Annoy (Fast Array Retrieval) -- James Mangan

Contents

1. Imports
2. Importing and cleaning nba dataset
3. What is Annoy?
4. Steps
5. Scatter Plot
6. Example

Imports

In [1]:
```
import pandas as pd
import numpy as np
import annoy
from annoy import AnnoyIndex
import plotly.express as px
import ipywidgets as widgets
from ipywidgets import interact

#pip install annoy
#pip install xlrd
#pip install openpyxl
```

Import and clean data

In [2]:
```
nba = pd.read_excel('nbaStats.xlsx') #updated as of last week
del nba['RANK'] # No values in entire dataframe
nba['id'] = np.arange(len(nba)) #easier to use in code
nba.head(6)
```

Out[2]:

	FULL NAME	TEAM	POS	AGE	GP	MPG	MIN%	USG%	TO%	FTA	...	TRB%	APG	AST%	S
0	Precious Achiuwa	Mia	F	21.54	47	12.6	26.3	19.5	14.8	94	...	15.9	0.5	6.7	0
1	Jaylen Adams	Mil	G	24.92	7	2.6	5.4	18.6	0.0	0	...	8.7	0.3	12.7	0.
2	Steven Adams	Nor	C	27.71	45	28.2	58.7	12.2	18.4	105	...	17.7	2.0	9.2	0
3	Bam Adebayo	Mia	C-F	23.72	44	33.6	70.1	24.1	14.9	253	...	15.9	5.3	27.4	0.
4	LaMarcus Aldridge	San	C-F	35.71	21	25.9	54.0	22.8	7.0	37	...	9.1	1.7	10.2	0.
5	LaMarcus Aldridge	Bro	C-F	35.71	2	28.0	58.3	17.0	4.6	2	...	12.0	3.5	17.1	1

6 rows × 29 columns

What is ANNOY?

Approximate Nearest Neighbors, Oh Yeah

The current implementation for finding k nearest neighbors in a vector space in gensim has linear complexity via brute force in the number of indexed documents, although with extremely low constant factors. The retrieved results are exact, which is an overkill in many applications: approximate results retrieved in sub-linear time may be enough. Annoy can find approximate nearest neighbors much faster. (source: https://markroxor.github.io/gensim/static/notebooks/annoytutorial.html)

Steps

In [3]:
```
# Step 1: Create the AnnoyIndex
# this requires the size of the vector and the type of Annoy you would like to d

annoy = AnnoyIndex(2, 'euclidean')

# The size of the vector can be much larger, but for this example I am going to
# There are two types of Annoy packages, euclidean and angular
```

In [4]:
```
# Step 2: Add items to the AnnoyIndex
# .add_item() has two arguments, i and v

df = nba.copy()

for n in range(len(df)):
    i = df.iloc[n,28]                          # i is the name that will be used
    vector = [df.iloc[n,26], df.iloc[n,27]]    # v is a vector of things you wou
    annoy.add_item(i,vector)

#Columns Used: 26-ORTG, 27-DRTG, 28-id
```

In [5]:
```
# Step 3: Make the decisions

annoy.build(10) # 1 is the number of 'trees' that will be made, the higher the n

# Note: at this point no more items can be added, in order to find a different r
```

Out[5]: True

In [6]:
```
# Step 4: Return your desired array
# You can do this with given stats or a player that you would like to find simil

arrayA = annoy.get_nns_by_vector([100,100], 5)     #look up players with specifi
print(arrayA)

arrayB = annoy.get_nns_by_item(298,10)              #look up players similar to
print(arrayB)

# player 298 is Lach LaVine
```

```
[39, 477, 558, 239, 462]
[298, 526, 224, 166, 326, 318, 41, 263, 478, 475]
```

In [9]:

```
# Step 5: Display names to make it easier to see

print(arrayA,ids_to_names(arrayA))
print(arrayB,ids_to_names(arrayB))
```

```
[39, 477, 558, 239, 462] ['Kent Bazemore', 'Alen Smailagic', 'James Wiseman', 'T
alen Horton-Tucker', 'Luka Samanic']
[298, 526, 224, 166, 326, 318, 41, 263, 478, 475] ['Zach LaVine', 'Dean Wade',
'Gordon Hayward', 'Evan Fournier', 'Lauri Markkanen', 'Trey Lyles', 'Bradley Bea
l', 'Ty Jerome', 'Marcus Smart', 'Anfernee Simons']
```

In [8]:

```
def ids_to_names(arr):
    array=[]
    for i in arr:
        array.append(nba.iloc[i,0])
    return(array)
```

Scatter Plot

In [10]:

```
dg = df.copy()

dg['cluster'] = 0
for i in arrayB:
    dg.iloc[i,29] = 1

px.scatter(dg,x="ORTG",y="DRTG",hover_name="FULL NAME",color="cluster")
```

Example

In [13]:

```
@interact(p=(2,10,1))
def f(p=2):
    annoy = AnnoyIndex(2, 'angular')

    df = nba.copy()
    dg = df.copy()

    for n in range(len(df)):
        i = df.iloc[n,28]
        vector = [df.iloc[n,26], df.iloc[n,27]]
        annoy.add_item(i,vector)

    annoy.build(10)

    arrayA = annoy.get_nns_by_item(298,p)

    dg['cluster'] = 0
    for i in arrayB:
        dg.iloc[i,29] = 1

    px.scatter(dg,x="ORTG",y="DRTG",hover_name="FULL NAME",color="cluster")
```

Sources:

https://pypi.org/project/annoy/

https://markroxor.github.io/gensim/static/notebooks/annoytutorial.html

https://github.com/spotify/annoy/blob/master/examples/simple_test.py

https://stackoverflow.com/questions/57039214/how-to-use-the-spotifys-annoy-library-in-python