Clustering and Performance Prediction in Sports: A Comparison of Unsupervised and Supervised Methods

Practical Machine Learning

Popa Andrei Ionut

412

Table of contents

1. Introduction

1.1 Clustering

Clustering is a powerful technique used to group similar data points based on their characteristics. In this project, clustering was chosen to analyze and split football players based on their performance metrics. The motivation behind this approach is to uncover meaningful patterns in player data without relying on predefined labels. Additionally, comparing clustering results with supervised models and random methods allows us to assess the effectiveness of these techniques in real-world scenarios.

1.2Project Goals

This project aims to achieve several objectives:

1. Utilize Two Clustering Methods: We implement and analyze two distinct clustering techniques, BIRCH and OPTICS, to group players based on their attributes.
2. Compare with Reference Methods: The clustering results are compared against a supervised model (Random Forest) and a random baseline to evaluate their accuracy and relevance.
3. Interpret Cluster Results: We interpret the clusters from a sports perspective, identifying patterns and linking them to player roles and performance levels.

2. Dataset

The dataset contains information about over 16,000 football players, providing a rich source of data for analysis and clustering. It includes 100 attributes per player, covering a wide range of features:

Demographics: Age, height, weight, nationality, and preferred foot.

Skills and Ratings: Overall rating, potential, skill moves, weak foot rating, and international reputation.

Technical Skills: Passing, shooting, dribbling, ball control, and defensive abilities.

Physical Attributes: Speed, stamina, strength, and agility.

Financial Information: Market value, wage, and release clause value.

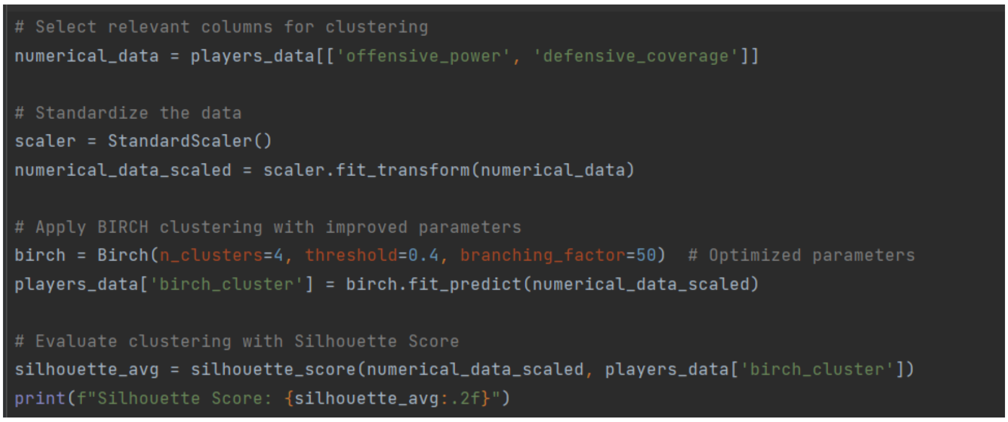
3. Birch

BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) is a clustering algorithm designed to handle large datasets efficiently. It builds a clustering feature tree (CF tree) to summarize the data and improves the clusters step by step as it processes more data. BIRCH is especially useful when working with high-dimensional data and can adapt dynamically to memory constraints by controlling the tree’s size.

3.1 Objective

The goal was to group football players into clusters based on two important performance metrics: ’offensive_power’ and ’defensive_coverage’. We aimed to identify roles such as ‚Forward’, ‚Midfielder’, ‚Defender’, ‚Goalkeeper’ and we evaluate how these clusters match with actual players roles.

3.2Methodology



```
# Select relevant columns for clustering
numerical_data = players_data[['offensive_power', 'defensive_coverage']]

# Standardize the data
scaler = StandardScaler()
numerical_data_scaled = scaler.fit_transform(numerical_data)

# Apply BIRCH clustering with improved parameters
birch = Birch(n_clusters=4, threshold=0.4, branching_factor=50)  # Optimized parameters
players_data['birch_cluster'] = birch.fit_predict(numerical_data_scaled)

# Evaluate clustering with Silhouette Score
silhouette_avg = silhouette_score(numerical_data_scaled, players_data['birch_cluster'])
print(f"Silhouette Score: {silhouette_avg:.2f}")
```

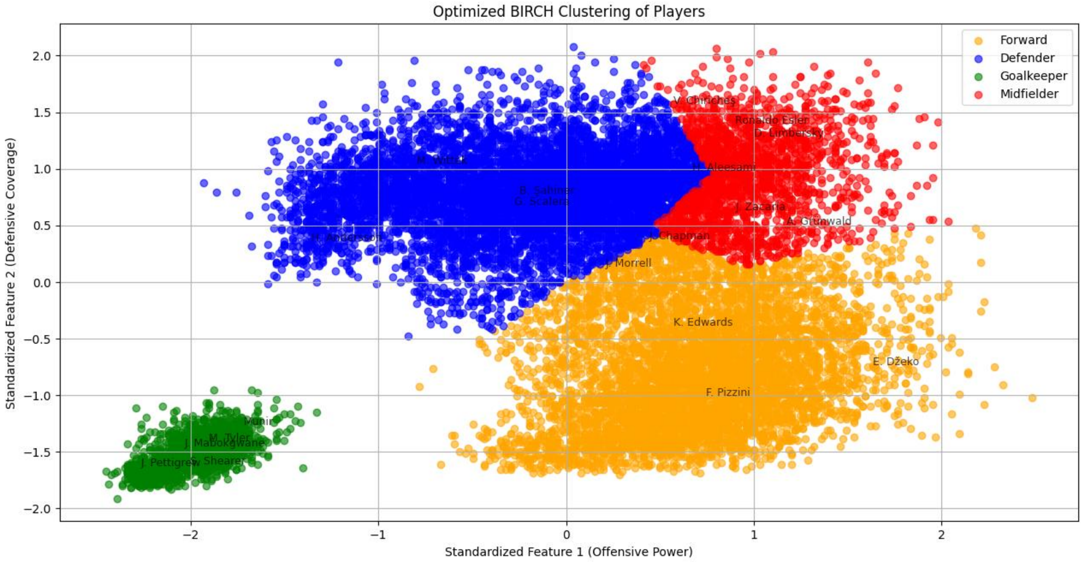
1. The features selected was ’offensive_power’ and ’defensive_coverage’, two numerical features because they sum key aspects about the playstyle of the player.
2. To ensure fair comparison and improve clustering accuracy, the data was standardized using StandardScaler from scikit-learn.
3. Optimized parameters were used for the BIRCH algorithm:

   n_clusters=4: Predefined number of clusters. It matches the numer of player roles.

   threshold=0.4: Maximum radius of sub-clusters. It was chosen to strike a balance between capturing meaningful similarities among players and avoiding overly fragmented clusters.

   branching_factor=50: Controls the number of CF tree nodes. It was chosen to strike a balance between capturing meaningful similarities among players and avoiding overly fragmented clusters.

3.3 Results



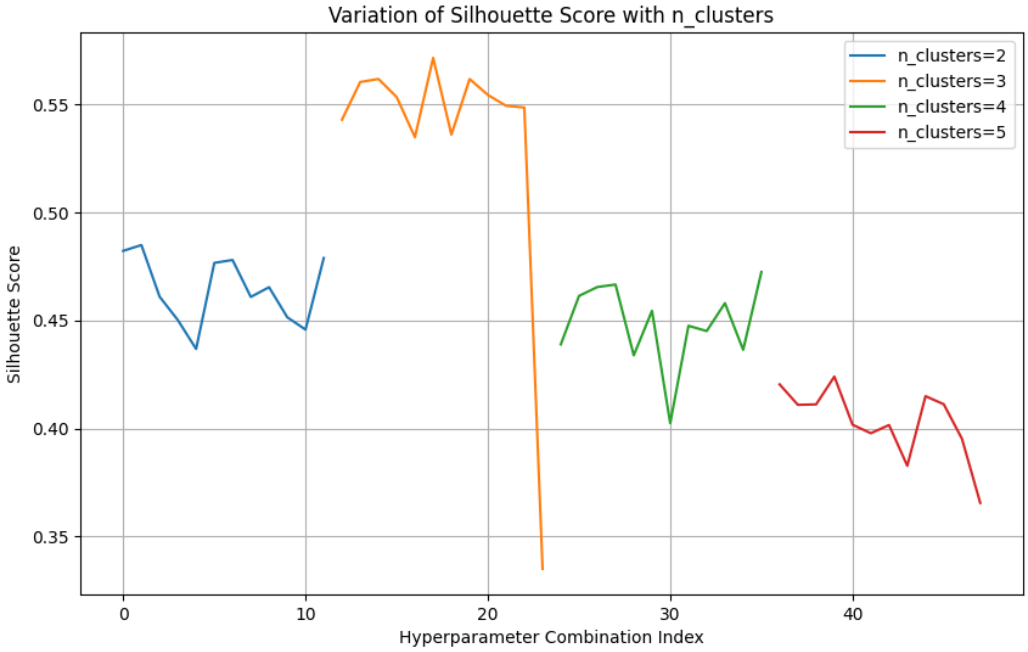
This plot shows the results of the BIRCH clustering algorithm applied to the football players' dataset. The two features used are Standardized Offensive Power (x-axis) and Standardized Defensive Coverage (y-axis). Each color represents a distinct cluster, which has been mapped to a football role.

We can observe a fairly accurate representation, as we have the forwards in yellow, where offensive power is dominant; the defenders in blue, with high defensive attributes and lower offensive performance; the midfielders in between, with balanced attributes; and a special category, far from the other three, represented by the goalkeepers.

Silhouette Score: 0.45

A score of 0.45 suggests that clusters are reasonably well-formed, but there is room for improvement.
The clusters are somewhat distinct, but some points may be near cluster boundaries or in overlapping regions.

In this case, the clusters represent football roles (e.g., Forward, Defender). While the score shows that the clustering has captured some distinctions between roles, there may be significant overlap in player attributes between roles (e.g., some midfielders might have similar characteristics to forwards).



The plot shows the relationship between the number of clusters (n_clusters) and the quality of clustering (Silhouette Score).

Relevance:

1. Helps identify the optimal number of clusters. For example:

The score peaks for n_clusters=3 and starts to decline for higher values.

This suggests that using 3 or 4 clusters might result in more meaningful groupings for the dataset.

2. Indicates that adding more clusters doesn’t always improve clustering quality.

3.4 Comparison with Supervised Method

The BIRCH clustering method predicts the correct role for 66.37% of the players in the dataset. This was verified by mapping the positions of players (which were already included in the dataset) to one of the four defined roles. If we were to use a random algorithm, we would achieve an accuracy of 25%, as the players are evenly distributed across four roles.

Another approach to predicting player roles based on their attributes involved training a Random Forest model. This supervised model aimed to achieve higher accuracy than the clustering method.

To train the model, the dataset was split into three files: train (80%), test (10%), and validation (10%). After training, the model achieved an accuracy of 88%.

The Random Forest model’s 88% accuracy demonstrates its ability to learn from labeled data and effectively classify players into their respective roles. The high accuracy reflects the model’s ability to capture complex relationships between the players' attributes and their roles.

Unlike clustering, which relies on unsupervised grouping, Random Forest benefits from knowing the true labels during training, enabling it to make more precise predictions.

In conclusion, while supervised learning (Random Forest) achieves higher accuracy, the unsupervised clustering approach (BIRCH) remains valuable for situations where labeled data is unavailable. Both methods significantly outperform random assignment, underlining their effectiveness in analyzing player roles.

3.5 Comparison with other clustering method

We attempted to perform the same process using a different clustering method to compare the results with those obtained previously. We chose to use K-Means to group the players into roles based on the attributes mentioned: offensive_power and defensive_coverage.

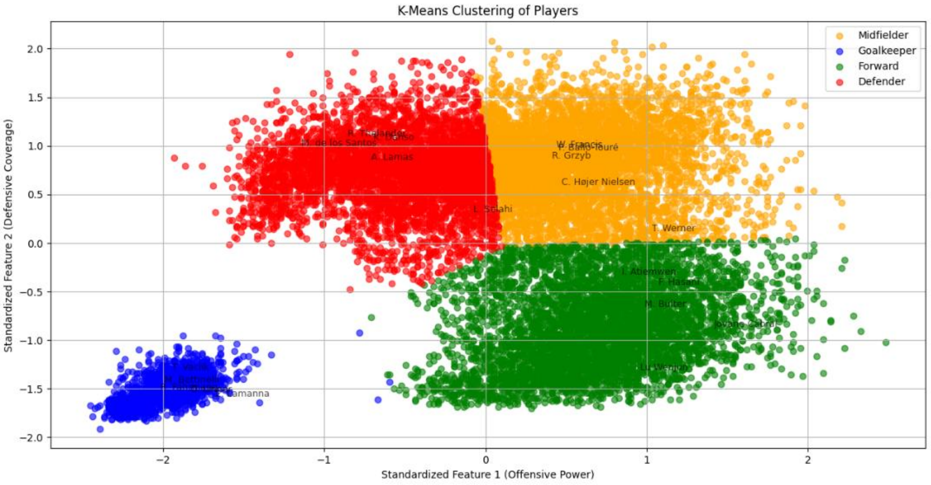
K-Means is a used clustering algorithm that segments data into a predefined number of clusters (k) based on their similarity. The algorithm works to minimize the intra-cluster variance (distance within the same cluster) and maximize the inter-cluster variance (distance between clusters).

The K-Means algorithm was applied to group the players into 4 clusters (n_clusters=4), representing: Forward, Defender, Midfielder, Goalkeeper

Key parameters used:

1. random_state=42 to ensure reproducibility.
2. n_init=10 to run the clustering process 10 times with different initial centroids and choose the best solution.
3. max_iter=300 to allow up to 300 iterations for convergence.

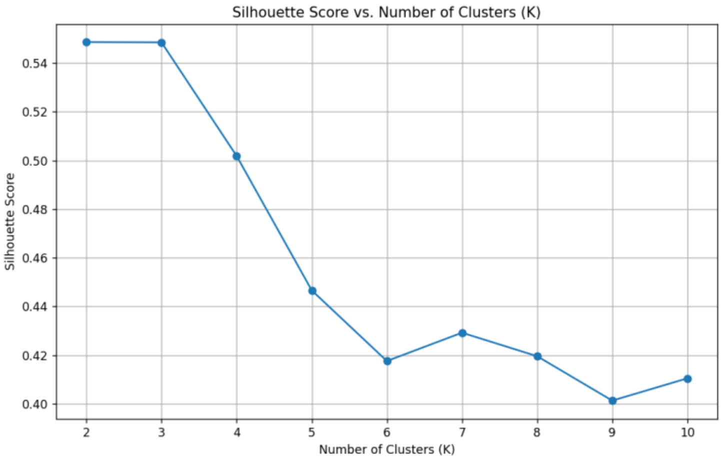
Results



The graph generated by the K-Means clustering method is quite similar to the one obtained using the BIRCH method. The only noticeable difference between the two graphs is the size of the midfielder category, which is proportionally larger in K-Means.

The Silhouette Score for the K-Means clustering method was 0.50, which is slightly higher compared to the 0.45 obtained with the BIRCH algorithm. This indicates that the clusters formed by K-Means are somewhat better defined, with clearer boundaries and greater separation between groups.

In the output file containing predicted_role and actual_role, which verifies the role assigned by clustering against the real role, the K-Means method achieves a result quite similar to that of the BIRCH method. With K-Means, we achieve 67.15%, while BIRCH achieved 66.37%, the difference being too small to be considered an improvement.



The graph of the Silhouette Score as a function of the number of clusters indicates that the K-Means method is more efficient for a smaller number of clusters. The interpretation is correct, as the Silhouette Score is highest when the number of clusters is lower (e.g., 2 or 3), suggesting better-defined and more cohesive clusters.

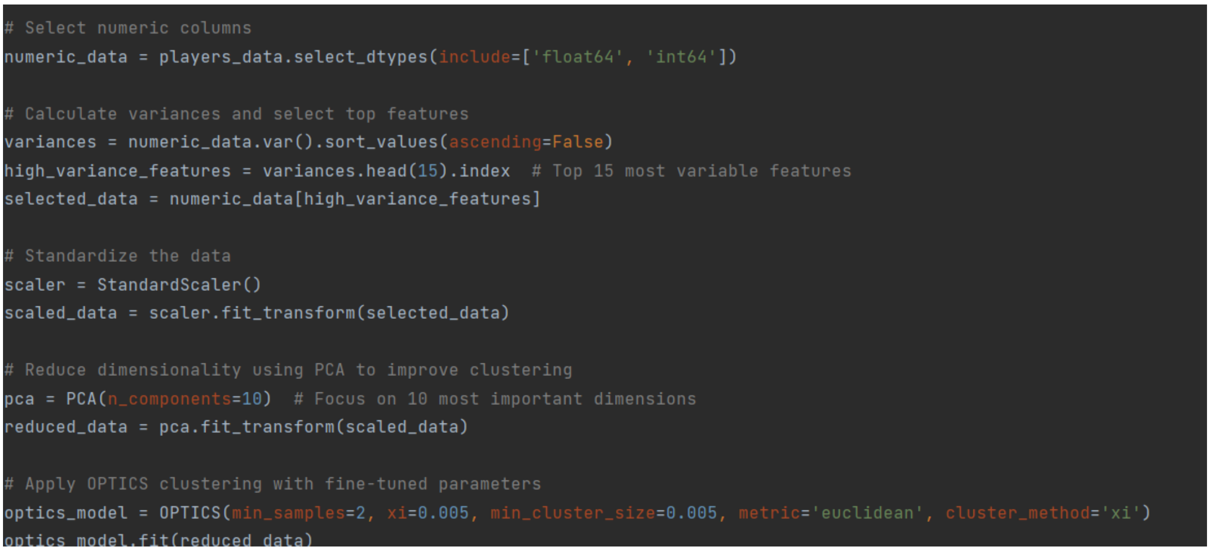
4. Optics

OPTICS (Ordering Points To Identify the Clustering Structure) is a density-based clustering algorithm that identifies clusters based on density connectivity between data points. Unlike traditional methods such as DBSCAN, OPTICS can handle varying densities and does not require a predefined number of clusters, making it well-suited for complex datasets.

4.1 Objective

The objective of clustering was to divide the players into categories based on their attributes without specifying the number of categories in which the players would be grouped.

4.2 Methodology



```
# Select numeric columns
numeric_data = players_data.select_dtypes(include=['float64', 'int64'])

# Calculate variances and select top features
variances = numeric_data.var().sort_values(ascending=False)
high_variance_features = variances.head(15).index  # Top 15 most variable features
selected_data = numeric_data[high_variance_features]

# Standardize the data
scaler = StandardScaler()
scaled_data = scaler.fit_transform(selected_data)

# Reduce dimensionality using PCA to improve clustering
pca = PCA(n_components=10)  # Focus on 10 most important dimensions
reduced_data = pca.fit_transform(scaled_data)

# Apply OPTICS clustering with fine-tuned parameters
optics_model = OPTICS(min_samples=2, xi=0.005, min_cluster_size=0.005, metric='euclidean', cluster_method='xi')
optics_model.fit(reduced_data)
```

1. Variance analysis was performed on the numerical features to select the top 15 attributes with the highest variability. These high-variance features captured the most meaningful variations in player performance.

2. StandardScaler was used to normalize the data, ensuring all features had a mean of 0 and a standard deviation of 1. This step ensured no single feature dominated the clustering process due to its scale.

3. Principal Component Analysis (PCA) was applied to reduce the dataset to 10 principal components, retaining most of the data's variability while improving computational efficiency.

4. OPTICS Parameters:

The following parameters were fine-tuned for the OPTICS algorithm:

- `min_samples=2`: Minimum number of points required to define a cluster.

In this project, using min_samples=2 enables the algorithm to detect clusters with very few points, which can be important for identifying rare or small groups (e.g., top-performing elite players).

- `xi=0.005`: Sensitivity parameter controlling the steepness of clusters.

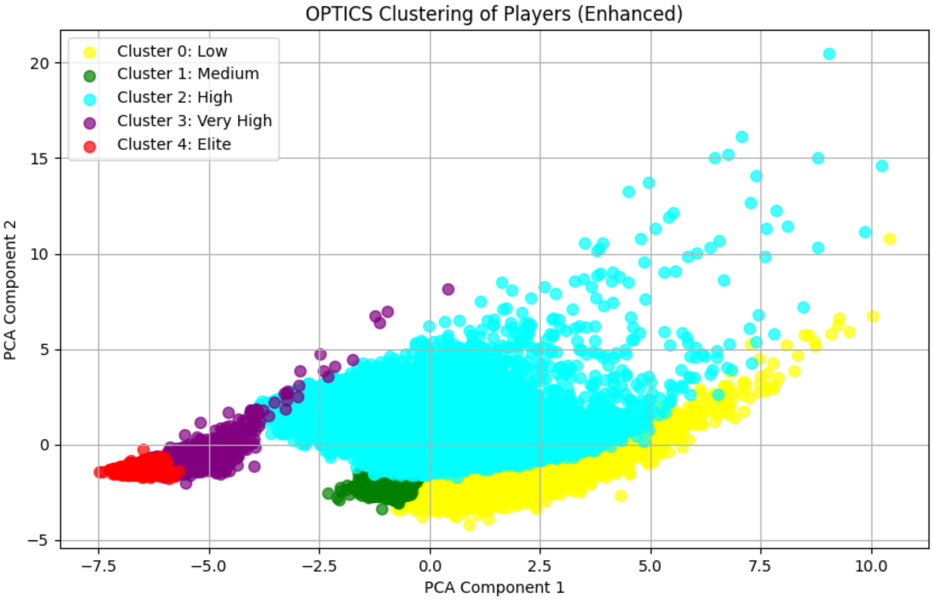
A smaller value for xi (e.g., 0.005) makes the algorithm more sensitive to subtle changes in density, leading to finer-grained clustering. In this case, using xi=0.005 allows the algorithm to better distinguish between performance levels (e.g., Low vs. Medium) by splitting clusters when subtle density differences are detected.

- `min_cluster_size=0.005`: Minimum proportion of the dataset required to form a cluster.

Setting this value ensures that clusters are large enough to be statistically significant but small enough to capture meaningful variations. For example, small groups of elite players or rare profiles are retained without being treated as noise.

- `metric='euclidean'`: Distance metric used to calculate proximity between points.

4.3 Results

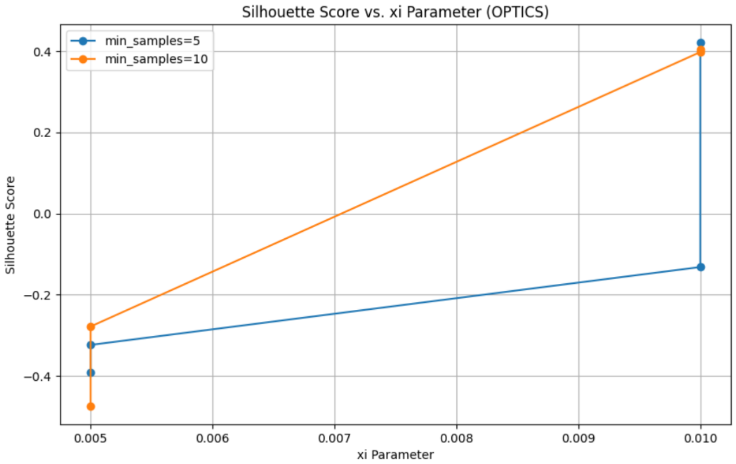


On the graph, we can observe how players were divided into performance categories based on the attributes they have in the dataset. The dominance of players in the 'High' category is easily noticeable.

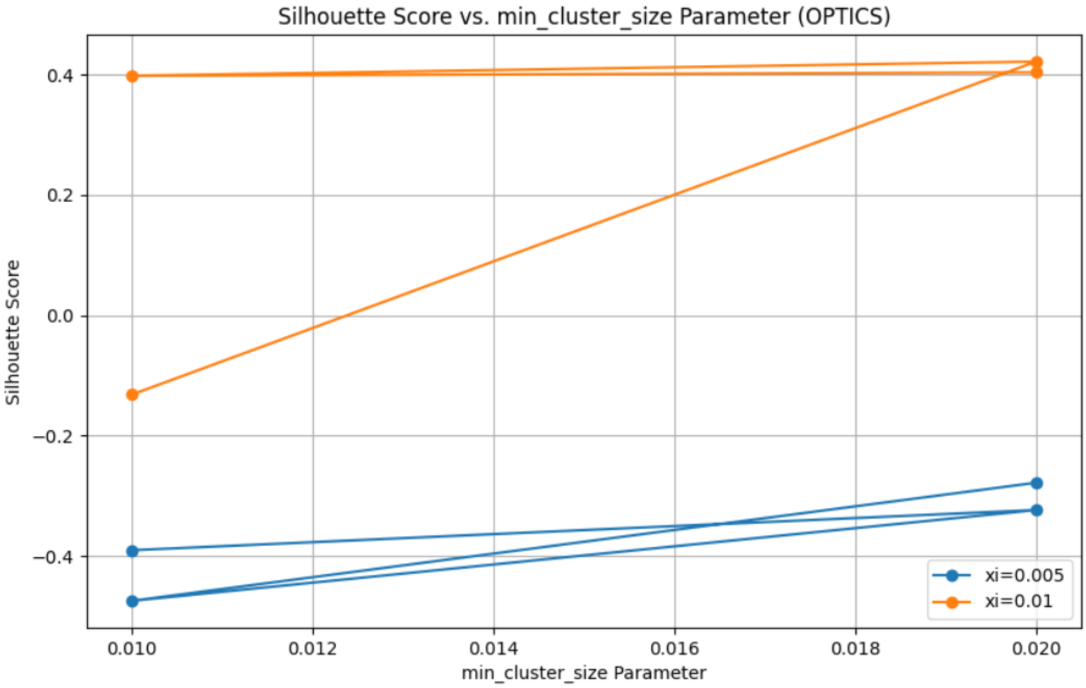
Average Silhouette Score: 0.13

The average silhouette score of 0.13 shows that the clusters are not very well-separated. This means many players in the dataset have attributes that overlap, making it harder for the algorithm to group them clearly into performance categories like Medium or High. The low score suggests the clusters are not very compact, and players near cluster edges could belong to multiple groups. To improve this, we could try selecting better features, fine-tuning the algorithm’s settings, or adjusting how the data was reduced with PCA.

These parameters (xi=0.01, min_samples=10, min_cluster_size=0.02) achieve the highest Silhouette Score of 0.49, but they divide the players into only two categories, which limits the diversity of the groups and does not fully capture the complexity of the data.



The graph shows that a higher value for min_samples (e.g., min_samples=10) combined with a larger xi parameter (e.g., xi=0.01) significantly improves the Silhouette Score, indicating better-defined and more cohesive clusters.



The graph indicates that increasing the min_cluster_size parameter leads to a slight improvement in the Silhouette Score, especially for xi=0.01. This suggests that larger clusters contribute to better-defined groups, but the impact is more pronounced for higher xi values.

4.4 Comparation with a supervised method

The Optics clustering method predicts the correct role for only 32.20% of the players in the dataset. This was verified by mapping the overall_rating of players (which were already included in the dataset) to one of the five defined categories. If we were to use a random algorithm, we would achieve an accuracy of 19,51%, as the players are evenly distributed across five roles.

To improve upon the accuracy achieved by the OPTICS clustering method, we trained a Random Forest Classifier, like we did as well for Birch method. The goal was to predict the player's performance category (Low, Medium, High, Very High, Elite) based on their numerical attributes. This approach leveraged the labeled data available in the dataset, as supervised methods are designed to learn from existing mappings. The Random Forest model achieved 90%, far outperforming both methods by a supervised approach. Clustering methods like OPTICS are helpful for exploratory analysis but struggle in scenarios with overlapping data or complex relationships between features. The Random Forest model demonstrates the power of supervised learning, particularly for structured datasets with clear labels, by effectively capturing patterns and relationships within the data. This highlights the importance of selecting the appropriate method based on the task and the dataset. While clustering methods can provide insights when labels are unavailable, supervised learning is the preferred choice for accurate predictions in labeled datasets.

5. Conclusion

In this project, we used clustering and supervised learning methods to analyze and predict the roles or the performance score of football players based on their performance. We tried three clustering techniques: BIRCH, K-Means, and OPTICS. Each method grouped players into categories without using predefined labels. BIRCH worked well with larger data, K-Means created slightly better clusters (with a Silhouette Score of 0.50), and OPTICS allowed us to find clusters of varying sizes.

However, the clustering methods were not as accurate as the supervised Random Forest model. BIRCH and K-Means predicted roles with around 66–67% accuracy, while OPTICS only achieved 32.20%. On the other hand, the Random Forest model, which used labeled data to learn, achieved a much higher accuracy of 90%, showing its strength in this type of task.

In conclusion, clustering methods are useful for finding patterns in data when labels aren't available, but supervised learning like Random Forest is much better for making accurate predictions when we have labeled data. This project shows the importance of choosing the right method based on the type of data and the goal of the analysis.