CSE564: Visualization LAB Assignment 2
Part 2 (Task 3 and Task 4)
Spring 21

Google drive video link: https://drive.google.com/file/d/1fd7ttWnZKFpoGK6hBKHbVm_p-QYdIdw9/view?usp=sharing

Youtube link: www.youtube.com/watch?v=TAEnm2Y5lS0

Name- Mayank Jain
SBU Id- 113263864

Task 3: MDS plots (numerical data dimensions only)

- Construct the data MDS plot (use the Euclidian distance) and visualize it via a scatterplot (use metric MDS – python sklearn.manifold.MDS).
- Color the points by cluster ID (see task 3).
- Construct the variables' MDS plot (use the (1-|correlation|) distance) and visualize it via a scatterplot (also here, use metric MDS).

Task 4: Parallel coordinates plot (PCP)

- Visualize the data in a parallel coordinates plot (all data dimensions, categorical and numerical)
- Come up with a meaningful axes ordering by user interaction
- Color the polylines by cluster ID (see task 3)

Extra Credit: find a good PCP axes ordering from correlations

- numerical values only: use the correlations observed in the variables' MDS plot to help with the axis ordering -- the user would click on points in sequence and the axes would be arranged in that sequence

1. Dataset:

The dataset that I have chosen to work on this assignment is the FIFA 21 player's data which contains every possible detail of a player which could be of interest to this sport's or game's fans.

In the original data there are more than 17k players which have been analyzed based on total of 107 parameters. But, for this assignment I have chosen top **799** players according to their overall rating and total **13** attributes to plot on the graphs.

13 Numerical Variables- Overall Rating (OVA), Age, Height, Pace, Shot, Pass, Defend, Physical, Goalkeeping, Penalties, Crossing, Finishing and Dribbling.

data

	ID	Name	Age	OVA	Height	Pace	Shot	Pass	Defend	Physical	Goalkeeping	Penalties	Crossing	Finishing	Dribbling
0	158023	L. Messi	33	93	5.58	85	92	91	38	65	54	75	85	95	96
1	20801	Cristiano Ronaldo	35	92	6.17	89	93	81	35	77	58	84	84	95	88
2	188545	R. Lewandowski	31	91	6.0	78	91	78	43	82	51	88	71	94	85
3	190871	Neymar Jr	28	91	5.75	91	85	86	36	59	59	92	85	87	95
4	192985	K. De Bruyne	29	91	5.92	76	86	93	64	78	56	84	94	82	88
5	200389	J. Oblak	27	91	6.17	87	92	78	52	90	437	11	13	11	12
6	192448	M. ter Stegen	28	90	6.17	88	85	88	45	88	439	25	18	14	21
7	203376	V. van Dijk	28	90	6.33	76	60	71	91	86	58	62	53	52	70
8	208722	S. Mané	28	90	5.75	94	85	80	44	76	56	71	76	90	91
9	209331	M. Salah	28	90	5.75	93	86	81	45	75	62	83	79	91	90
10	212831	Alisson	27	90	6.25	86	88	85	51	91	439	23	17	13	27
11	231747	K. Mbappé	21	90	5.83	96	86	78	39	76	42	70	78	91	92
12	153079	S. Agüero	32	89	5.67	78	90	77	33	73	59	75	70	94	88
13	155862	Sergio Ramos	34	89	6.0	71	70	76	88	85	46	92	66	65	65
14	165153	K. Benzema	32	89	6.08	74	85	81	40	76	41	84	75	88	87
15	167495	M. Neuer	34	89	6.33	87	87	91	57	86	440	47	15	13	30
16	192119	T. Courtois	28	89	6.5	84	89	74	48	85	420	27	14	14	13
17	200145	Casemiro	28	89	6.08	65	73	76	86	91	67	66	58	64	69
18	121939	P. Lahm	32	88	5.58	67	56	82	86	64	47	69	84	47	82
19	162835	S. Handanovič	35	88	6.33	88	85	73	53	89	424	23	12	10	18
20	182521	T. Kroos	30	88	6.0	54	81	91	71	69	51	73	88	76	80
21	183277	E. Hazard	29	88	5.75	88	82	83	35	66	45	87	77	82	93
22	201024	K. Koulibaly	29	88	6.17	75	28	55	89	86	43	33	30	22	69
23	202126	H. Kane	26	88	6.17	68	91	80	47	83	54	90	75	94	80
24	202652	R. Sterling	25	88	5.58	93	81	79	45	67	63	69	78	85	90
25	210257	Ederson	26	88	6.17	86	82	93	63	86	435	17	20	14	23
26	211110	P. Dybala	26	88	5.83	85	85	84	43	63	26	86	82	84	91
27	212622	J. Kimmich	25	88	5.75	71	72	86	81	79	60	44	91	66	83
28	215914	N. Kanté	29	88	5.5	77	66	76	86	82	54	54	68	65	79
29	488	O. Kahn	38	87	6.17	83	88	62	60	92	346	21	21	21	21
30	138956	G. Chiellini	35	87	6.17	66	46	58	90	79	15	50	54	33	59

2. Backend setup:

All the plots for tasks 1 and 2 are made using d3.js and the data required for that is fetched from live Python(Flask) server using various APIs.

Python file: PCA.py

Imports required to run the python code:

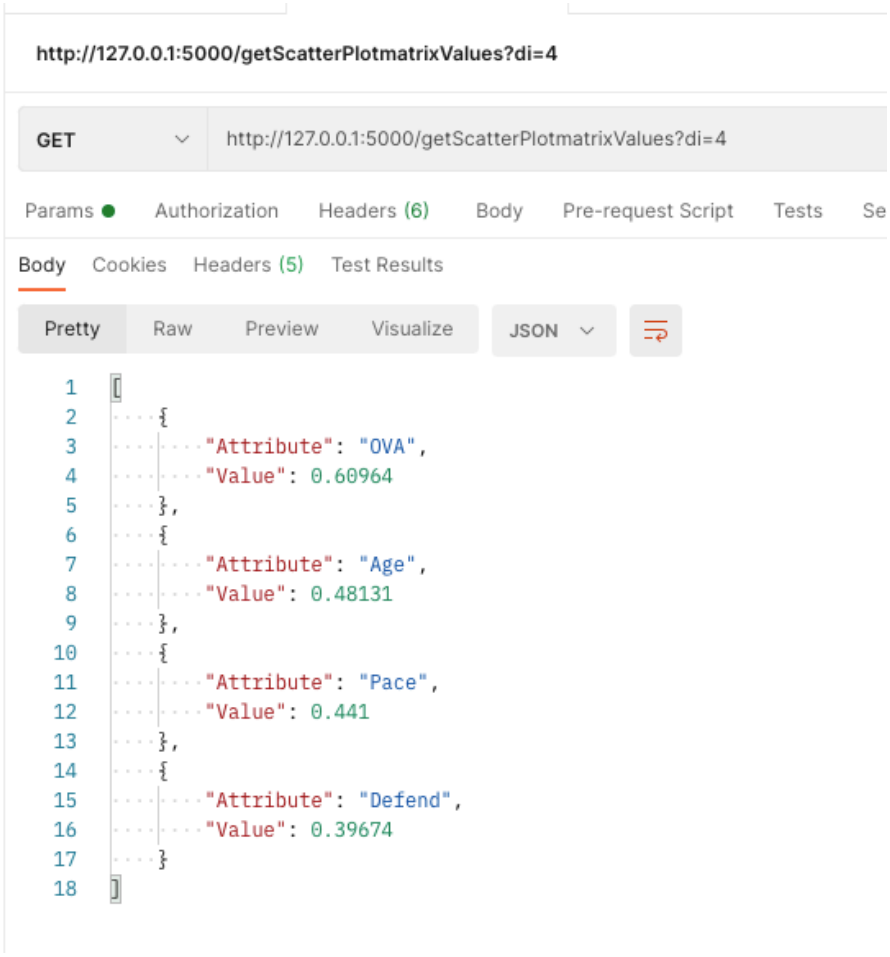
```
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
from scipy.spatial.distance import cdist
from sklearn.cluster import KMeans
from flask import Flask
from flask import request, jsonify
from kneed import KneeLocator
from sklearn.metrics import pairwise_distances
from sklearn import manifold
```

APIs made for fetching the data:

```
app = Flask(__name__)

@app.route('/', methods=['GET'])
def index():
    return "Hello!!"

@app.route('/getpca', methods=['GET'])
def getpca():
    response = jsonify(pca())
    response.headers.add('Access-Control-Allow-Origin', '*')
    return response

@app.route('/getpca1and2', methods=['GET'])
def getpca1and2():
    response = jsonify(pca1and2())
    response.headers.add('Access-Control-Allow-Origin', '*')
    return response

@app.route('/getmds_euc', methods=['GET'])
def getmds_euc():
    response = jsonify(getmds_euclidian())
    response.headers.add('Access-Control-Allow-Origin', '*')
    return response

@app.route('/getmds_corr', methods=['GET'])
def getmds_corr():
    response = jsonify(getmds_correlation())
    response.headers.add('Access-Control-Allow-Origin', '*')
    return response


@app.route('/getScatterPlotmatrixValues', methods=['GET'])
def getScatterPlotmatrixValues():
    response = jsonify(getSquaredList(int(request.args.get("di"))))
    response.headers.add('Access-Control-Allow-Origin', '*')
    return response

if __name__ == "__main__":
    app.run(debug=True)
```

Sample API call:

- To fetch top 4 Attributes when Intrinsic dimensionality index is passed as 4 from the D3 interaction element



3. Web page design:

(a) Home Page -

By default Scree plot appears on the home page.

Five buttons provided on Navigation bar on top to choose the type of plot the user wants to see.

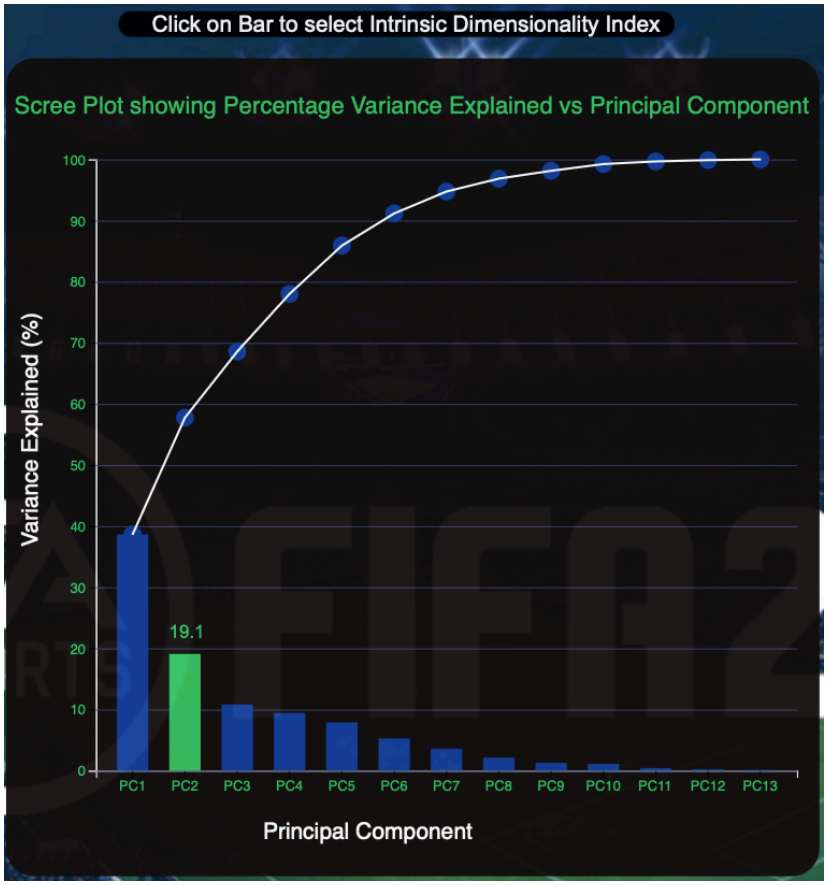
- Button 1 : Scree Plot
- Button 2 : PCA based Bi Plot
- Button 3 : MDS Euclidian
- Button 4 : MDS Attributes
- Button 5 : PCP Plot

(b) Scree Plot-

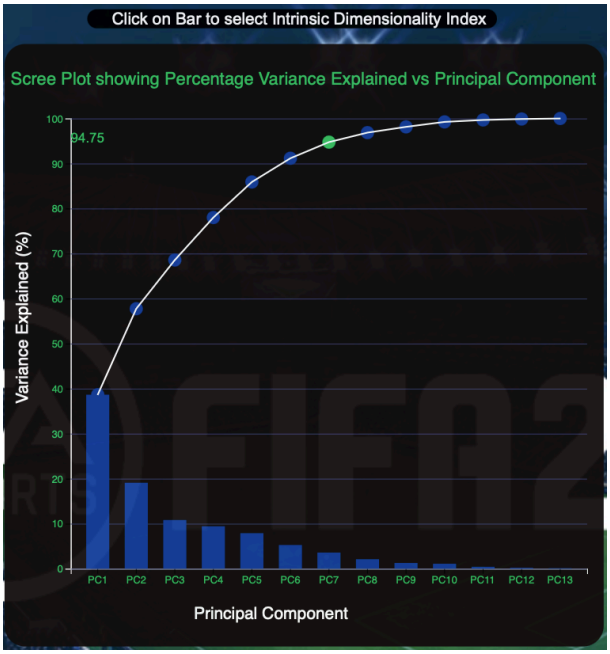
To implement interaction elements using d3, option is provided to click on any bar and get the corresponding scatter plot matrix for that dimensionality index. (Bar number is taken as the dimensionality index that the user has chosen)

- Variance Explained Percentage on Y- axis and Principal Components on X- axis.

- Hover on any bar to see the corresponding variance percentage for that PC number.



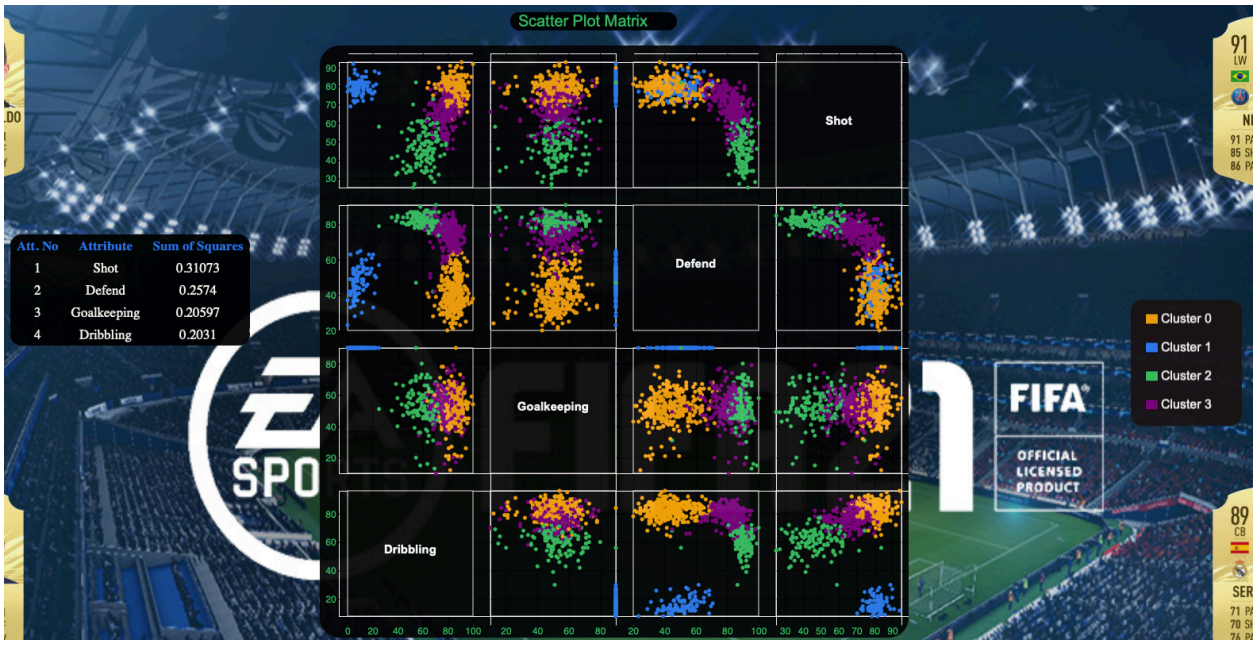
- Hover on the circles to see the cumulative percentage variance till that PC number.



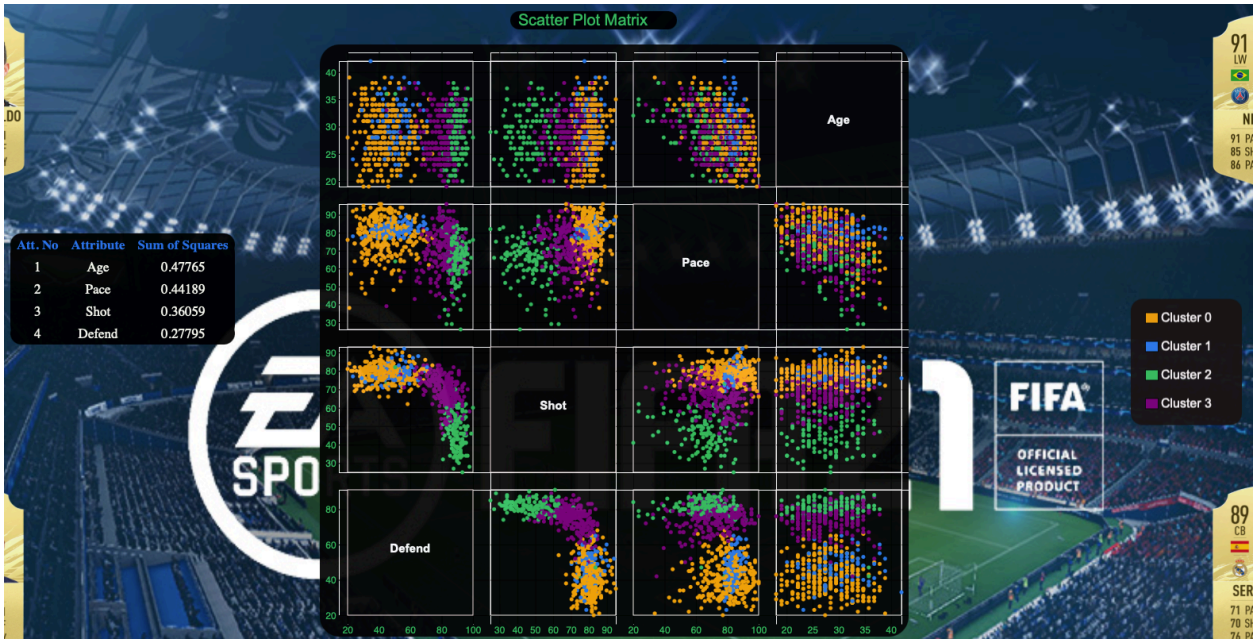
(c) Scatter Plot Matrix-

- Click on any bar (PC) to plot the scatter matrix and view top 4 Attributes for that dimensionality index. Table is generated dynamically on the webpage based on the dimensionality index chosen

Eg. 1- When clicked on PC2 bar (di = 2)

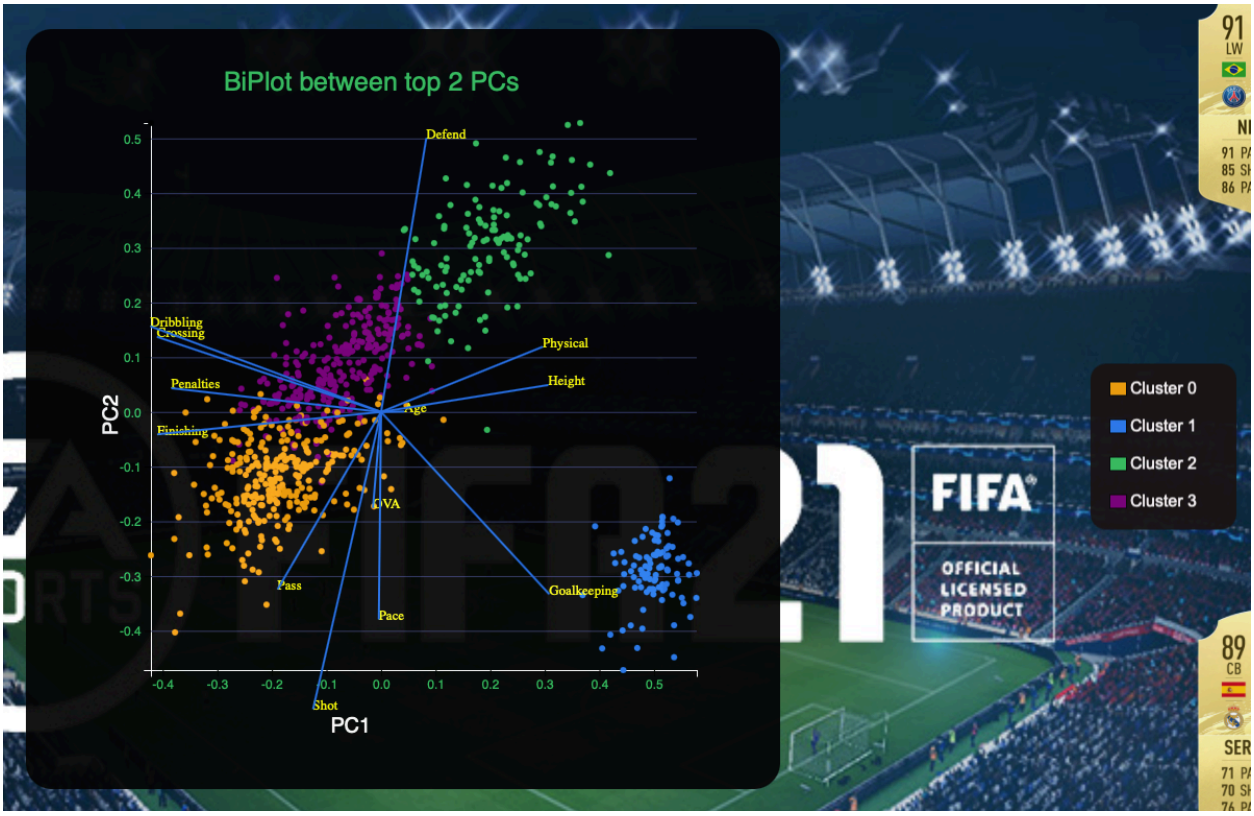


Eg. 2- When clicked on PC3 bar (di = 3)



(d) PCA based Biplot-

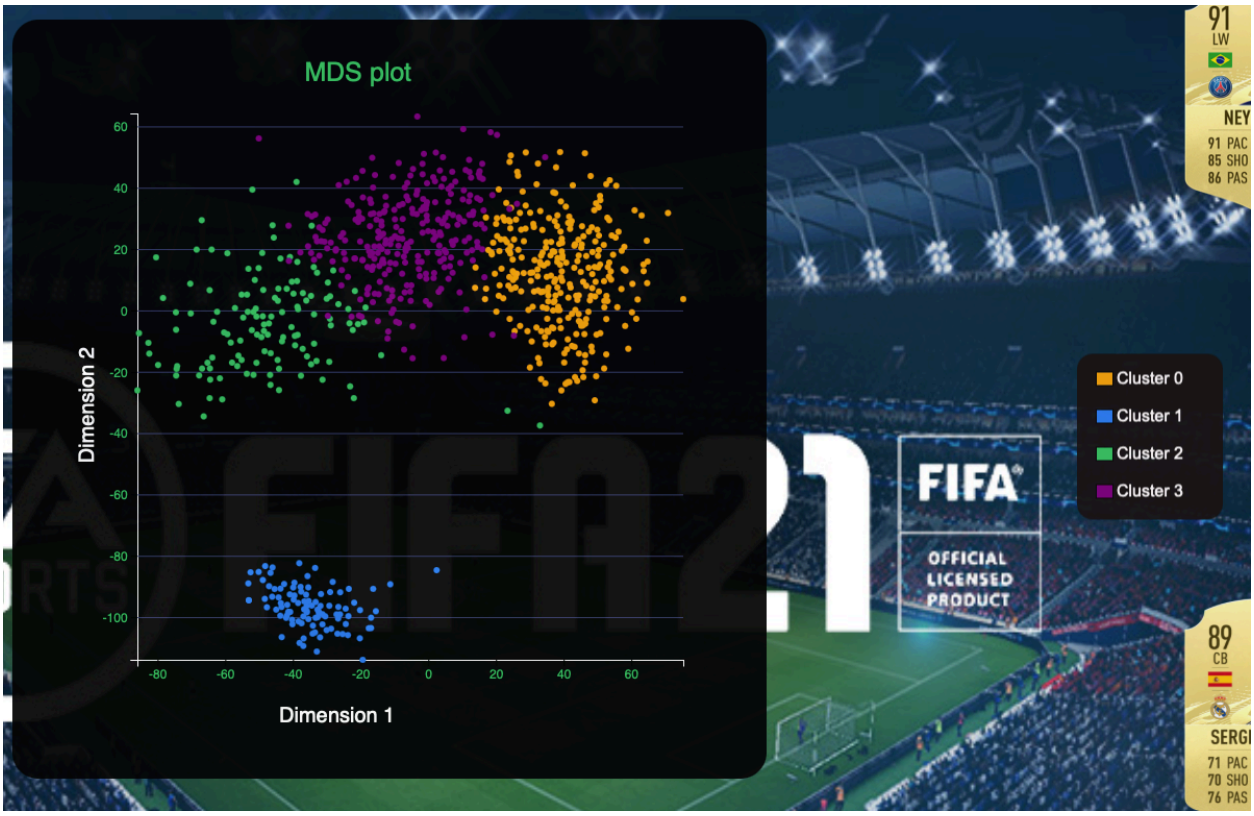
- Can hover on any attribute name or the attribute line to highlight it.



Note :-
Color Legends in the ScatterPlot Matrix and the Biplot are selected on the basis of cluster id which is calculated by implementing k-Means in python backend code (gave 4 clusters after the implementation).
Cluster Id is appended in the data fetched initially from data.csv file and a newdata.csv file is made with 1 column extra containing the Cluster id with column name- “ColorCluster”.

(e) MDS Plot-

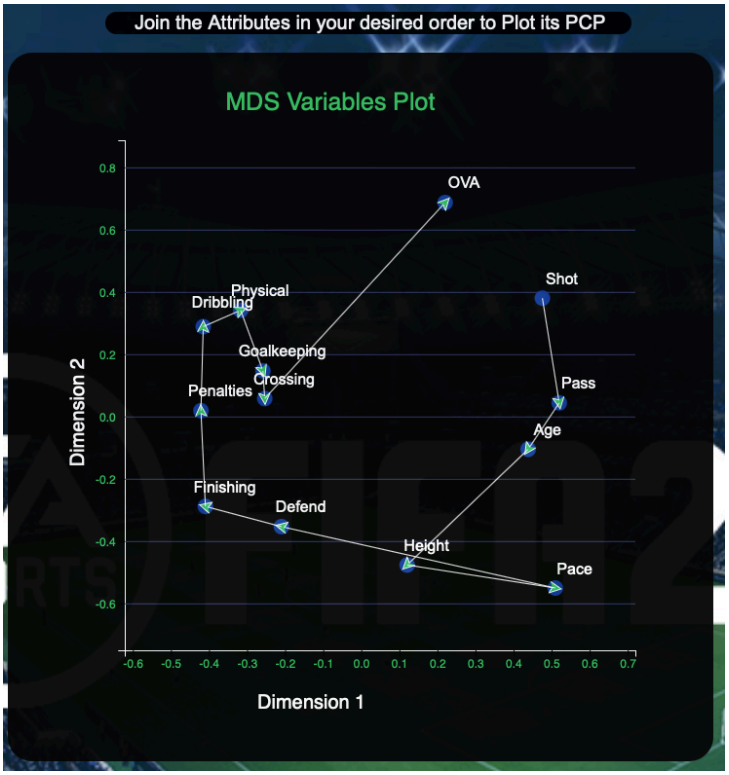
MDS plot created using euclidean dissimilarity by clicking on the “MDS Euclidian” button provided on top of homepage. Here it can be seen that the data is clustered into different clusters.



(f) MDS Variables Plot-

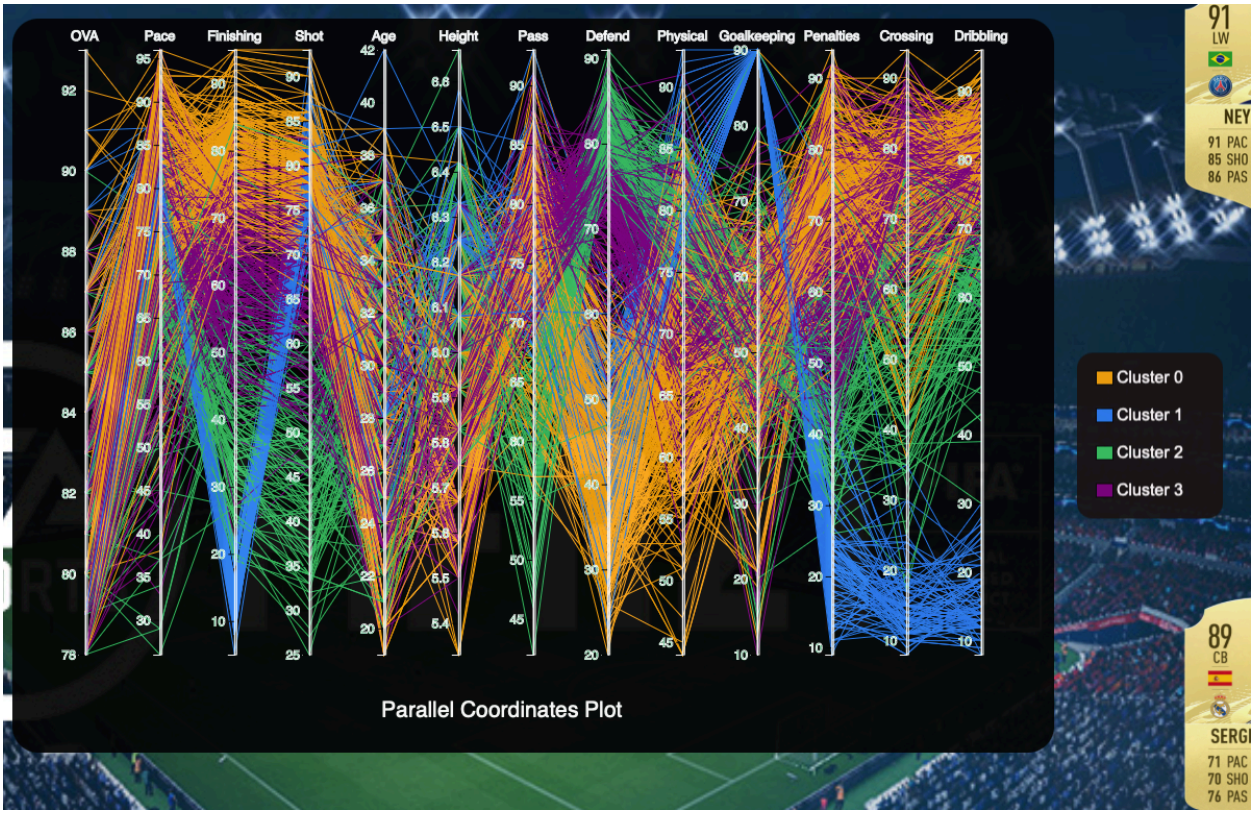
MDS variables plot created using 1-Icorrelationl distance by clicking on the button provided on the navigation bar

In this plot, user gets the option to select the ordering of the attributes that can be visualized on the parallel coordinates plot. Ordering can be done by clicking on the attribute circles plotted as scatter plot. This was done as a part of extra credit requirement.



(g) Parallel Coordinates Plot-

When clicked on the PCP plot button on the navigation bar, a PCP plot appears which has one default ordering.

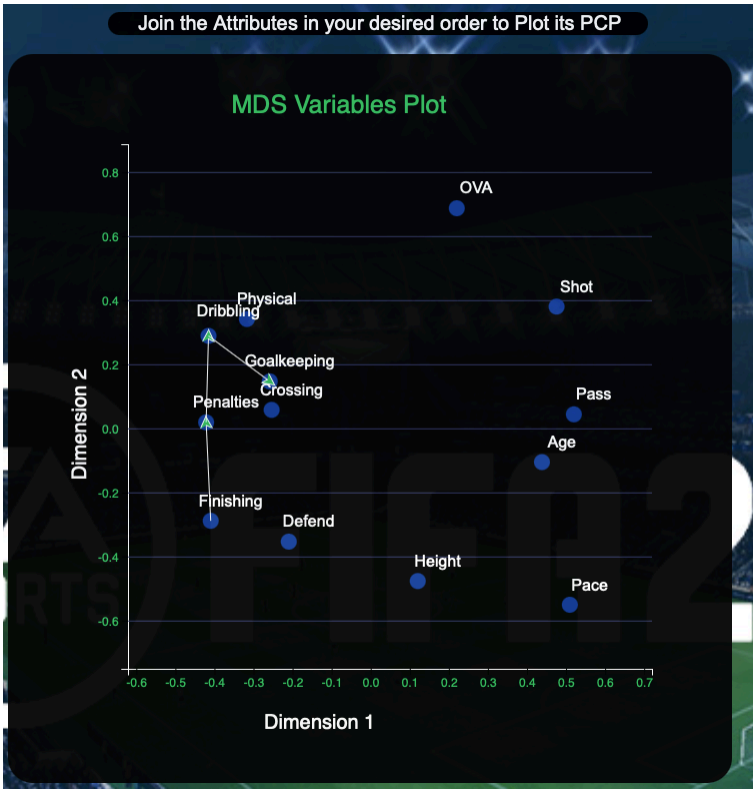


Noteworthy points about the implementation:

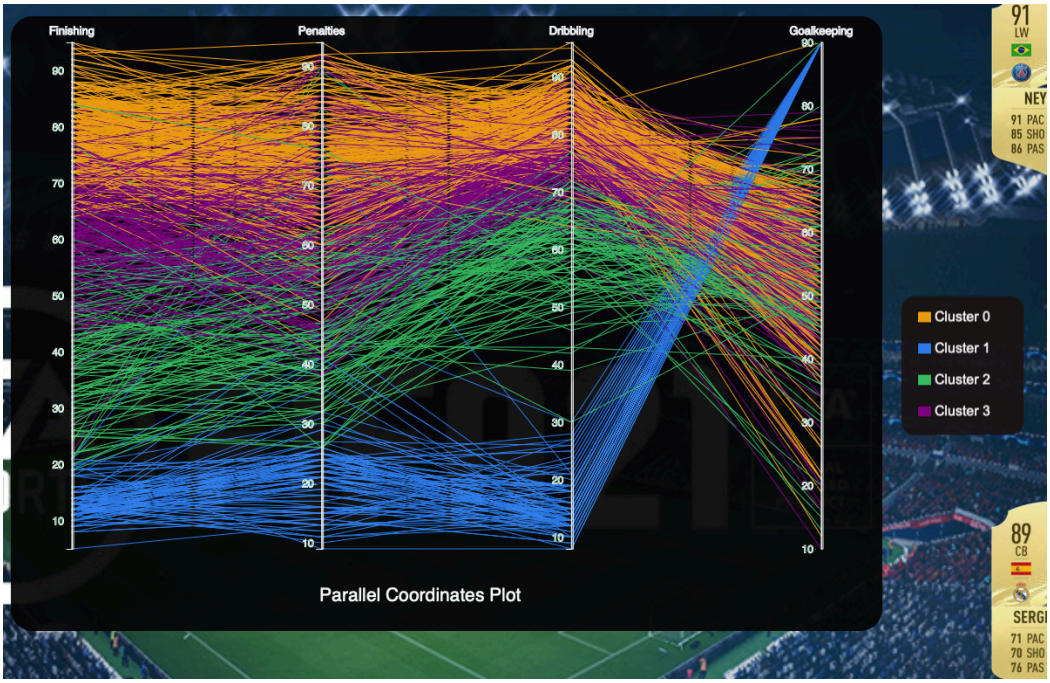
- This ordering and even the number of attributes to be shown in PCP plot can be changed by selecting the attributes in “MDS Attributes” plot and then clicking on the “PCP plot” button.

For eg.

If the user selects just 4 attributes in this order: “ Finishing -> Penalties -> Dribbling -> Goalkeeping”.

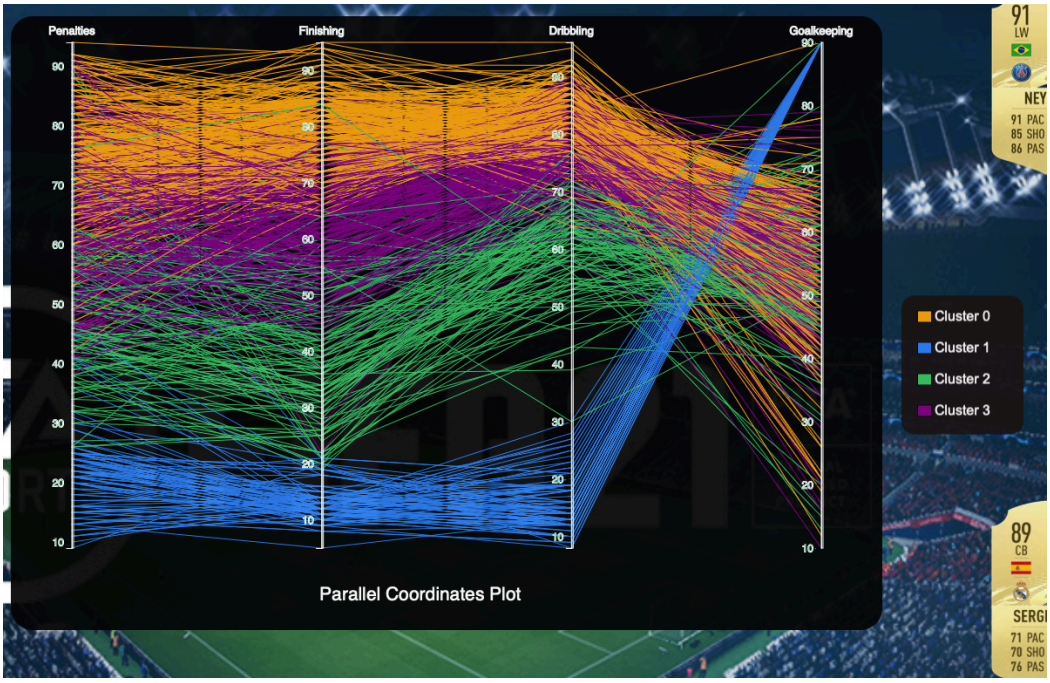


Then the following PCP plot will appear if the user clicks on the PCP button:

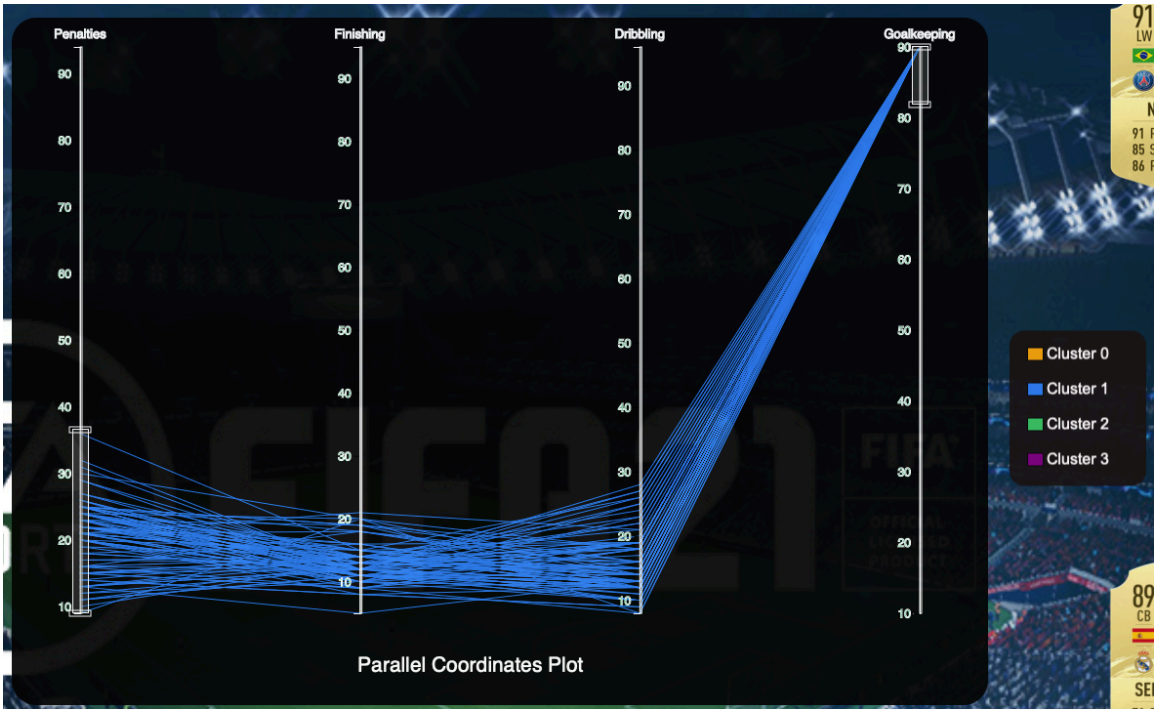


- Users can also click on any axis and drag it to left or right depending on their choice of which attribute they want to see first.

Below it can be seen that “Penalties” attributes is shifted to the beginning it is dragged to left.



- Using brushing feature implemented in PCP plot user can select a range of few variables and see the relation among them .



Some interesting observations can be made using the PCP plot of some selected variables.
As in above diagram, you can see that usually the High rated goalkeepers will have low finishing/penalty taking and dribbling capabilities. This negative relation can be confirmed from Biplot as well that we have already plotted. In the biplot it can be seen that the 4 selected attributes lie opposite to each other.