CHAPTER 5
IMPLEMENTATION AND TESTING

5.1 Implementation

This program is made with Jupyter Notebook and with the python programming language. Below will explain the code.

```
1. import numpy as np
2. import pandas as pd
3. import matplotlib.pyplot as plt
4.
5. from sklearn.cluster import DBSCAN
6. from collections import Counter
7. from pylab import rcParams
8. from sklearn.neighbors import NearestNeighbors
9. from sklearn import metrics
10. rcParams['figure.figsize'] = 12, 7
11. from sklearn.cluster import KMeans
12. from sklearn.metrics import silhouette_samples, silhouette_score
```

This is the code to call all the libraries that will be used.

```
13. X = pd.read_csv ("C:/Users/Lenovo/Desktop/BITCOIN/all_seasons.csv")
14. X.isna().sum()
15. dbscan_X = X[['player_height', '~performance']]
16. dbscan_X = dbscan_X.values.astype('float64', copy=False)
```

Line 13 is used to name and call the CSV file or data file that is used as X. Then line 14 is used to check for empty data on X, line 15 is used to call the required data and name it dbscan_X. On the 16th line, change the dbscan_X data into an array.

```
17. neigh = NearestNeighbors(n_neighbors=4)
18. nbrs = neigh.fit(dbscan_X)
19. distances, indices = nbrs.kneighbors(dbscan_X)
20. distances = np.sort(distances, axis=0)
21. distances = distances[:,2]
22. plt.plot(distances)
23. plt.xlabel("index")
```

```
24. plt.ylabel("distances")
25. plt.show()
```

Lines 17 to 19 are used to run the KNN algorithm. Then next in lines 20 to 25 are used to display the K-dist graph for later use to calculate the optimal value of epsilon with the Elbow Method.

```
26. range_eps = []
27. for i in range_eps :
28.     print("eps value is "+str(i))
29.     model = DBSCAN(eps= i, min_samples=4).fit(dbscan_X)
30.     zero = np.zeros_like(model.labels_, dtype=bool)
31.     zero[model.core_sample_indices_] = True
32.     labels = model.labels_
33.     print(set(labels))
34.     silhouette_avg = metrics.silhouette_score(dbscan_X, labels)
35.     print("For eps value =" +str(i), labels,
36.           "The average is :",silhouette_avg)
```

.

In line 27 it is used to fill in the epsilon value to be calculated. Then on lines 29 to 32 what is done is to run DBSCAN but the epsilon that will be filled depends on the number entered in line 27. Then in line 33 it is instructed to display the labels created from the previous DBSCAN model. In line 34, what is done is to calculate the silhouette score by comparing the array that was created previously with the labels created from the previous model.

```
37. range_n_clusters = []
38.38.
39. for n_clusters in range_n_clusters:
40.     clusterer = KMeans(n_clusters= n_clusters, random_state=42)
41.     cluster_labels = clusterer.fit_predict(dbscan_X)
42.42.
43.     silhouette_avg = silhouette_score(dbscan_X,cluster_labels)
44.     print("for n_clusters = ", n_clusters,
45.           "The average silhouette_score is :", silhouette_avg)
46.46.
47.     sample_silhouette_values = silhouette_samples(dbscan_X,
    cluster_labels)
```

In line 37, it functions almost the same as line 27 which fills in the value of n clusters with values that are filled in itself, then on lines 40 to 41 create a k-means model with values of n clusters which are filled automatically with numbers in line 37. Then in lines, 43 to 47 is to compare the previously created array with labels to get the silhouette value and then print it.

```
48. model = DBSCAN(eps = , min_samples = ).fit(dbscan_X)
49. zero = np.zeros_like(model.labels_, dtype=bool)
50. zero[model.core_sample_indices_] = True
51. labels = model.labels_
```

The 48th line is used to create a DBSCAN model from the results of the previous parameter calculations. Then in lines 49 and 50 to change the value of -1 or noise to false and values other than noise to true. The 51st line is used to create labels.

```
52. metrics.homogeneity_score(X['player_height'],labels)
53. metrics.completeness_score(X['player_height'],labels)
54. metrics.v_measure_score(X['player_height'],labels)
```

The 52nd row is used to calculate homogeneity by comparing the table player_height or height with the label of the DBSCAN model that was previously created. In the 53rd row, the completeness calculation is carried out in the same way as homogeneity as well as the v-measure calculation on the 54th row.

```
55. metrics.homogeneity_score(X['reb'],labels)
56. metrics.completeness_score(X['reb'],labels)
57. metrics.v_measure_score(X['reb'],labels)
```

In rows 55 to 57, the calculation of homogeneity, completeness, and v-measure values in the performance table are the same as in rows 52 to 54.

```
58. unique_labels = set(labels)
```

```
colors = [plt.cm.Spectral(each) for each in np.linspace(0, 1, len(unique_labels))]
for k, col in zip(unique_labels, colors):
    if k == -1:
        col = [0, 0, 0, 1]

    label1 = labels == k

    xy = dbscan_X[label1 & zero]
    plt.plot(
        xy[:, 0],
        xy[:, 1],
        "o",
        markerfacecolor=tuple(col),
        markeredgecolor="k",
        markersize=10,
    )

    xy = dbscan_X[label1 & ~zero]
    plt.plot(
        xy[:, 0],
        xy[:, 1],
        "o",
        markerfacecolor=tuple(col),
        markeredgecolor="k",
        markersize=10,
    )

plt.title("Estimated number of clusters: %d" % n_clusters_)
plt.xlabel('player_height')
plt.ylabel('reb')
plt.show()
```

On line 58, make settings for the labels that will be visualized and on line 59 make a color difference for each cluster with the name color. Then on lines 60 to 64 give the command if the cluster -1 or noise will be separated and black. On lines 66 to 83, the command is executed to identify cluster -1 or noise and other clusters, then followed by lines 86 to 89 are used to display the visualization results.

5.2 Testing

To get the results of the data needed to be able to advise athletes on the optimal position, the data will be processed. In testing, there are 4 steps, namely Processing Data, define parameters, Calculating Cluster Performance with Homogeneity, Completeness, and V-Measure, Evaluation Result. With these 4 processes carried out to get the best results in carrying out this project, a detailed explanation can be seen below

1. Processing Data

After knowing the data type of each attribute that will be used, it is found that the data type is int then the data type is changed to float because it will be more optimal because the data used is decimal then the data can be processed in DBSCAN the next step is to convert it into an array because DBSCAN it is more optimal if used in data with dimension 2, the author will divide it into 3 different arrays, namely the player_height pts, player_height reb and, player_height ast arrays with the following results:

```
array([[213.36,   4.8 ],
       [210.82,   0.3 ],
       [208.28,   4.5 ],
       ...,
       [195.58,   6.1 ],
       [203.2 ,  13.4 ],
       [203.2 ,  12.4 ]])
```

Figure 1.2. Array pts

This is an array view with the contents of height and points. Height is identified as player_height or in the image on the left and points are recognized as pts on the right in the image above. So the image above displays an array form with data type float64 from data player_height and points which will be used for visualization of DBSCAN points.

```
array([[213.36,   4.5 ],
       [210.82,   0.8 ],
       [208.28,   1.6 ],
       ...,
       [195.58,   1.1 ],
       [203.2 ,   4.1 ],
       [203.2 ,   5.7 ]])
```

Figure 1.3. Array reb

Then the picture above is an array of player_height and reb. The player_height pictured above can be seen on the left and the rebound or rebound can be seen on the right. So figure above is an array of data for visualization of DBSCAN rebound.

```
array([[213.36,   0.5 ],
       [210.82,   0.  ],
       [208.28,   0.9 ],
       ...,
       [195.58,   0.6 ],
       [203.2 ,   1.  ],
       [203.2 ,   3.2 ]])
```

Figure 1.4. Array ast

Then for the ast array, it can be seen above, the same as before, on the left, is the player_height or height data and the right is assist or ast. so the picture above is an array image for the assist data which will later be used for visualization of the DBSCAN assist.

2. Define Parameters

In doing clustering with DBSCAN, the most important thing is to recognize epsilon and minimum points. Recognizing epsilon and minimum points in DBSCAN, there are various ways in this project, elbow methods and silhouette calculations will be used in determining DBSCAN parameters. Determination of parameters is very important to get optimal results so that the parameters must be precise. The calculation of parameters in this project can be seen with the explanation below.

Elbow Method

Elbow method is used to determine the epsilon value by plot a k-distance and choose the epsilon value at the "elbow" of the graph.

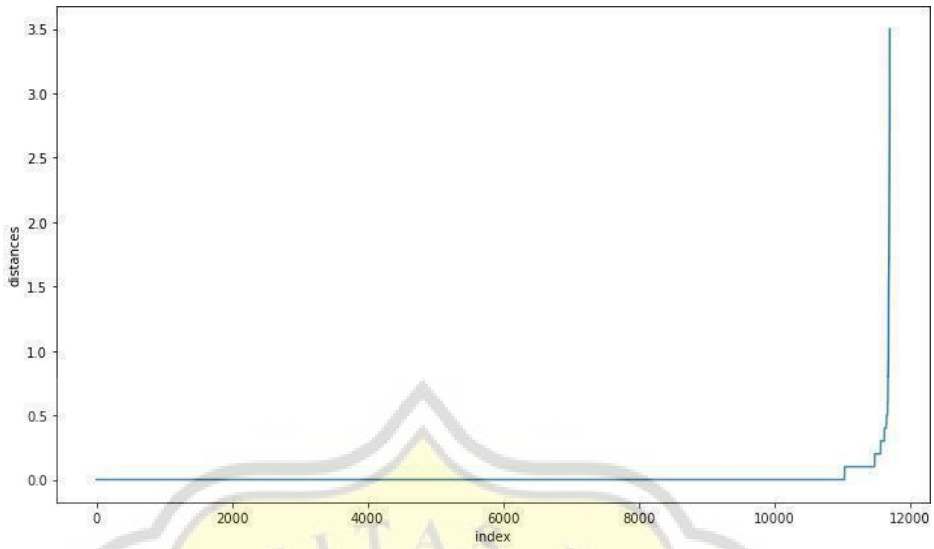


Figure 1.5. Elbow pts

The picture above is the result of the elbow method from player_height and pts or height and points. From these results, it can be seen that the recommended epsilon value in the elbow graph is between 0.1 to 0.5. So from the elbow results, an epsilon between 0.1 and 0.5 will be used.

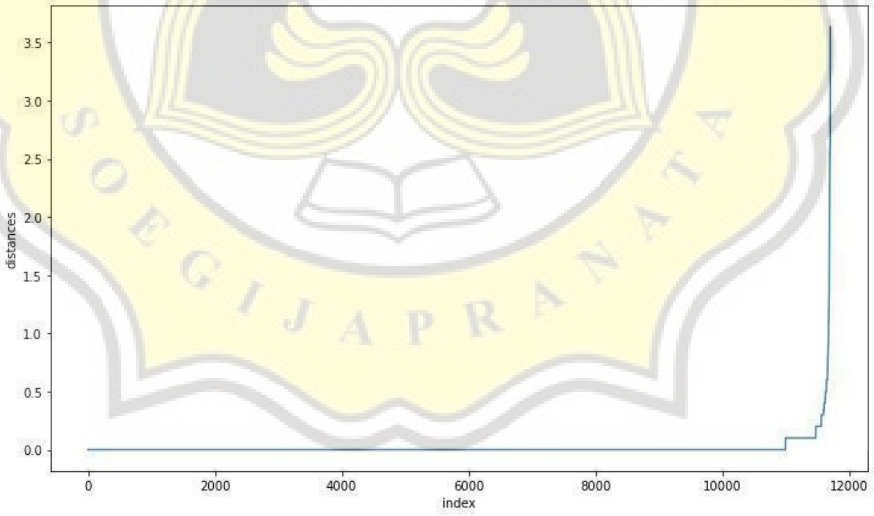


Figure 1.6. Elbow reb

The image above is the result of the elbow method player_height and reb. Then from player_height and reb or height and rebound the result is 0.1 to 0.5. So from the results of the elbow graph above, an epsilon between 0.1 and 0.5 will be used for rebound.

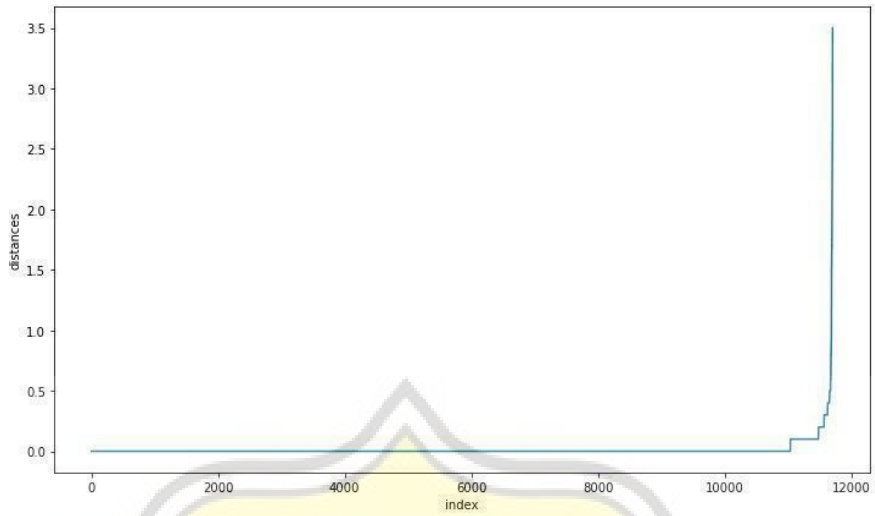


Figure 1.7. Elbow ast

For the last ones are player_height and ast or height and assists, which is the most optimal value in the numbers 0.1 to 0.5. So it can be concluded that the best epsilon according to the elbow method is 0.1 to 0.5 in every comparison of player height and performance.

Silhouette

In the elbow graph, the epsilon value obtained is between 0.0 to 0.5, so to get the epsilon value the next step is to find the silhouette score because epsilon requires a specific value to be used as a DBSCAN parameter. In this process, what will be done is to calculate silhouettes from 0.1 to 0.5 in each 1 array which includes points height, rebound height, assist height. The formula to get the epsilon value is:

$$s(i) = \frac{b(i) - a(i)}{max\ \{a(i), b(i)\}} \qquad (1)$$

where

a(i) = The average distance of that data point with all other points on the same cluster.

b(i) = The average distance of that data point with all member from closest cluster.

From this formula, the following results are obtained :

Table 1.3 : Silhouette Epsilon

Attributes	Epsilon	Sillhouete Score	Number of Cluster
pts	0.1	0.4337728516676882	803
pts	0.2	0.022486297484630338	159
pts	0.3	-0.0848043229195178	102
pts	0.4	-0.004100076428562309	71
pts	0.5	0.025565428625584204	49
reb	0.1	0.6119029681990025	544
reb	0.2	0.27348386404451663	71
reb	0.3	0.2603117726498396	39
reb	0.4	0.34975338931087274	34
reb	0.5	0.3538775380391027	31
ast	0.1	0.5789924892829579	444
ast	0.2	0.42925860574902186	62
ast	0.3	0.4289278801612192	38
ast	0.4	0.5243169308165907	37
ast	0.5	0.5323757032535927	30

With these results, it can be seen that the most optimal silhouette value in all attributes is 0.1. But there is a problem with the number of clusters created because the clusters created have a very large value for pts it produces 803 clusters while reb 544 then assists 444 clusters. The number of clusters created later will make it difficult to read the results of this study, therefore a larger epsilon is used to be able to read the results more easily and the selected epsilon is 2.54 for all attributes. Because the value of 2.54 is the most optimal value to reduce the number of clusters with the result :

Table 1.4 : Silhouette epsilon 2

Attributes	Epsilon	Sillhouete Score	Number of Cluster
pts	2.54	0.20488030533531085	13
reb	2.54	0.3882232972867212	11
ast	2.54	0.4474785739369151	9

After getting the epsilon value, it must know whether the number of clusters is optimal, therefore the number of clusters will be compared with the silhouette to determine the accuracy of the number of clusters with the results :

Table 1.5 : Silhouette Number of Clusters

Attributes	Number of Clusters	Silhouette
pts	803	0.5694014120442495
pts	13	0.34303459693078636
reb	544	0.6075749842986854
reb	11	0.41197972123526666
ast	444	0.7298066783007097
ast	9	0.4821576782297272

Then to determine the minimum points parameter can use the formula

$$minPts = D + 1 \qquad (2)$$

Where

D = Dimension of the data

From this formula, the following results are obtained :

2+1 = 3

According to the existing formula, the minimum points that will be obtained is 3, but because the dimensions of the data are 2, it is also recommended to use 4 minimum points. Because the epsilon value to be used is 2.54 and with MinPts 3 or 4 it will not change the number of clusters created, the MinPts value used is 4. It is different with the epsilon value of 0.1 or the most optimal value for this study, the number of cluster numbers will change with the results :

Table 1.6 : Minimum Points and Number of Cluster Table

Attributes	MinPts	Number of Clusters
pts	3	957
pts	4	803
reb	3	603
reb	4	544
ast	3	499
ast	4	444

From the silhouette calculation above, the parameters in this study will use epsilon 2.54 and minimum points 4.

3. Calculating Cluster Performance with Homogeneity, Completeness, and V-Measure

Homogeneity is used to calculate each cluster that has data points with belonging labels. Homogeneity describes the clustering algorithm's closeness to perfection while completeness calculates where all data points belonging to the same class are clustered into the same cluster then the V-Measure is the harmonic mean of the homogeneity and completeness. The homogeneity, completeness, and V-measure values obtained from this study are :

Table 1.7 : Player Height Performance

Attributes	Epsilon	Minimum Points	Homogeneity Player Height	Completeness Player Height	V-Measure Player Height
player_height (pts)	0.1	3	0.8941839827735589	0.37698735176599135	0.5303707572661326
player_height (pts)	0.1	4	0.8497894445485022	0.3750950025609377	0.5204601538233078

player_height (pts)	2.54	4	0.5559214584063839	0.9956534547405822	0.7134752127489582
player_height (reb)	0.1	3	0.9689238628665702	0.42359827395814986	0.5894835924734811
player_height (reb)	0.1	4	0.9512859719244088	0.42227987932188943	0.5849139668991582
player_height (reb)	2.54	4	0.556710212102431	0.9988450562153692	0.7149437302918358
player_height (ast)	0.1	3	0.9697915783277755	0.4822778128121279 5	0.6441964332191868
player_height (ast)	0.1	4	0.9535133084467091	0.4816732741828626	0.6400308960733964
player_height (ast)	2.54	4	0.5556791365095679	1.0000000000000000	0.7143878496131651

From the results of the performance calculation for height, the best homogeneity was found at epsilon 0.1 and a minimum of points 3 then completeness was best at 2.54 and 4 and for V-measure the best at 2.54 and 4.

Table 1.8 : Performance of player stat

Attributes	**Epsilon**	**Minimum Points**	**Homogeneity**	**Completeness**	**V-Measure**
pts	0.1	3	0.7578480154126559	0.6337101793333428	0.6902420661497859
pts	0.1	4	0.7190420837373666	0.6294959312118262	0.6712959680262313
pts	2.54	4	0.0165719613679954	0.0588677384399091 26	0.0258631434041102 57
reb	0.1	3	0.8314144988189736	0.6048475761253169	0.7002608411638341
reb	0.1	4	0.8158345027099769	0.6026361577051073	0.6932133088921649
reb	2.54	4	0.0326782130219536 2	0.0975643278535340 1	0.0489583183422299 05
ast	0.1	3	0.7576809787417903	0.5503331486819 18	0.6375725613127027

ast	0.1	4	0.7421941490982191	0.5476003679912512	0.6302178894112105
ast	2.54	4	0.040971028361628636	0.10768945711421035	0.059358716441051666

Then for the calculation of player performance with cluster performance, it can be seen that homogeneity, completeness, and V-measure are best at epsilon 0.1 and at minimum points 3. so that according to the calculation of the most optimal performance when using epsilon 0.1 and at least points 3 because of the calculation of height and performance. epsilon 0.1 and minimum points 3 players get the highest total score and if epsilon 0.1 and minimum points 4 are also good but the results are slightly worse than 0.1 and 3 while for epsilon 2.54 and at minimum points 4 the height performance is very good even outperforming 0.1 and 3 but in player performance, the results are not good.

4. Evaluation Result

For the visualization results of each epsilon and the minimum points aimed are:

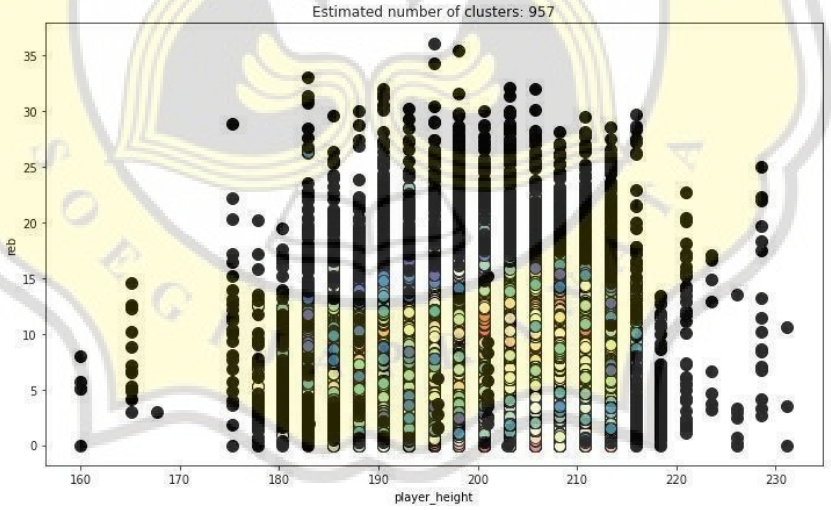


Figure 1.8. Points

The picture above is a visualization with epsilon 0.1 and a minimum of 3 points. From the visualization results obtained above, it can be seen that the results have a lot of noise and a lot of clusters with different density levels. So from the results above, no conclusions can be drawn for the optimal height in points.

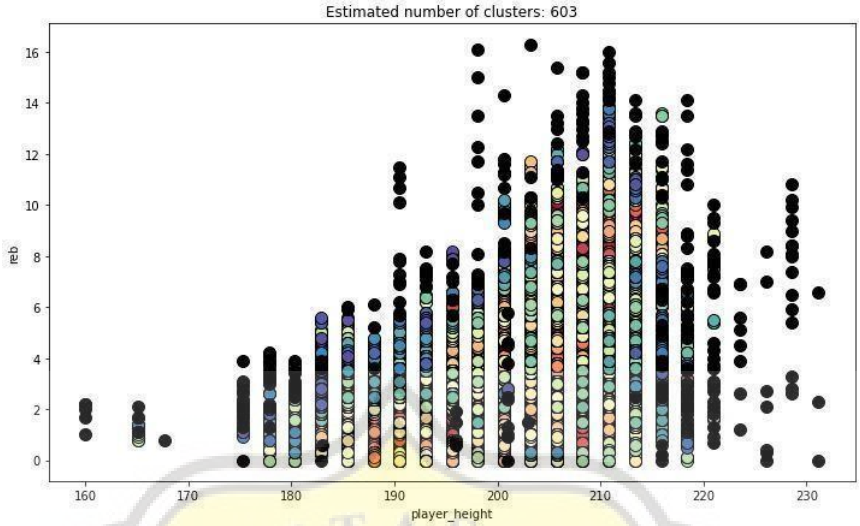


Figure 1.9. Rebound

Next is the result of the rebound visualization with epsilon 0.1 and minimum points 3. From the results of the visualization, it is still the same as before, namely points that have a lot of noise, lots of clusters, and different densities of each cluster so that it cannot be concluded that the optimal height for rebounding cannot be concluded.

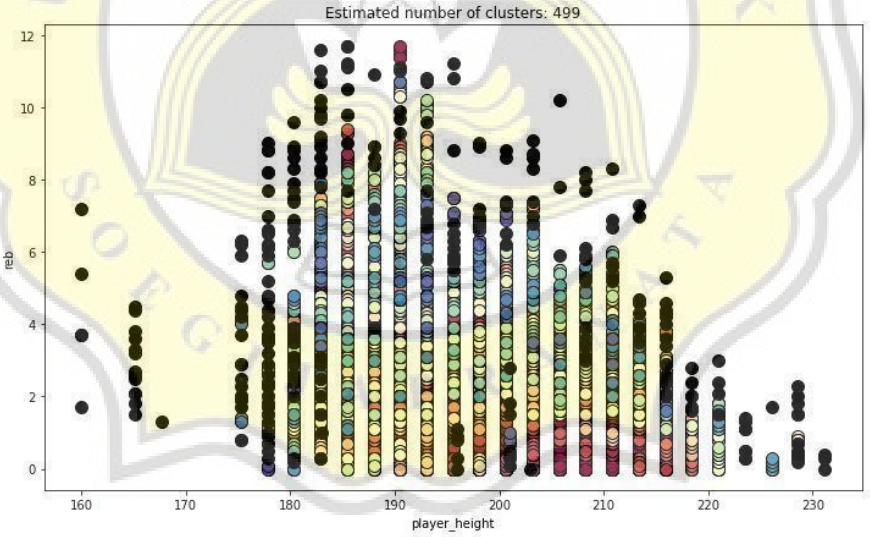


Figure 1.10. Assist

Then for visualization assists with epsilon 0.1 and minimum points 3 is also the same as a lot of noise, a large number of clusters, and varying densities so that it cannot be concluded that the optimal height for assists cannot be drawn. So for epsilon 0.1 and minimum points 3 will not be

used because from the visualization results it can not be concluded that the optimal height for each player's performance.

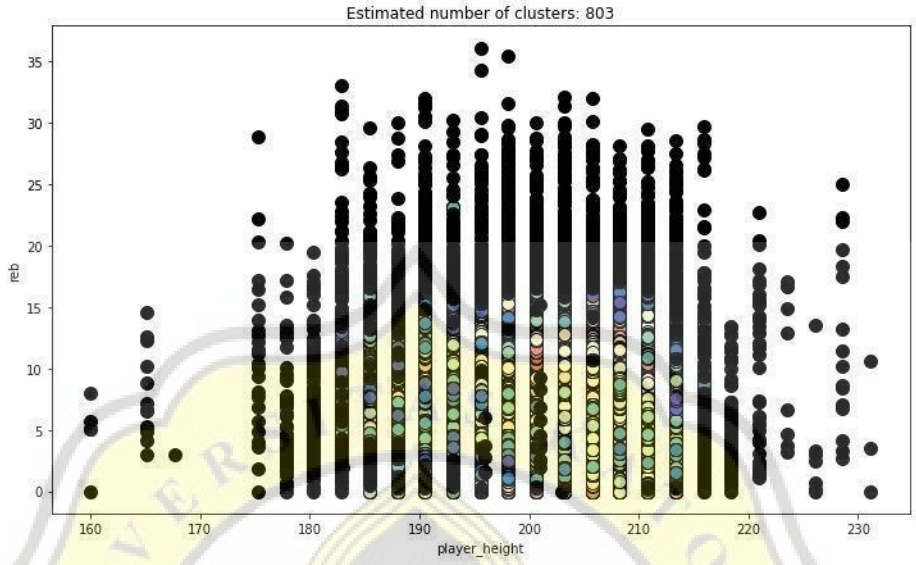


Figure 1.11. Points

The next image is a visualization of epsilon 0.1 and minimum points 4 of points. From the picture above, it can be seen that there is still a lot of noise even though it has reduced from epsilon 0.1 and minimum points 3 and also the number of clusters is still large and the cluster density is still diverse, so from the results of visualization of points with epsilon 0.1 and minimum points 4, it cannot be concluded that high optimal body.

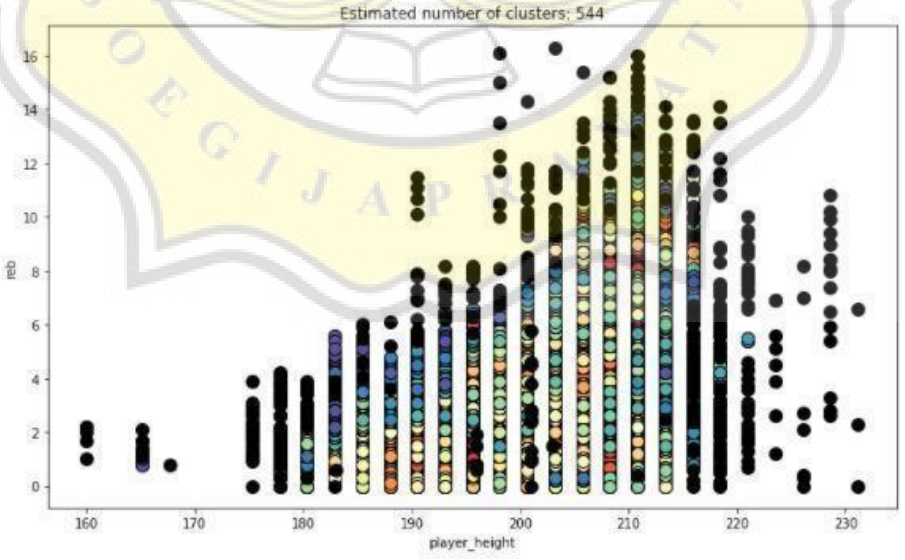


Figure 1.12. Rebound

Then next is the visualization of the rebound with epsilon 0.1 and minimum points 4 the results can be seen that the noise has decreased from epsilon 0.1 and minimum points 3 but is still the same in a large number of clusters and varying cluster densities so it is still not possible to conclude for optimal height.

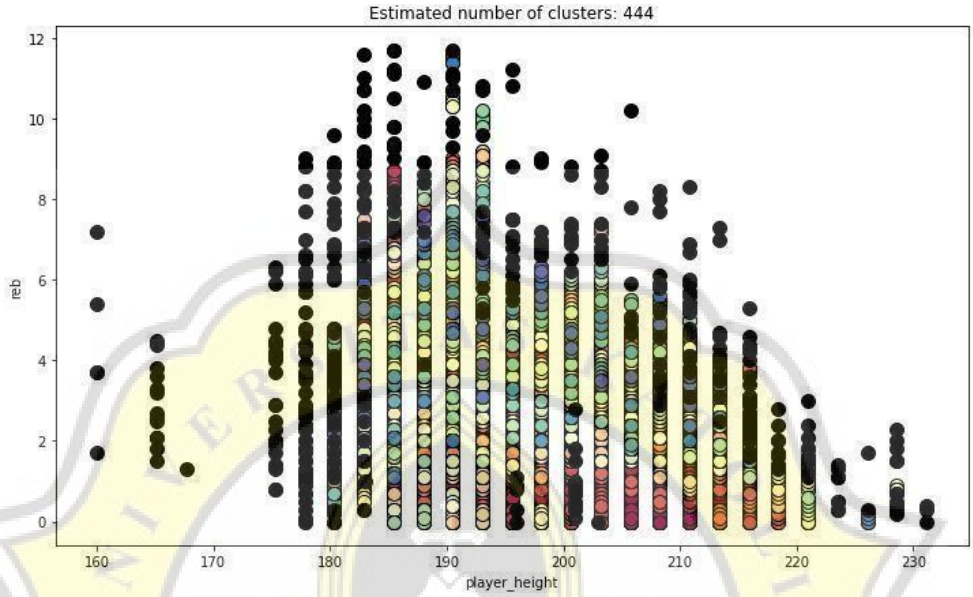
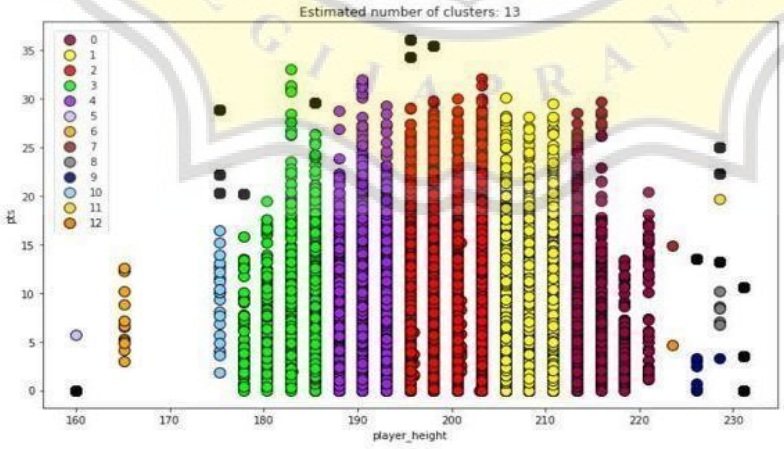


Figure 1.13. Assist

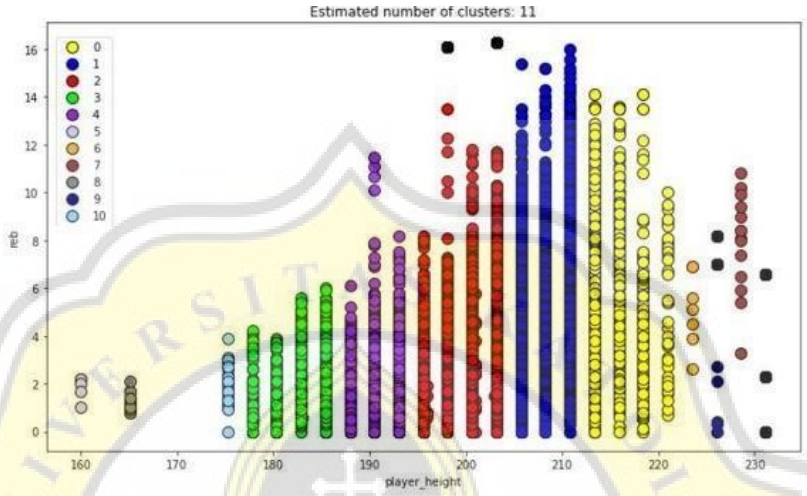
Then on the assist epsilon 0.1 and minimum points 4 it can be seen that the noise has decreased but the number of clusters and the density of clusters is still the same as epsilon 0.1 and minimum points 3. So for epsilon 0.1 and minimum points 4 will not be used because no conclusions can be drawn.




```
Counter({2: 4134, 1: 3581, 4: 1984, 0: 1020, 3: 898, 10: 22, -1: 16, 6: 14, 9: 9, 8: 6, 5: 4, 12: 4, 7: 4, 11: 4})
number of clusters 13
```

Figure 1.14. Points

Then with that, the author uses epsilon 2.54 and minimum points 4 to be able to overcome the above problem. It can be seen from the results above that with epsilon 2.54 and minimum points 4 the visualization results of points noise is reduced then cluster density and the number of clusters also improve so it can be concluded that the author chooses to use epsilon 2.54 and minimum points 4 in making optimal height decisions for points.




```
Counter({2: 4135, 1: 3581, 4: 1984, 0: 1020, 3: 900, 10: 25, 7: 16, 8: 14, 6: 8, -1: 7, 5: 5, 9: 5})
number of clusters 11
```

Figure 1.15. Rebound

Next is the rebound visualization with epsilon 2.54 and the minimum points 4 are the same as the points the author chooses to use epsilon 2.54 and minimum points 4 because the noise produced is less and the number of clusters and cluster density is improving so that conclusions can be drawn for the optimal height in rebounding.

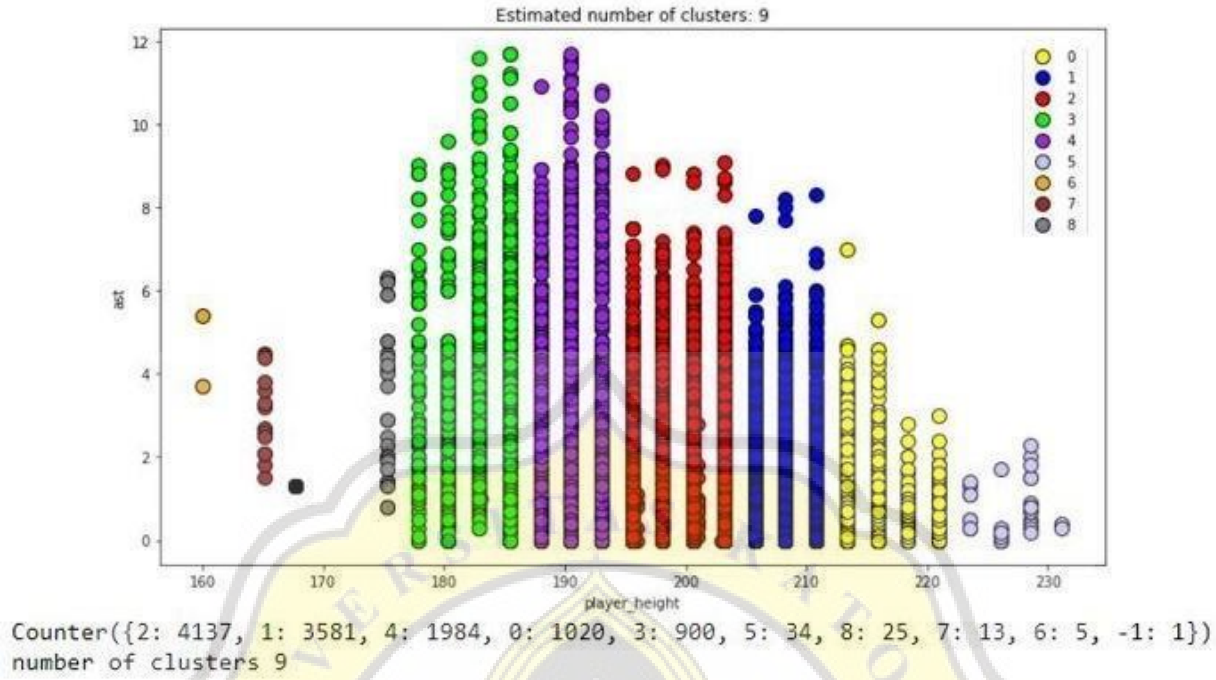


Figure 1.16. Assist

The assists are also the same as points and rebounds, the author uses epsilon 2.54 and minimum points 4 because the number of clusters and the density of clusters is improving from the previous epsilon and minimum points and noise is also reduced. So the visualization of 2.54 and minimum points 4 can be concluded so that the author chooses to use it

Table 1.9 : Result

Position	Height
Point Guard	187-205 cm
Shooting Guard	195-205 cm
Small Forward	195-205 cm
Power Forward	197-210 cm
Center	197-210 cm

Based on the tests carried out on this project in determining the parameters, it is estimated that the usage method is not appropriate. The drawback is in determining the elbow method using the K-dist Graph, in the elbow method the Epsilon results are 0.1 to 0.5 and the performance results achieved are very satisfactory with the largest being at 0.1 then for MinPts using D + 1 or 4

formulas for 2-dimensional data. Then the results obtained are not satisfactory because there is a lot of noise in the middle of the data. then after the elbow method is used, the silhouette comparison is quite satisfactory by getting a value of 2.54 with MinPts 4 can greatly reduce the amount of noise previously obtained and the data is grouped better and more legible. Then for the classification of the position of basketball athletes according to height using DBSCAN taken from the highest cluster and the densest cluster for cluster points and height, it was found that the acquisition of points in many heights was quite average but the cluster results obtained showed that cluster 2 was the most numerous so that for The ratio of points and heights for shooting guards and small forwards is 195-205 cm. For rebounds and height for the power forward and center, it was found that the densest cluster was cluster 2, which means that players with a height of 195-205 are the players who do the most rebounds, but because of the distance that is quite far with the highest cluster, namely, cluster 1 which is also the second-largest cluster. After cluster 2, for the comparison of height and rebound for the power forward and center, 2 clusters were taken, namely clusters 1 and 2 which ranged from 195-210 cm in height, then the last one for the comparison of height and assists for the densest point guard cluster was also in cluster 2. and the same as the rebound, the distance between the densest cluster and the highest value is also quite far, so 2 clusters are also taken for the comparison of height and assists so that they are 187-205 cm tall.