Basketball event classification

Manuel
Recall is just for fun

June 11, 2025

The data we have selected can be found in https://github.com/linouk23/NBA-Player-Movements. The set consists of basketball players positions on the court during a game. The repository has every game of the NBA 2015-2016 regular season. We will only focus on the game between the Phoenix suns and Golden state warriors played 27-11-2015. The main objective of the project is to classify every play made in the game based on the ball trajectory.

Data cleaning

First of all, data comes in .json format. Each game is a nested dictionary that looks like this

```
{
  "gameid": "...",
  "gamedate": "...",
  "events": [
    {
      "eventId": "...",
      "visitor": { ... },    \\ Visitor players names, numbers and Id
      "home": { ... },       \\ home players names, numbers and Id
      "moments": [ ... ]     \\ players position on the court recorded each
          0.04 seconds
    },
    ...
  ]
}
```

where the relevant information to our analysis are the events, more specifically, the moments of each event. The structure of the moments is the following

```
"moments":[
  unkown,
  unkown,
  unkown,
  shot_clock,           \\time left in possession
  null,
  [
    [gameId, playerId, x, y, z],     \\ball Id=-1
    [gameId, playerId, x, y, z],
    ...
    (11 entries total, one for each player and the ball)
  ]
]
```

Now that we know what we are dealing with, we can freely select the information we want and convert it to a more manageable, usable, format. As we said in the intro we will only focus on the basketball trajectories. Now we can convert the .json file to a more manageable, usable, format. During this process, we note there are in fact some empty events. In particular, 28 trajectories out of 486 are empty. Before moving on to the description, there is an important remark to make. As we mentioned before, the whole game is recorded in intervals of 0.04 seconds, but there is no apparent criteria to what they consider an "event". Thus events do not overlap perfectly, i.e, for any two trajectories $f_{n+1}(0) \neq f_n(T)$, where T denotes the last point in the domain. This means that we have a lot of repeated data. However, we can assume each trajectory f is independent of the others, and treat them as two completely different.

Exploratory analysis

We have a function $f : \mathbb{R} \to \mathbb{R}^3$ for each event. In particular, the range of each f is contained in $[0, 94] \times [0, 50] \times [0, \infty)$, where those intervals are the official measurements of a basketball court in feet. Note, that obviously the height would be bounded. Regarding the domain, we have a clear problem, trajectories are not of the same length. Here in figure 1 we see three different events, each one of them with three really different lengths. Note the time is recorded in seconds.

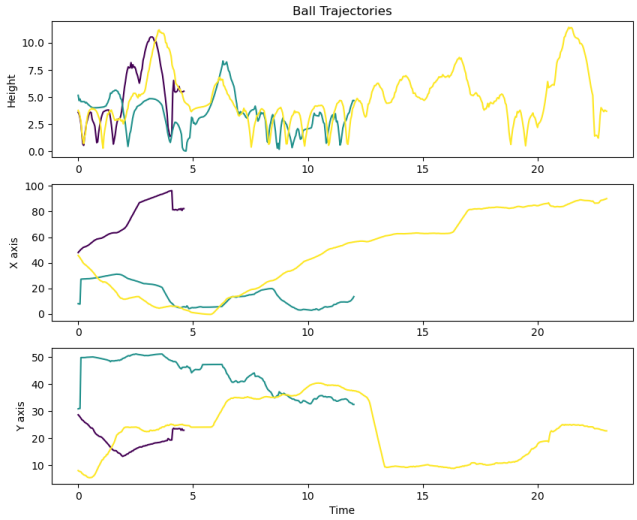
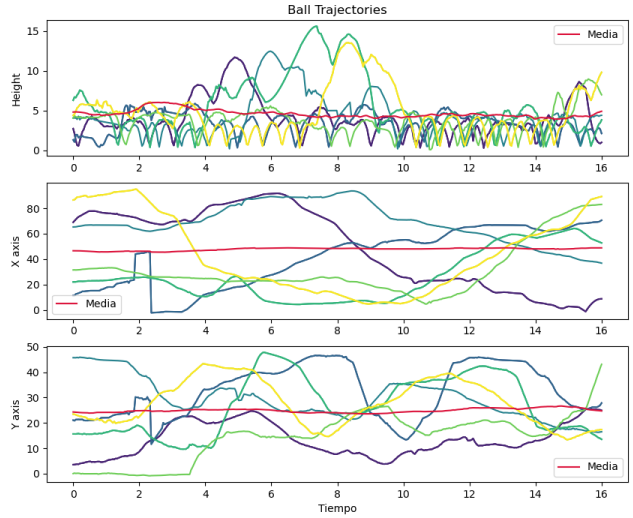


Figure 1: 1-dimensional projections

In order to make some descriptive analysis the easiest option is to bound the length and select only trajectories that have greater or equal length.



The plot consists of the first 10 trajectories along with the projections of the functional mean. Besides the fact that the functional mean is almost constant, there is nothing notable. This should start to warn us that treating the whole event as one trajectory is not the best option. If we aspire to classify events like a shot, a pass or a dribble, those are much more "local" events, meaning, in a sixteen seconds span several of those events can occur. This will be dealt with in the preprocessing section. For now, we note that the height of the ball seems to contain more "information" due to its highly oscillatory nature.

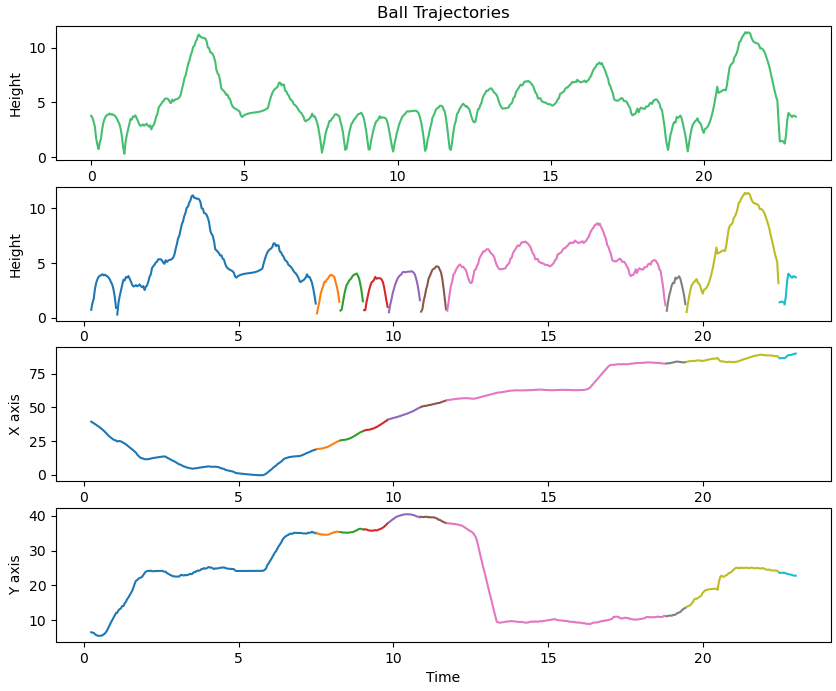
This problem will reappear during the model selection. Whence, we continue with the analysis only for the height projections.

Preprocessing

This part is crucial in the understanding of what is to come. Our aim is to classify events like shots, passes and dribbles just using the trajectories of the ball. The main problem we detected before was that, as of right now, our trajectories are too long in the sense that they contain many different events. So much that even the means dilute and do not capture the geometry of our trajectories. We want to obtain from each trajectory the "relevant" information. This may seem quite difficult but our approach is rather easy. We define "relevant" information as anything that occurs between bounces, i.e, every time $f(t) = [a, b, 0]$, $a, b \geq 0$. Theoretically this would be immediate, let $\{T_1, \cdots T_N\}$ be the set of $t \in \mathbb{R}$ such that $f(t) = [\cdot, \cdot, 0]$. Now we consider

$$f_i = f|_{[T_i, T_{i+1}]} \quad 1 \leq i \leq N.$$

Hence, from one trajectory we obtain N more. However, in reality, measurements are not perfect and there are no recordings of the ball at exactly height 0. We proceeded finding the local minima and keeping those under height 1.



Now this is the data we will classify. Note the problem of having different lengths reappear. We believe this is in the heart of our problem and the solution must take that into consideration.

Regarding outliers, for the sake of the project we are going to find some in the case where we truncated the lengths and considered the whole play, we will note that they are not in reality "out

of place", or at least when you know the underlying phenomena. For instance, we can use the Band Depth as a reference, see [3]. We define the band determined by n trajectories as the set

$$B(f_1, f_2, \cdots, f_n) = \{(t, y) \ : \ t \in (0, 16), \ \min_{i \leq n} f_i(t) \leq y \leq \max_{i \leq n} f_i(t)\},$$

For any f_i and fixed $2 \leq j \leq n$, the quantity

$$BD_n^{(j)}(f) = \binom{n}{j}^{-1} \sum 1\{G(f) \subseteq B(f_{i_1}, \cdots, f_{i_j})\},$$

where the sum is over every possible combination of j functions out of the n and the function $1\cdots$ is one if the graph of f is contained in the band determined by the f_{i_j}. Now, the depth of f is defined by fixing a $2 \leq J \leq n$ and summing over $j \leq J$, i.e,

$$BD_{n,J}(f) = \sum_{j=2}^{J} BD_n^{(j)}(f).$$

Besides the need of picking a J, that indicator function pose a clear problem, we only consider trajectories that stay inside the band completely. A much more logic approach and the one that we use, is to consider the "amount of time" that the trajectory stays inside the band. For any function f_i $1 \leq i \leq n$ and for $2 \leq j \leq n$, fix $f_{i_1}, \cdots f_{i_j}$ and let

$$A_j(f) = \{t \in (0, 16) \ : \ \min_j f_{i_j}(t) \leq f_i \leq \max_j f_{i_j}(t)\}$$

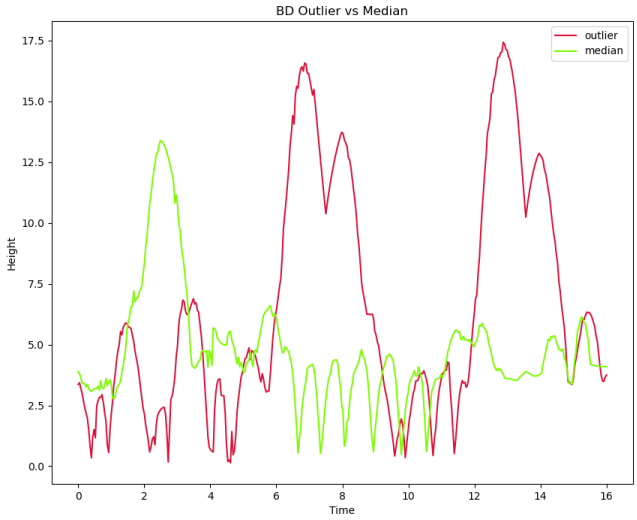
be the time that the function f is in the band defined by the fixed f_{i_j}. Note the abuse of notation f_{i_j} denotes the fixed family and also the last element of said family. Now,

$$MBD_n^{(j)}(f) = \binom{n}{j}^{-1} \sum m(A_j(f))/m(0, 16),$$

where again the sum is taken over every possible combination of j functions, and m denotes the lebesgue measure. The depth is defined as before, fix a $2 \leq J \leq n$ and

$$MBD_{n,J}(f) = \sum_{j=2}^{J} MBD_n^{(j)}(f).$$

Thus we have



where we took the median as the deepest element. Note that based on the definition of band depth, it is believable that said function is the least deep, but this is only because it presents two late "peaks". However, if we consider the information broken into pieces as we said we will, each piece is not necessarily an outlier as there are many others with similar geometry and magnitude. This supports the idea that the correct approach is to find outliers inside our classes. In fact, this outlier just represents a play where two threes were shot in the sixteen seconds span.

Recall we have some trajectories repeated, and in this case the problem is notable. The three deepest trajectories are in fact the same one, and the same occurs for the two least deep. However, this still does not concerns us since we will break down every trajectory into pieces and our objective is to classify. But, this goes to show that depending on the problem you are tackling, data cleaning can be critical. Furthermore, this gives us an easy criteria to get rid of those trajectories. Just order the trajectories based on their depth, and if two consecutive ones have the same exact depth, get rid of one.

Model

We take two parallel approaches. First, we build a rule-based classifier using our "expert" knowledge of the phenomena. Then, we will manually classify 49 plays and build a centroids classifier based on the Dynamic Time Warping (DTW) distance. Our main objective is to show that when there is a clear understanding of what we are trying to classify, even a really basic form of the classical approach (rule based) has a great performance.

In order to classify, we need groups. Due to the fact that we will continue to work on this idea in the future, and to show possible extensions, there are 9 labels, of which we only use 6. We can classify a trajectory in the following classes

- Pass
- Alley oop
- Post move
- Three
- Mid range
- Layup
- Free throw
- Dribble
- Ball movement

where the three gray coloured are the ones we avoid for now. Recall the usual notation for the projections $\pi_i : \mathbb{R}^3 \to \mathbb{R}$, $i \leq 3$, where π_i projects to the ith coordinate.

The rule based classifier is the following function $g : C((0,\infty), \mathbb{R}^3) \to \{0, 1, \cdots, 8\}$, denoting by M the preimage of $\max_t \pi_3 f(t)$

$$g(f) = \begin{cases} 2 & \text{if} \quad f(M) \in [0.94] \times [0.50] \times [0, 10) \;\&\; 1.4 \leq m(sop(f)) \leq 2.4 \\ 3 & \text{if} \quad f(M) \in [0.94] \times [0.50] \times [15, \infty) \\ 4 & \text{if} \quad f(M) \in [0, 9]^c \times [22, 28]^c \times [10, 15] \\ 5 & \text{if} \quad f(M) \in [0, 9] \times [22, 28] \times [10, 15) \cup [85, 94] \times [22, 28] \times [10, 15) \\ 8 & \text{if} \quad f(M) \in [0.94] \times [0.50] \times [0, 10) \;\&\; m(sop(f)) \geq 2.4 \\ 7 & \text{otherwise} \end{cases}$$

where m is again the lebesgue measure and *sop* denotes the support of f.

Writing it in the form of a function is more a fun exercise than anything necessary. Lets describe it intuitively. The cutoff for a three point shot is that it achieves height 15 or more, reason is that the average three pointer achieves 15.5 feet. Then, a layup has height between 10 and 15, and we must be under the basket, i.e, in the first 5 feet of the court in the X and in the middle for the Y. Note the symmetry when it comes to the X axis, there is a basket in the opposite side. A mid range shot is anything considered a shot, i.e, height higher than 10 feet (the official height of

the rim), that is not a three or a layup. Regarding non shooting plays, we consider a single pass if the support of f is greater than 1.4 seconds, translating that to our situation is that the ball do not bounce in 1.4 seconds, recall our trajectories are always between bounces. Moreover, if it is a non shooting play and the support is greater than 2.4 seconds we say is ball movement, i.e, several consecutive passes without bouncing. Finally, anything we could not classify is a dribble. Note everything that is not classified are trajectories of support less than 1.4 seconds and height less than 10 feet.

Lets see how it works. Note we are using the information of the other axis, but we will only show the height since it is the representative.

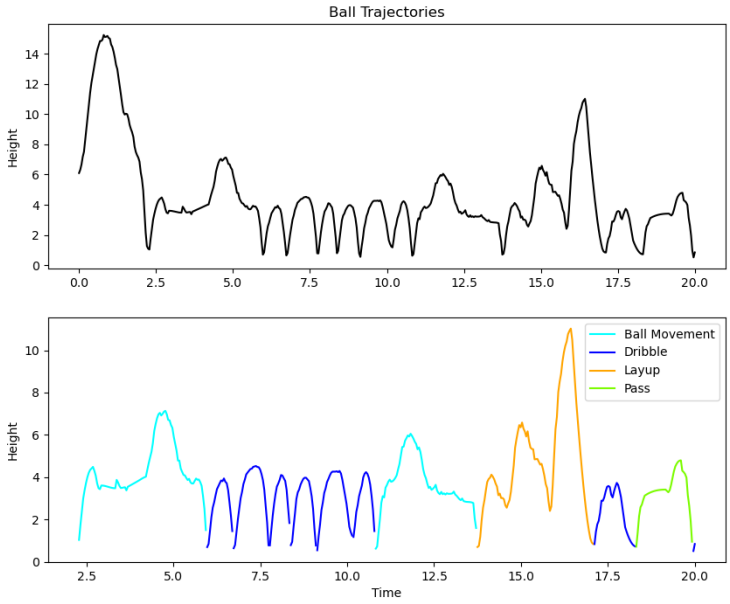


Figure 2

It should be noted that because of how we break down trajectories, we start at the first zero value of $\pi_3 f$. Thus we do not keep that first peak in 2. However, note that time axis record it perfectly.

In the rule based classifier we have been able to capitalize that our functions f have image in $\mathbb{R}^3$. Moreover, the fact that every trajectory has different length was not a problem, in fact it was also used. However, if we want to apply any of the theory we have seen in class, these two facts are major drawbacks. In order to overcome them, we must introduce a new framework.

We start by introducing the notion of warping based distance, see [1]. We denote

$$T_i : ((f_i(t_1), t_1), \cdots, (f_i(t_{n^{(i)}}), t_{n^{(i)}}))$$

the time series induced by our function f_i, where t_n is taken to be the last point of f_i domain. Note that by definition, our functions domains are closed bounded intervals, our problem is that for two different f_i f_j, $t_{n^{(i)}} \neq t_{n^{(j)}}$.

In order to define a warping distance, we want to compare two T_i, T_j. Thus, we arrange them to form a $n^{(i)} \times n^{(j)}$ grid G. Each cell $g_{k,l}$ corresponds to the pair $(f_i(t_k), f_j(t_l))$.

Definition 1. A warping path, $W = w_1, \cdots w_{|W|}$ crosses the grid G such that,

- $w_1 = g_{1,1}$
- $w_{|W|} = g_{n^{(i)},n^{(j)}}$
- If $w_k = g_{a,b}$ then $w_{k+1} = g_{a+1,b}, g_{a,b+1}$ or $g_{a+1,b+1}$.

The distance is then defined as

Definition 2.

$$D(T^i, T^j) = \min_W \left(\sum_{k=1}^{|W|} \delta(w_k) \right)$$

where $\delta(w_k) = \delta(g_{a,b}) = \delta(f_i(t_a), f_j(t_b))$ is the cost function and the minimum is taken over all warping paths.

The well known dynamical time warping is just when taking the cost function the euclidean norm. It is notable that this idea comes from the Levenshtein distance, which is a string metric that measures the amount of edits a word must suffer to become the other one. There are many papers discussing different cost functions, [2], [7]. This is also a topic on which we will keep working in the future, specially the fact that some type of theoretical generalization must be possible.

In order to understand the next step, it is crucial to build some intuition on DTW. Take for instance two sines

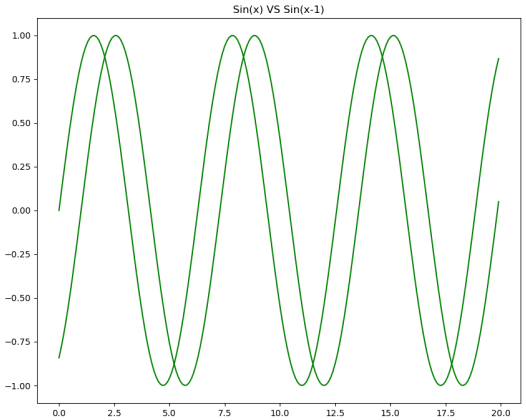


Figure 3

Although one of them is shifted, they still are the same function and we would like for them to have distance 0, or at least close to the phasing there is between them. First lets see their alignment,

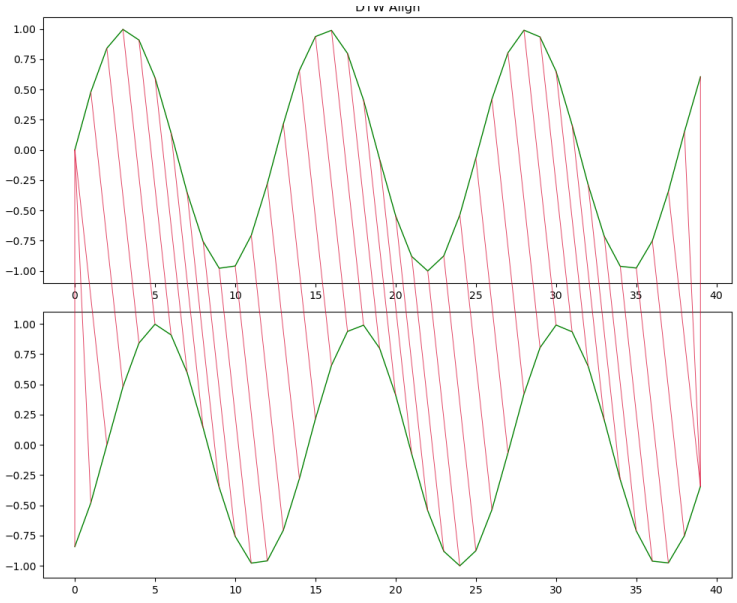


Figure 4

This shows the alignment that corresponds to the path that minimizes distance. Note the first three points of the shifted sin correspond to only the first of the standard sin, that is how you "align". Then, the correspondence is one to one until the end, meaning in the grid G the path would move diagonally, this can be seen in Figure 5.

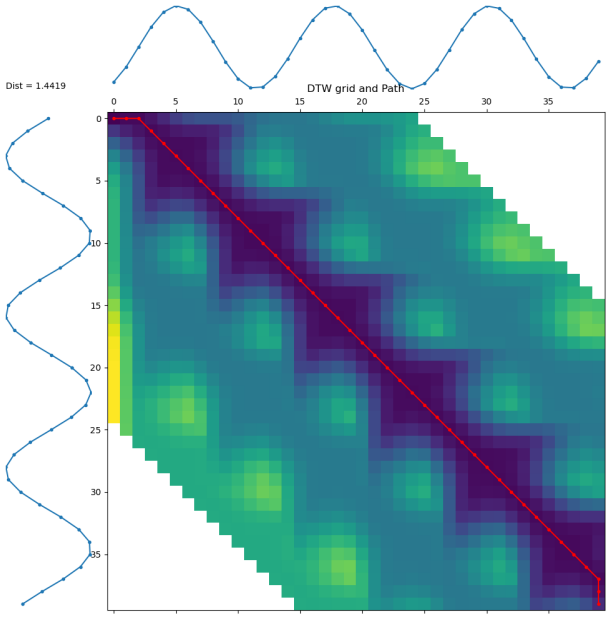


Figure 5

In the context of classification, many algorithms require a method to represent in one object, informations from a set of objects. Algorithms like k-medioids are using the medioid of a set of objects. Classical k-means need to average a set of objects finding a mean. This is really

straightforward when using an euclidean distance. However, this is much more difficult for objects under DTW. The classical approach of finding an element that minimizes the squared distances still holds. Say a g such that

$$\sum_{i=1}^{n} DTW^2(g, f_i) \leq \sum_{i=1}^{n} DTW^2(h, f_i) \forall h \in X$$

where X denotes the function space we are working in.

The first two main averaging methods for DTW were the Non linear alignment and averaging filters (NLAAF), and the Prioritized shape averaging (PSA), see [4], [5]. However, we are going to use the averaging method presented in [6], it has established as the standard when working with DTW. Mainly, because it gives a global approach and is completely intuitive.

The main idea is to select a trajectory inside the group we want to average, we can technically select a totally random one not from the group but there is no point, then compute every DTW path. Now, the first point of the selected trajectory will correspond to many others points from the other trajectory (note that at least one), thus first point of the average is just the arithmetic mean of those points. Once you finished, you can average now with the new trajectory as the fixed. The idea is really easy but the literature lacks formalization. It should be noted that the new average sequence will have the length of the fixed one. Before moving on to the classifier we would like to note the importance of the reference function.

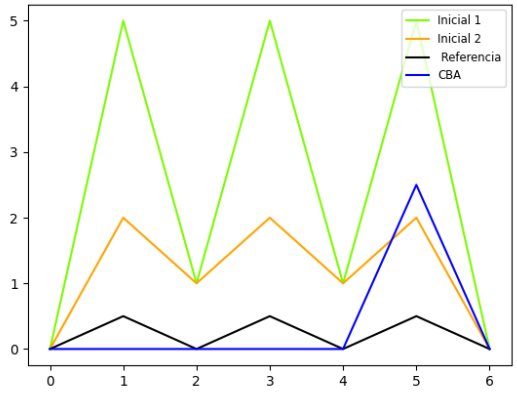


Figure 6

What is underlying here is the low height of the reference function, it leads to unusual aligning where of course as many points as possible will be assigned to the 0 value.

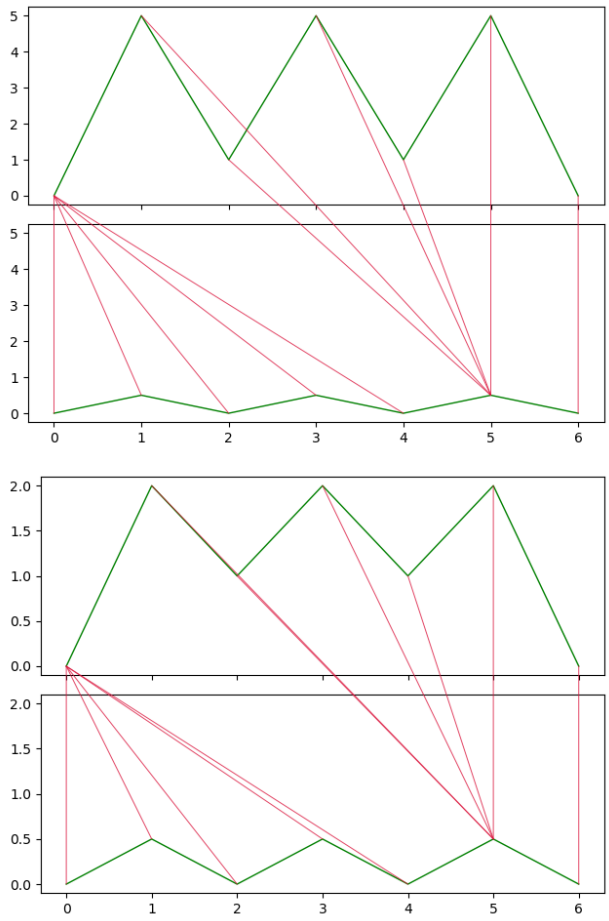


Figure 7

However, a simple modification leads to a much more apparent result.

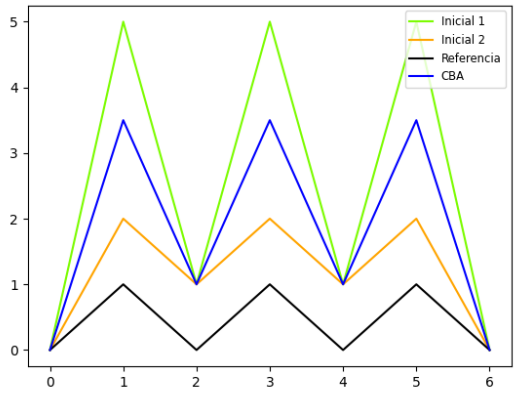


Figure 8

Note the alignment now is perfect, meaning completely 1 to 1, and we thus just compute the

arithmetic mean.

We manually classified a sample made of 7 passes, 11 threes, 5 mid range shots, 5 layups, 14 dribbles and 7 ball movements. Using the DBA algorithm on said sample we obtain a look on how the average trajectory of a class looks. It should be noted that this approach do not take into account the x and y coordinates. Theoretically it could be done because we can use DTW with the euclidean norm for $\mathbb{R}^3$, but there are some problems. Mainly because the units are completely different in every axis.

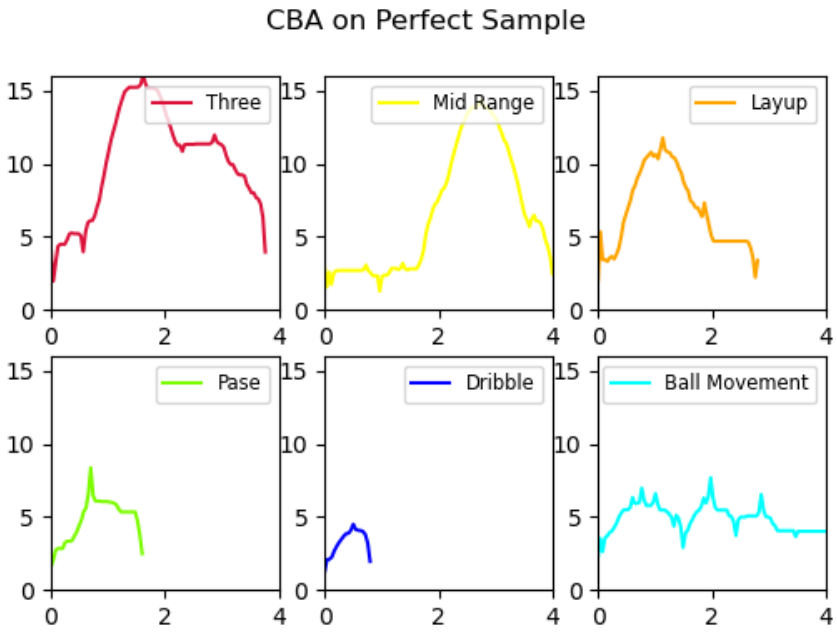


Figure 9

Note how they capture the form and magnitude of the class. If we did the standard pointwise approach to find an average we would fail completely. It should be noted that we did the "pointwise" mean using all the points available, recall our trajectories have different lengths.

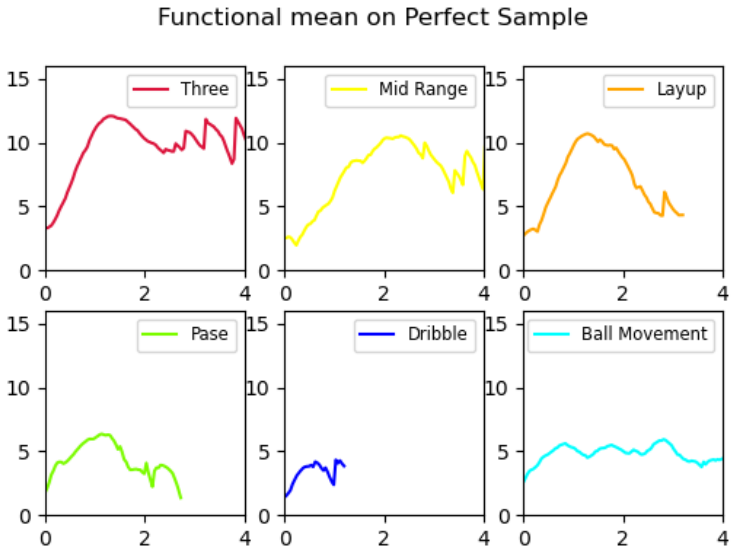


Figure 10

Having those DTW "means", we can define a standard medioids classifier that takes a new trajectory, one dimensional in this case, and assign to the class which has the minimal DTW distance between the trajectory and the class representative. Before moving on to the validation, we can see how it works on the trajectory of Figure 2.

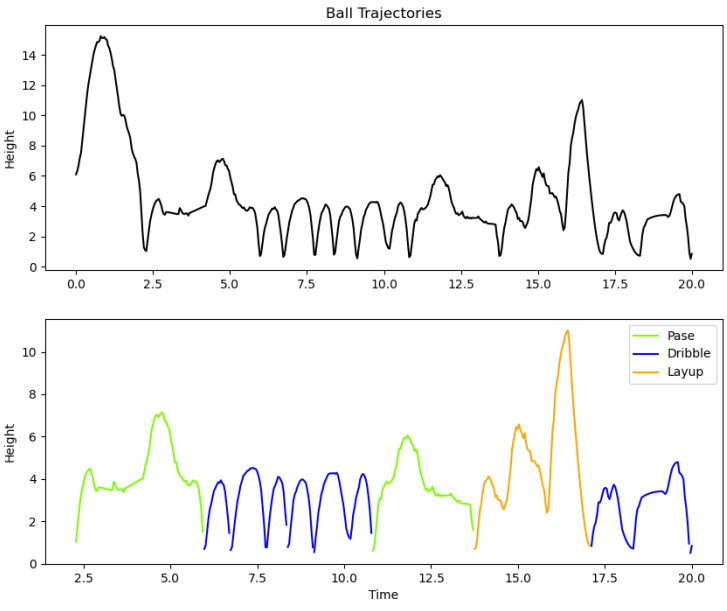


Figure 11

Validation

Te approach for the rule based is straightforward, checking how well it classifies our perfect sample. For the DTW classifier we will also use it on the sample, but we will also do cross validation.

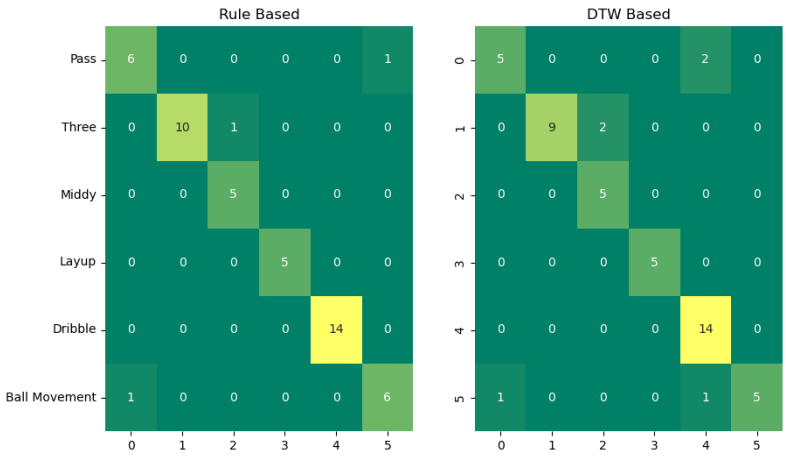


Figure 12

The accuracy was of 93.8% and 87.7% respectively. Via cross validation (leave one out), accuracy does not vary at all

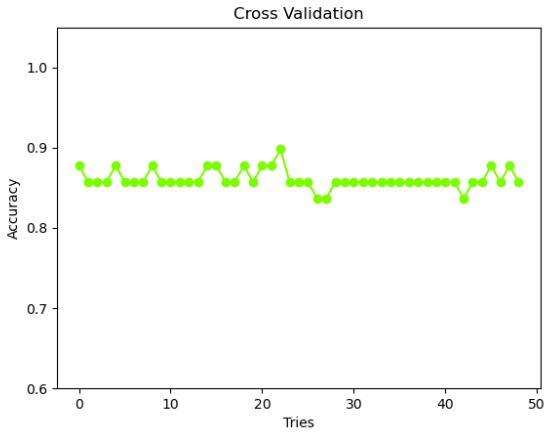


Figure 13

References

[1] Philippe Besse, Brendan Guillouet, Jean-Michel Loubes, and Royer François. Review and perspective for distance based trajectory clustering. *arXiv preprint arXiv:1508.04904*, 2015.

[2] Lei Chen and Raymond Ng. On the marriage of lp-norms and edit distance. In *Proceedings*

of the Thirtieth International Conference on Very Large Data Bases - Volume 30, VLDB '04, page 792–803. VLDB Endowment, 2004.

[3] Sara López-Pintado and Juan Romo and. On the concept of depth for functional data. *Journal of the American Statistical Association*, 104(486):718–734, 2009.

[4] Vit Niennattrakul and Chotirat Ann Ratanamahatana. On clustering multimedia time series data using k-means and dynamic time warping. In *2007 International Conference on Multimedia and Ubiquitous Engineering (MUE'07)*, pages 733–738, 2007.

[5] Vit Niennattrakul and Chotirat Ann Ratanamahatana. Shape averaging under time warping. In *2009 6th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*, volume 02, pages 626–629, 2009.

[6] François Petitjean, Alain Ketterlin, and Pierre Gançarski. A global averaging method for dynamic time warping, with applications to clustering. *Pattern Recognition*, 44(3):678–693, 2011.

[7] M. Vlachos, G. Kollios, and D. Gunopulos. Discovering similar multidimensional trajectories. In *Proceedings 18th International Conference on Data Engineering*, pages 673–684, 2002.