Bowlers Ranking

May 18, 2021

1 TOPSIS Ranking for Bowlers

[1]:
```
import numpy as np                # for linear algebra
import pandas as pd               # for tabular output
from scipy.stats import rankdata # for ranking the candidates
```

[2]:
```
attributes_data = pd.read_csv('data/bowling_criteria.csv')
attributes_data
```

[2]:

	Name	Ranking	Ideally
0	SR	1	Lower
1	Econ	2	Lower
2	Avg	3	Lower
3	Wkts	4	Higher
4	Runs	5	Lower
5	Inns	6	Higher
6	TBB	7	Higher
7	4w	8	Higher
8	Mat	9	Higher

[3]:
```
benefit_attributes = set()
attributes = []
ranks = []
n = 0

for i, row in attributes_data.iterrows():
    attributes.append(row['Name'])
    ranks.append(float(row['Ranking']))
    n += 1

    if row['Ideally'] == 'Higher':
        benefit_attributes.add(i)

ranks = np.array(ranks)
```

[4]:
```
weights = 2 * (n + 1 - ranks) / (n * (n + 1))
weights
```

[4]:
```
array([0.2       , 0.17777778, 0.15555556, 0.13333333, 0.11111111,
       0.08888889, 0.06666667, 0.04444444, 0.02222222])
```

[5]:
```
original_dataframe = pd.read_csv('data/bowlers.csv')
candidates = original_dataframe['Name'].to_numpy()
raw_data = pd.DataFrame(original_dataframe, columns=attributes).to_numpy()

dimensions = raw_data.shape
m = dimensions[0]
n = dimensions[1]

pd.DataFrame(data=raw_data, index=candidates, columns=attributes)
```

[5]:

	SR	Econ	Avg	Wkts	Runs	Inns	TBB	4w	Mat
Andre Russell	16.45	9.51	26.09	11.0	287.0	12.0	181.0	0.0	14.0
Ben Stokes	16.83	11.23	31.50	6.0	189.0	6.0	101.0	0.0	9.0
Chris Morris	15.23	9.27	23.54	13.0	306.0	9.0	198.0	0.0	9.0
Dwayne Bravo	22.45	8.02	30.00	11.0	330.0	12.0	247.0	0.0	12.0
Imran Tahir	14.85	6.70	16.58	26.0	431.0	17.0	386.0	2.0	17.0
Jofra Archer	23.45	6.77	26.45	11.0	291.0	11.0	258.0	0.0	11.0
Kagiso Rabada	11.28	7.83	14.72	25.0	368.0	12.0	282.0	2.0	12.0
Keemo Paul	18.11	8.72	26.33	9.0	237.0	8.0	163.0	0.0	8.0
Lasith Malinga	16.81	9.77	27.38	16.0	438.0	12.0	269.0	2.0	12.0
Moeen Ali	25.00	6.76	28.17	6.0	169.0	9.0	150.0	0.0	11.0
Mohammad Nabi	21.88	6.65	24.25	8.0	194.0	8.0	175.0	1.0	8.0
Rashid Khan	21.18	6.28	22.18	17.0	377.0	15.0	360.0	0.0	15.0
Sam Curran	19.80	9.79	32.30	10.0	323.0	9.0	198.0	1.0	9.0
Sunil Narine	26.60	7.83	34.70	10.0	347.0	12.0	266.0	0.0	12.0
Trent Boult	22.80	8.58	32.60	5.0	163.0	5.0	114.0	0.0	5.0

[6]:
```
divisors = np.empty(n)
for j in range(n):
    column = raw_data[:,j]
    divisors[j] = np.sqrt(column @ column)

raw_data /= divisors
pd.DataFrame(data=raw_data, index=candidates, columns=attributes)
```

[6]:

	SR	Econ	Avg	Wkts	Runs	Inns	\
Andre Russell	0.212905	0.293421	0.249384	0.207142	0.239655	0.283870	
Ben Stokes	0.217824	0.346490	0.301096	0.112987	0.157822	0.141935	
Chris Morris	0.197115	0.286016	0.225010	0.244804	0.255521	0.212902	
Dwayne Bravo	0.290561	0.247449	0.286758	0.207142	0.275561	0.283870	
Imran Tahir	0.192197	0.206721	0.158482	0.489608	0.359900	0.402149	
Jofra Archer	0.303503	0.208881	0.252825	0.207142	0.242995	0.260214	
Kagiso Rabada	0.145992	0.241586	0.140703	0.470777	0.307293	0.283870	
Keemo Paul	0.234390	0.269046	0.251678	0.169480	0.197903	0.189246	

```
Lasith Malinga  0.217565  0.301443  0.261715  0.301297  0.365745  0.283870
Moeen Ali       0.323564  0.208573  0.269266  0.112987  0.141121  0.212902
Mohammad Nabi   0.283184  0.205179  0.231796  0.150649  0.161997  0.189246
Rashid Khan     0.274124  0.193763  0.212010  0.320129  0.314808  0.354837
Sam Curran      0.256263  0.302060  0.308743  0.188311  0.269716  0.212902
Sunil Narine    0.344272  0.241586  0.331684  0.188311  0.289757  0.283870
Trent Boult     0.295091  0.264727  0.311610  0.094155  0.136111  0.118279

                     TBB        4w       Mat
Andre Russell   0.197151  0.000000  0.319173
Ben Stokes      0.110012  0.000000  0.205182
Chris Morris    0.215668  0.000000  0.205182
Dwayne Bravo    0.269040  0.000000  0.273576
Imran Tahir     0.420443  0.534522  0.387567
Jofra Archer    0.281021  0.000000  0.250778
Kagiso Rabada   0.307163  0.534522  0.273576
Keemo Paul      0.177545  0.000000  0.182384
Lasith Malinga  0.293003  0.534522  0.273576
Moeen Ali       0.163385  0.000000  0.250778
Mohammad Nabi   0.190615  0.267261  0.182384
Rashid Khan     0.392123  0.000000  0.341971
Sam Curran      0.215668  0.267261  0.205182
Sunil Narine    0.289735  0.000000  0.273576
Trent Boult     0.124172  0.000000  0.113990
```

[7]:
```
raw_data *= weights
pd.DataFrame(data=raw_data, index=candidates, columns=attributes)
```

[7]:
```
                     SR      Econ       Avg      Wkts      Runs      Inns  \
Andre Russell   0.042581  0.052164  0.038793  0.027619  0.026628  0.025233
Ben Stokes      0.043565  0.061598  0.046837  0.015065  0.017536  0.012616
Chris Morris    0.039423  0.050847  0.035001  0.032641  0.028391  0.018925
Dwayne Bravo    0.058112  0.043991  0.044607  0.027619  0.030618  0.025233
Imran Tahir     0.038439  0.036750  0.024653  0.065281  0.039989  0.035747
Jofra Archer    0.060701  0.037134  0.039328  0.027619  0.026999  0.023130
Kagiso Rabada   0.029198  0.042949  0.021887  0.062770  0.034144  0.025233
Keemo Paul      0.046878  0.047830  0.039150  0.022597  0.021989  0.016822
Lasith Malinga  0.043513  0.053590  0.040711  0.040173  0.040638  0.025233
Moeen Ali       0.064713  0.037080  0.041886  0.015065  0.015680  0.018925
Mohammad Nabi   0.056637  0.036476  0.036057  0.020086  0.018000  0.016822
Rashid Khan     0.054825  0.034447  0.032979  0.042684  0.034979  0.031541
Sam Curran      0.051253  0.053700  0.048027  0.025108  0.029968  0.018925
Sunil Narine    0.068854  0.042949  0.051595  0.025108  0.032195  0.025233
Trent Boult     0.059018  0.047063  0.048473  0.012554  0.015123  0.010514

                     TBB        4w       Mat
Andre Russell   0.013143  0.000000  0.007093
```

```
Ben Stokes      0.007334  0.000000  0.004560
Chris Morris    0.014378  0.000000  0.004560
Dwayne Bravo    0.017936  0.000000  0.006079
Imran Tahir     0.028030  0.023757  0.008613
Jofra Archer    0.018735  0.000000  0.005573
Kagiso Rabada   0.020478  0.023757  0.006079
Keemo Paul      0.011836  0.000000  0.004053
Lasith Malinga  0.019534  0.023757  0.006079
Moeen Ali       0.010892  0.000000  0.005573
Mohammad Nabi   0.012708  0.011878  0.004053
Rashid Khan     0.026142  0.000000  0.007599
Sam Curran      0.014378  0.011878  0.004560
Sunil Narine    0.019316  0.000000  0.006079
Trent Boult     0.008278  0.000000  0.002533
```

[8]:
```
a_pos = np.zeros(n)
a_neg = np.zeros(n)
for j in range(n):
    column = raw_data[:,j]
    max_val = np.max(column)
    min_val = np.min(column)

    # See if we want to maximize benefit or minimize cost (for PIS)
    if j in benefit_attributes:
        a_pos[j] = max_val
        a_neg[j] = min_val
    else:
        a_pos[j] = min_val
        a_neg[j] = max_val

pd.DataFrame(data=[a_pos, a_neg], index=["$A^*$", "$A^-$"], columns=attributes)
```

[8]:
```
              SR      Econ       Avg      Wkts      Runs      Inns       TBB  \
$A^*$  0.029198  0.034447  0.021887  0.065281  0.015123  0.035747  0.028030
$A^-$  0.068854  0.061598  0.051595  0.012554  0.040638  0.010514  0.007334

              4w       Mat
$A^*$  0.023757  0.008613
$A^-$  0.000000  0.002533
```

[9]:
```
sp = np.zeros(m)
sn = np.zeros(m)
cs = np.zeros(m)

for i in range(m):
    diff_pos = raw_data[i] - a_pos
    diff_neg = raw_data[i] - a_neg
```

```
    sp[i] = np.sqrt(diff_pos @ diff_pos)
    sn[i] = np.sqrt(diff_neg @ diff_neg)
    cs[i] = sn[i] / (sp[i] + sn[i])

pd.DataFrame(data=zip(sp, sn, cs), index=candidates, columns=["$S^*$", "$S^-$",␣
↪"$C^*$"])
```

[9]:

	S^*	S^-	C^*
Andre Russell	0.056819	0.040468	0.415961
Ben Stokes	0.075085	0.034796	0.316672
Chris Morris	0.053264	0.043989	0.452315
Dwayne Bravo	0.062330	0.033812	0.351688
Imran Tahir	0.026770	0.081947	0.753762
Jofra Archer	0.060683	0.039074	0.391692
Kagiso Rabada	0.024786	0.079581	0.762512
Keemo Paul	0.062163	0.036585	0.370485
Lasith Malinga	0.048952	0.050299	0.506789
Moeen Ali	0.073078	0.037874	0.341355
Mohammad Nabi	0.061346	0.042463	0.409050
Rashid Khan	0.047658	0.055154	0.536456
Sam Curran	0.063257	0.030364	0.324327
Sunil Narine	0.072042	0.030814	0.299582
Trent Boult	0.078508	0.031140	0.284000

[10]:
```
def rank_according_to(data):
    ranks = (rankdata(data) - 1).astype(int)
    storage = np.zeros_like(candidates)
    storage[ranks] = candidates
    return storage[::-1]
```

[11]:
```
cs_order = rank_according_to(cs)
sp_order = rank_according_to(sp)
sn_order = rank_according_to(sn)

pd.DataFrame(data=zip(cs_order, sp_order[::-1], sn_order), index=range(1, m +␣
↪1),
             columns=["$C^*$", "$S^*$", "$S^-$"])
```

[11]:

	C^*	S^*	S^-
1	Kagiso Rabada	Kagiso Rabada	Imran Tahir
2	Imran Tahir	Imran Tahir	Kagiso Rabada
3	Rashid Khan	Rashid Khan	Rashid Khan
4	Lasith Malinga	Lasith Malinga	Lasith Malinga
5	Chris Morris	Chris Morris	Chris Morris
6	Andre Russell	Andre Russell	Mohammad Nabi
7	Mohammad Nabi	Jofra Archer	Andre Russell
8	Jofra Archer	Mohammad Nabi	Jofra Archer

```
9       Keemo Paul      Keemo Paul       Moeen Ali
10    Dwayne Bravo    Dwayne Bravo      Keemo Paul
11       Moeen Ali      Sam Curran      Ben Stokes
12      Sam Curran    Sunil Narine    Dwayne Bravo
13      Ben Stokes       Moeen Ali     Trent Boult
14    Sunil Narine      Ben Stokes    Sunil Narine
15     Trent Boult     Trent Boult      Sam Curran
```

[12]:
```
print("The best candidate/alternative according to C* is " + cs_order[0])
print("The preferences in descending order are " + ", ".join(cs_order) + ".")
```

```
The best candidate/alternative according to C* is Kagiso Rabada
The preferences in descending order are Kagiso Rabada, Imran Tahir, Rashid Khan,
Lasith Malinga, Chris Morris, Andre Russell, Mohammad Nabi, Jofra Archer, Keemo
Paul, Dwayne Bravo, Moeen Ali, Sam Curran, Ben Stokes, Sunil Narine, Trent
Boult.
```