 3Rd Place
Winner: 3Rd Place

Winner Of Semi-Final 1 Men VS Winner Of Semi-Final 2 Men
Winner: Winner Of Semi-Final 2 Men

/usr/local/lib/python3.10/dist-packages/sklearn/base.py:432: UserWarning: X has feature names, but RandomForestClassifie
  warnings.warn(
```

```
from collections import Counter
winner_names = [match['Winner'] for match in match_winners]

win_count = Counter(winner_names)

import pandas as pd
win_count_df = pd.DataFrame.from_dict(win_count, orient='index', columns=['Wins']).reset_index()
win_count_df.columns = ['Team', 'Wins']

win_count_df = win_count_df.sort_values(by='Wins', ascending=False)
win_count_df['Position'] = range(1, len(win_count_df) + 1)

win_count_df
```

	Team	Wins	Position
1	Pakistan	9	1
4	Australia	8	2
6	India	7	3
0	England	6	4
5	New Zealand	5	5
3	South Africa	4	6
2	Bangladesh	3	7
7	Sri Lanka	2	8
8	Afghanistan	1	9
9	4Th Place	1	10
10	3Rd Place	1	11
11	Winner Of Semi-Final 2 Men	1	12

```

semifinal_1 = (win_count_df.loc[win_count_df['Position'] == 1, 'Team'].iloc[0],
               win_count_df.loc[win_count_df['Position'] == 3, 'Team'].iloc[0])

semifinal_2 = (win_count_df.loc[win_count_df['Position'] == 2, 'Team'].iloc[0],
               win_count_df.loc[win_count_df['Position'] == 4, 'Team'].iloc[0])

semi = [semifinal_1, semifinal_2]
semi


```

```
[('Pakistan', 'India'), ('Australia', 'England')]
```

```
 def clean_and_predict(matches, ranking, final, logreg):


    positions = []
    finalist = []


    for match in matches:
```

```
        positions.append(ranking.loc[ranking['Country'] == match[0],'Position'].iloc[0])
        positions.append(ranking.loc[ranking['Country'] == match[1],'Position'].iloc[0])


    pred_set = []


    i = 0
    j = 0


    while i < len(positions):
        dict1 = {}


        if positions[i] < positions[i + 1]:
            dict1.update({'Team_1': matches[j][0], 'Team_2': matches[j][1]})
        else:
            dict1.update({'Team_1': matches[j][1], 'Team_2': matches[j][0]})


        pred_set.append(dict1)
        i += 2
        j += 1


    pred_set = pd.DataFrame(pred_set)
    backup_pred_set = pred_set


    pred_set = pd.get_dummies(pred_set, prefix=['Team_1', 'Team_2'], columns=['Team_1', 'Team_2'])


    missing_cols2 = set(final.columns) - set(pred_set.columns)
    for c in missing_cols2:
        pred_set[c] = 0
    pred_set = pred_set[final.columns]

    pred_set = pred_set.drop(['Winner'], axis=1)

    # Predict!
    predictions = logreg.predict(pred_set)
    for i in range(len(pred_set)):
        print(backup_pred_set.iloc[i, 1] + " VS " + backup_pred_set.iloc[i, 0])
        if predictions[i] == 1:
            print("Winner: " + backup_pred_set.iloc[i, 1])
        else:
            print("Winner: " + backup_pred_set.iloc[i, 0])
        print("")

semi_winners = clean_and_predict(semi, ranking, final, rf)
```

```
India VS Pakistan
Winner: Pakistan

England VS Australia
Winner: Australia

/usr/local/lib/python3.10/dist-packages/sklearn/base.py:432: UserWarning: X has feature names, but RandomForestClassifie
  warnings.warn(
```

```
finals = [('Pakistan', 'Australia')]
clean_and_predict(finals, ranking, final, rf)
```

```
Australia VS Pakistan
Winner: Pakistan

/usr/local/lib/python3.10/dist-packages/sklearn/base.py:432: UserWarning: X has feature names, but RandomForestClassifie
  warnings.warn(
```