FInalExam_WQD180104_Q5

July 8, 2020

1 Final Exam

1.1 Question 5

5.1 Importing required libraries

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import DBSCAN
from sklearn.cluster import KMeans
```

5.2 Load the dataset into a DataFrame object

[2]:
```
df = pd.read_csv('data/players_stats.csv')
df.head()
```

[2]:

	name	photo_url	positions	age	birth_date	height	weight	football_club	national_team
0	Lionel Andrés Messi Cuccittini	https://cdn.sofifa.com/players/158/023/20_120.png	RW,ST,CF	32	1987/Jun/24	170	72	FC Barcelona	Argentina
1	Cristiano Ronaldo dos Santos Aveir	https://cdn.sofifa.com/players/020/801/20_120.png	ST,LW	34	1985/Feb/5	187	83	Juventus	Portugal
2	Neymar da Silva Santos Júnior	https://cdn.sofifa.com/players/190/871/20_120.png	LW,CAM	27	1992/Feb/5	175	68	Paris Saint-Germain	Brazil
3	Virgil van Dijk	https://cdn.sofifa.com/players/203/376/20_120.png	CB	27	1991/Jul/8	193	92	Liverpool	Netherlands
4	Jan Oblak	https://cdn.sofifa.com/players/200/389/20_120.png	GK	26	1993/Jan/7	188	87	Atlético Madrid	Slovenia

```
   overall_rating  …  Strength  LongShots  Aggression Interceptions  \
0              94  …        68         94          48            40
1              93  …        78         93          63            29
2              92  …        49         85          51            36
3              91  …        92         64          83            90
4              91  …        78         12          34            19

   Positioning Vision  Penalties  DefensiveAwareness  StandingTackle  \
0           94     94         75                  33              37
1           95     82         85                  28              32
2           87     90         92                  35              30
3           47     65         62                  93              93
4           11     65         11                  27              12

  SlidingTackle
0            26
1            24
2            29
3            86
4            18

[5 rows x 51 columns]
```

[3]:
```
df.columns
```

[3]:
```
Index(['name', 'photo_url', 'positions', 'age', 'birth_date', 'height',
       'weight', 'football_club', 'national_team', 'overall_rating',
       'potential', 'value', 'wages', 'best_position', 'best_rating',
       'Preferred Foot', 'Weak Foot', 'Skill Moves',
       'International Reputation', 'Work Rate', 'Body Type', 'Real Face',
       'Release Clause', 'Crossing', 'Finishing', 'HeadingAccuracy',
       'ShortPassing', 'Volleys', 'Dribbling', 'Curve', 'FKAccuracy',
       'LongPassing', 'BallControl', 'Acceleration', 'SprintSpeed', 'Agility',
       'Reactions', 'Balance', 'ShotPower', 'Jumping', 'Stamina', 'Strength',
       'LongShots', 'Aggression', 'Interceptions', 'Positioning', 'Vision',
       'Penalties', 'DefensiveAwareness', 'StandingTackle', 'SlidingTackle'],
      dtype='object')
```

5.3 Visualize the data, use only two of these attributes at the time

[4]:
```
selectedAttr = df.sample(250)[['overall_rating', 'potential']]
selectedAttr
```

[4]:
```
       overall_rating  potential
11500              65         65
18977              53         63
```

```
19370              51          60
19581              49          69
7590               68          68
…                  …           …
11706              65          81
9949               66          68
8860               67          76
6594               69          69
282                82          82

[250 rows x 2 columns]
```

5.4 Normalise the attributes

[5]:
```
scaler = StandardScaler().fit(selectedAttr)
data = scaler.transform(selectedAttr)
data
```

[5]:
```
array([[-0.19546059, -0.98490243],
       [-1.95109463, -1.30159131],
       [-2.24370031, -1.77662464],
       [-2.53630598, -0.35152466],
       [ 0.24344792, -0.5098691 ],
       [ 0.24344792, -0.19318022],
       [-0.78067194, -1.6182802 ],
       [-0.78067194, -0.35152466],
       [ 0.09714508, -0.19318022],
       [ 0.53605359, -0.19318022],
       [ 0.38975076,  0.91523087],
       [ 1.12126494,  1.3902642 ],
       [-1.36588329, -0.03483578],
       [-1.95109463, -2.88503573],
       [-1.21958045,  0.12350866],
       [ 1.12126494,  1.54860864],
       [ 0.53605359, -0.19318022],
       [-0.48806626,  0.12350866],
       [-1.21958045,  0.28185311],
       [ 0.24344792, -0.5098691 ],
       [-0.48806626, -0.19318022],
       [ 1.56017345,  0.91523087],
       [ 0.38975076,  0.91523087],
       [-1.36588329, -0.98490243],
       [-0.6343691 , -1.45993576],
       [ 0.24344792, -0.5098691 ],
       [ 1.56017345,  0.91523087],
       [ 0.09714508,  1.70695309],
       [ 0.97496211,  0.44019755],
```

```
[ 0.24344792,  0.28185311],
[-0.34176343, -1.14324687],
[-0.48806626,  0.28185311],
[ 1.26756778,  0.59854199],
[ 1.85277913,  1.23191976],
[ 0.24344792,  0.12350866],
[ 1.12126494,  0.44019755],
[ 0.97496211,  1.3902642 ],
[-0.34176343, -1.14324687],
[ 0.82865927,  0.12350866],
[-0.48806626, -0.19318022],
[ 0.09714508,  0.12350866],
[-0.34176343,  1.07357532],
[-0.34176343,  0.28185311],
[ 0.24344792,  0.44019755],
[ 0.82865927,  1.07357532],
[-0.19546059, -0.98490243],
[ 0.09714508, -0.66821355],
[-0.34176343, -0.66821355],
[-1.51218612, -0.82655799],
[-1.07327761, -1.93496908],
[-0.34176343, -1.14324687],
[ 0.09714508,  1.86529753],
[ 0.53605359, -0.19318022],
[ 1.99908196,  1.3902642 ],
[ 1.70647629,  1.54860864],
[-0.92697477, -1.77662464],
[-0.6343691 , -0.82655799],
[ 1.26756778,  0.59854199],
[ 0.68235643,  1.07357532],
[ 1.12126494,  0.44019755],
[ 0.97496211,  1.07357532],
[-0.04915775, -0.82655799],
[-1.65848896,  0.12350866],
[ 0.68235643,  0.44019755],
[ 2.1453848 ,  2.34033085],
[ 0.24344792, -0.5098691 ],
[-0.92697477, -0.19318022],
[ 1.12126494,  2.81536418],
[-0.34176343,  1.23191976],
[ 0.38975076, -0.35152466],
[-0.48806626, -1.14324687],
[ 0.97496211,  1.86529753],
[-0.48806626,  0.91523087],
[-0.19546059,  1.70695309],
[-0.34176343, -1.14324687],
[-1.51218612, -2.25165797],
```

```
[-0.78067194, -0.5098691 ],
[-1.51218612, -0.19318022],
[ 0.97496211,  0.28185311],
[-1.8047918 , -0.82655799],
[-0.6343691 ,  0.75688643],
[ 0.68235643,  0.91523087],
[ 0.38975076, -0.35152466],
[ 0.82865927,  1.07357532],
[-1.65848896,  0.59854199],
[ 0.38975076,  0.59854199],
[-1.65848896, -1.14324687],
[ 0.68235643, -0.03483578],
[-0.6343691 ,  0.12350866],
[-0.04915775, -0.19318022],
[-1.21958045, -0.5098691 ],
[ 0.68235643, -0.03483578],
[ 1.12126494,  0.44019755],
[-0.6343691 , -1.45993576],
[ 2.87689898,  2.97370862],
[ 0.09714508,  0.91523087],
[-1.21958045, -0.98490243],
[-1.51218612, -2.41000241],
[-0.48806626, -0.98490243],
[-0.6343691 ,  0.12350866],
[-0.04915775, -0.19318022],
[ 0.82865927,  0.12350866],
[ 0.09714508,  0.12350866],
[-0.6343691 , -0.98490243],
[ 0.68235643, -0.03483578],
[-0.48806626, -1.30159131],
[ 0.09714508, -0.66821355],
[-1.95109463, -0.98490243],
[-0.78067194,  0.28185311],
[ 0.09714508, -0.35152466],
[-0.48806626,  0.44019755],
[-0.48806626,  2.34033085],
[ 2.43799047,  2.34033085],
[ 1.41387062,  0.75688643],
[ 0.82865927,  0.28185311],
[-0.48806626, -0.03483578],
[ 0.97496211,  0.28185311],
[ 0.68235643,  1.23191976],
[ 1.85277913,  2.18198641],
[ 1.70647629,  1.07357532],
[ 0.09714508, -0.03483578],
[ 0.09714508,  0.12350866],
[ 0.24344792, -0.5098691 ],
```

```
[-1.51218612, -0.5098691 ],
[ 0.82865927,  1.3902642 ],
[-0.78067194, -1.6182802 ],
[ 0.24344792, -0.5098691 ],
[-0.19546059, -0.35152466],
[-0.92697477,  0.44019755],
[-0.48806626, -1.30159131],
[ 0.38975076,  0.28185311],
[ 1.12126494,  0.59854199],
[ 0.53605359, -0.03483578],
[ 1.56017345,  1.07357532],
[-0.48806626, -0.19318022],
[ 0.53605359,  0.91523087],
[-0.19546059, -0.19318022],
[ 1.85277913,  1.23191976],
[-0.48806626, -0.98490243],
[-0.6343691 , -0.82655799],
[-0.04915775, -0.66821355],
[ 0.09714508,  1.70695309],
[ 0.24344792, -0.5098691 ],
[-0.04915775, -0.82655799],
[ 0.82865927,  0.28185311],
[ 0.24344792,  0.75688643],
[-2.24370031, -1.14324687],
[-0.04915775, -0.82655799],
[ 0.24344792, -0.35152466],
[ 1.56017345,  1.23191976],
[-0.34176343, -1.14324687],
[ 0.68235643, -0.03483578],
[ 0.38975076, -0.35152466],
[ 0.24344792, -0.5098691 ],
[-0.04915775, -0.5098691 ],
[-0.19546059, -0.66821355],
[-0.92697477, -0.19318022],
[-0.34176343,  0.75688643],
[ 0.97496211,  0.28185311],
[ 0.68235643, -0.03483578],
[-1.36588329, -0.5098691 ],
[ 0.68235643,  0.44019755],
[-0.19546059, -0.35152466],
[ 0.24344792,  1.3902642 ],
[ 0.09714508, -0.66821355],
[ 1.56017345,  0.91523087],
[-0.92697477,  0.75688643],
[-0.19546059,  0.44019755],
[-1.51218612, -0.98490243],
[ 1.12126494,  1.23191976],
```

```
[-1.51218612, -1.45993576],
[ 1.70647629,  1.70695309],
[ 0.24344792, -0.5098691 ],
[-1.51218612, -0.98490243],
[ 0.53605359, -0.19318022],
[-1.8047918 , -0.19318022],
[ 1.70647629,  2.4986753 ],
[ 0.38975076, -0.19318022],
[ 0.09714508, -0.66821355],
[-1.07327761, -0.5098691 ],
[ 0.97496211,  1.86529753],
[-0.04915775, -0.82655799],
[-0.34176343, -1.14324687],
[-1.21958045, -0.35152466],
[-0.78067194, -1.6182802 ],
[-1.8047918 ,  0.28185311],
[-0.04915775, -0.82655799],
[-1.51218612, -0.35152466],
[ 1.70647629,  1.07357532],
[ 0.38975076, -0.19318022],
[-2.68260882, -1.45993576],
[-0.34176343, -1.14324687],
[ 1.26756778,  0.59854199],
[ 0.09714508, -0.66821355],
[-0.34176343, -1.14324687],
[ 0.38975076, -0.35152466],
[-0.34176343, -0.35152466],
[-0.48806626, -1.30159131],
[ 1.26756778,  0.59854199],
[-0.6343691 ,  0.12350866],
[ 1.41387062,  0.75688643],
[-0.78067194, -0.98490243],
[ 0.24344792, -0.5098691 ],
[-0.34176343, -0.66821355],
[ 1.70647629,  1.23191976],
[-1.36588329, -2.25165797],
[-1.07327761, -0.5098691 ],
[-0.34176343, -0.35152466],
[ 0.24344792, -0.35152466],
[ 0.82865927,  0.75688643],
[ 0.68235643,  1.54860864],
[ 0.24344792, -0.5098691 ],
[-0.92697477, -0.66821355],
[-0.6343691 , -0.35152466],
[ 1.56017345,  0.91523087],
[ 1.41387062,  1.07357532],
[ 0.68235643, -0.03483578],
```

```
       [ 0.09714508, -0.66821355],
       [-0.48806626, -0.19318022],
       [-1.21958045, -0.19318022],
       [-0.34176343, -0.66821355],
       [-0.34176343, -0.03483578],
       [-0.92697477, -1.77662464],
       [ 0.24344792, -0.03483578],
       [ 1.12126494,  0.59854199],
       [-0.92697477, -0.66821355],
       [-0.48806626,  0.28185311],
       [-1.65848896, -0.5098691 ],
       [ 0.24344792, -0.5098691 ],
       [ 0.53605359,  0.12350866],
       [-0.48806626, -1.30159131],
       [-0.6343691 , -1.45993576],
       [-1.21958045, -0.19318022],
       [ 1.56017345,  0.91523087],
       [ 0.38975076, -0.19318022],
       [ 0.68235643,  1.54860864],
       [-1.21958045,  0.12350866],
       [-0.19546059,  1.07357532],
       [ 0.53605359,  0.28185311],
       [ 0.38975076, -0.35152466],
       [ 0.24344792,  1.70695309],
       [-1.8047918 , -0.35152466],
       [ 1.56017345,  0.91523087],
       [-0.92697477, -1.77662464],
       [ 0.53605359, -0.19318022],
       [-0.19546059,  1.54860864],
       [-0.04915775, -0.5098691 ],
       [ 0.09714508,  0.75688643],
       [ 0.38975076, -0.35152466],
       [ 2.29168764,  1.70695309]])
```

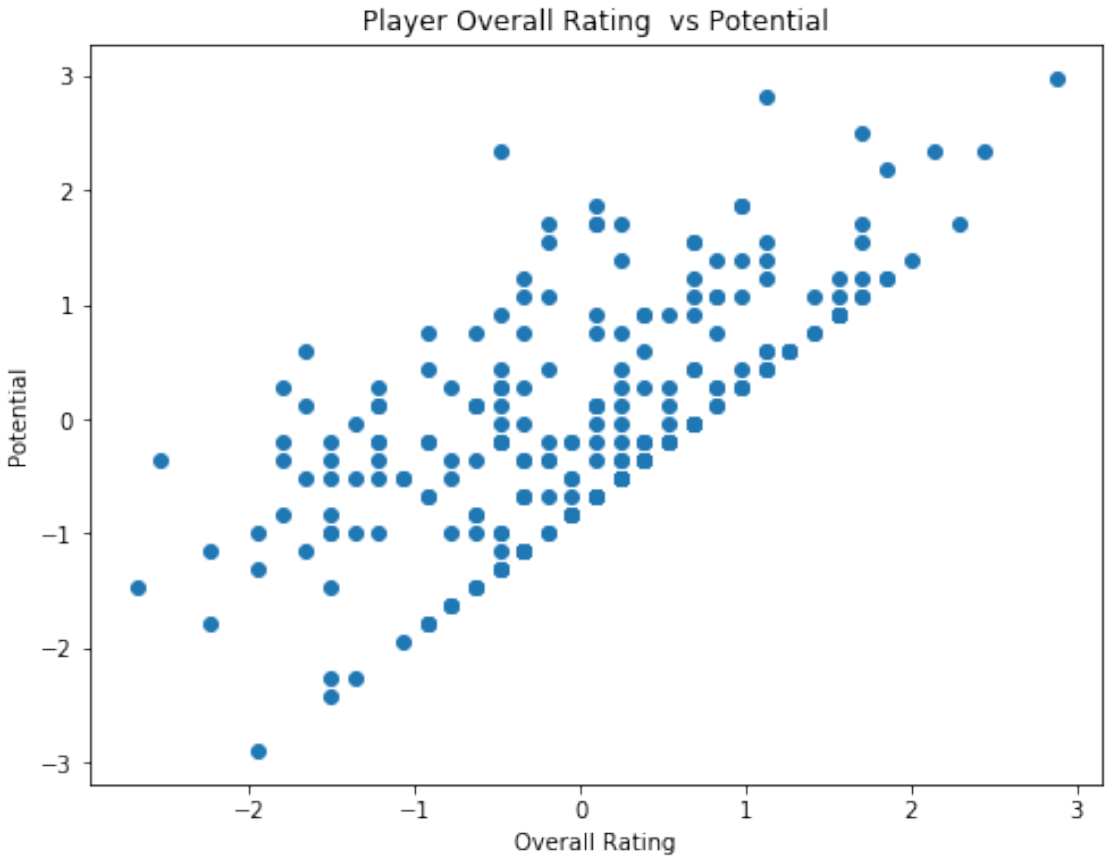
5.5 Show correlation

[6]:
```
selectedAttr.corr(method = 'pearson')
```

[6]:

	overall_rating	potential
overall_rating	1.000000	0.663993
potential	0.663993	1.000000

[7]:
```
plt.figure(figsize=(8, 6))
plt.scatter(data[:,0], data[:,1])
plt.xlabel("Overall Rating")
plt.ylabel("Potential")
plt.title("Player Overall Rating  vs Potential");
```



5.6 Construct a density-based clustering model and extract cluster labels and outliers to plot your results

[8]:
```
db = DBSCAN(eps=0.3, min_samples=5).fit(data)
core_samples_mask = np.zeros_like(db.labels_, dtype=bool)
core_samples_mask[db.core_sample_indices_] = True
labels = db.labels_
```

[9]:
```
n_clusters_ = np.unique(labels)
n_clusters_
```

[9]:
```
array([-1,  0,  1,  2,  3], dtype=int64)
```

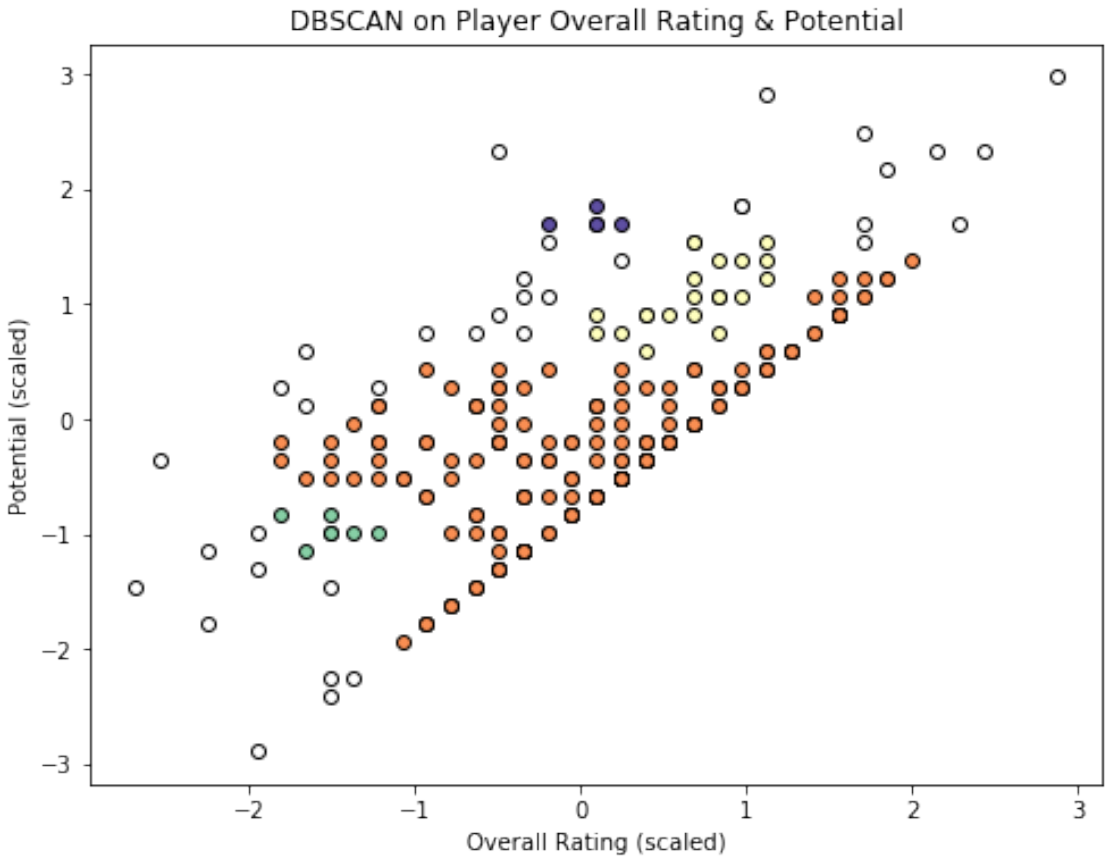
[10]:
```
colors = plt.cm.Spectral(np.linspace(0,1, len(n_clusters_)))
colors
```

[10]:
```
array([[0.61960784, 0.00392157, 0.25882353, 1.        ],
       [0.97485582, 0.557401  , 0.32272203, 1.        ],
       [0.99807766, 0.99923106, 0.74602076, 1.        ],
       [0.52733564, 0.8106113 , 0.64521338, 1.        ],
```

```
        [0.36862745, 0.30980392, 0.63529412, 1.        ]])
```

5.6.1 Density-based spatial clustering of applications with noise (DBSCAN)

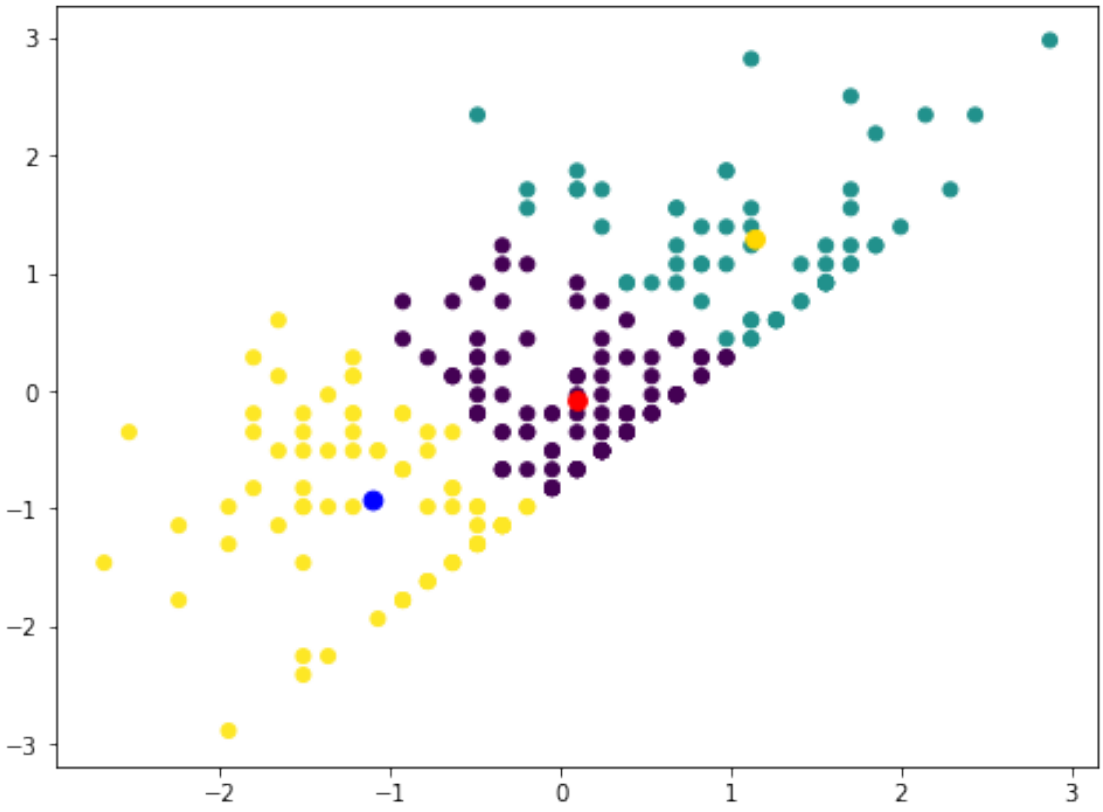
[11]:
```
plt.figure(figsize=(8, 6))
for (label, color) in zip(n_clusters_, colors):
    if label == -1:
        # White used for noise.
        color = 'white'

    class_member_mask = (labels == label)
    xy = data[class_member_mask & core_samples_mask]
    plt.plot(xy[:,0],xy[:,1], 'o', markerfacecolor = color,
                                   markersize = 6,
                                   markeredgecolor='k')

    xy2 = data[class_member_mask & ~core_samples_mask]
    plt.plot(xy2[:,0],xy2[:,1], 'o', markerfacecolor = color,
                                     markersize = 6,
                                     markeredgecolor='k')

plt.title("DBSCAN on Player Overall Rating & Potential")
plt.xlabel("Overall Rating (scaled)")
plt.ylabel("Potential (scaled)");
```



5.6.2 K-means Clustering

[12]:
```
kmeans = KMeans(n_clusters = 3).fit(data)
labels_kmeans = kmeans.labels_
centroids = kmeans.cluster_centers_
```

[13]:
```
plt.figure(figsize=(8, 6))
plt.scatter(data[:,0], data[:,1], c = labels_kmeans)
plt.scatter(centroids[:,0], centroids[:,1], c = ['red',"gold", "blue"], s = 60␣
↪);
```



1.2 References

- DBSCAN Clustering in ML | Density based clustering
- Clustering Code Implementation
- Understanding Density-based Clustering
- K-Means vs. DBSCAN Clustering — For Beginners
- KMeans vs. DBScan

[ ]: