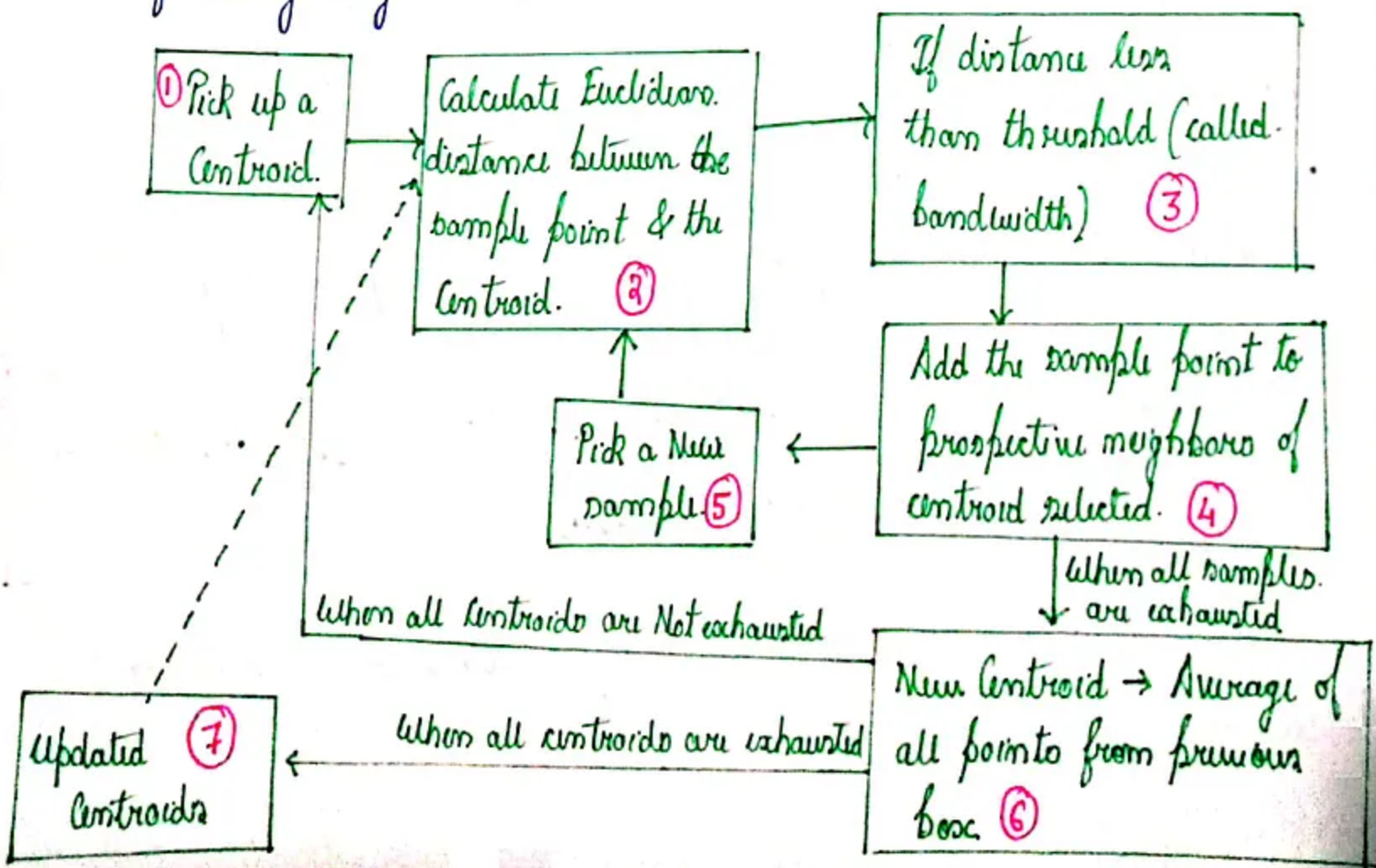
Mean Shift Clustering

Mean Shift clustering is a centroid based clustering algorithm which iteratively update centroids in the data till blobs are discovered in the sample provided.

Following steps are involved in Mean shift clustering procedure -

Step 1:- Centroid Initialization → All data points are initialized to cluster centroids. So we start with as many clusters as data points & then aim to achieve the optimal number of clusters.

Step 2:- Duration based cluster update - We use two parameters here - bandwidth & Number of iterations. We then use both the parameters in following way -



We will continue the above procedure iteratively for a Number of steps till we get final updated centroids. The samples having same centroids belong to the same cluster.

We will now look at the entire process using a self built code before understanding how sklearn parameters work.

```
X = np.array([ [1,1], [1, 1.5], [1.9, 2.2],
               [7.5, 6.5], [6.6, 5.6], [9,10], [9,11], [10, 9.86]])
y = [1, 1, 1, 2, 2, 3, 3, 3]
```

We can clearly see that the data has 3 clusters. We can further plot them to see how clusters are distributed (using true class labels y).

We now proceed with setting the parameters for mean shift clustering.

```
bandwidth = 3
n_iter    = 10
```

We will now perform Step 1 of the algorithm discussed previously.

```
centroids = dict(zip(range(X.shape[0]), np.array([X[i,:] for i in
                 range(X.shape[0])])))
```

We will now look to perform necessary iterations to determine final clusters.

```
for n in range (n_iter):
    new_centroids = centroids.copy()
    updated_samples = []
    for i in range (len(centroids)):
        centroid = centroids[i]  ---> Picking up one centroid.
        neighbor = []
        index    = []
        for j in range (X.shape[0]):
            sample = X[j, :]  ---> Box (5) of Step 2.
            if np.linalg.norm (sample - centroid) < bandwidth:
                neighbor.append(sample)
                index.append (j)
        new_centroid = np.average (neighbor, axis=0)  ---> Box (6) of Step 2
        for K. in range (len (neighbor)):
            new_centroids [index [k]] = new_centroid.  ---> Box (6) of Step 2
    centroids = new_centroids  ---> Box (7) of Step 2.
```

→ Performing calculations as per box ②, ③, ④ of Step 2.

The necessary clusters will be computed & stored in 'centroids' variable.

Now we will look forward to see how Mean Shift clustering is implemented in sklearn's clustering library & analyse its parameters & attributes.

Parameters :-

1. bandwidth - The bandwidth parameter does exactly what our own bandwidth parameter did. but we use a rbf kernel over here instead of computing Euclidean distance.

Attributes :-

1. cluster_centers_ :- Coordinates of cluster centers.
In our case it was- [(1.3, 1.57), (7.05, 6.05), (9.33, 10.29)]

2. labels_ :- Cluster labels assigned for each point.

Let us create a fairly bigger data (as compared to data used in our example) & then apply sklearn's implementation of mean shift clustering.

```
centers = [ [1,1], [-1,-1], [-1,1] ]
X, labels_true = make_blobs (n_samples = 100, centers = centers,
                             cluster_std = 0.5, random_state = 0)
meanshift = MeanShift (bandwidth = 1). fit (X)
our_labels = meanshift. labels_
```

→ Predicted Labels / Clusters

→ Instance of MeanShift with bandwidth parameter equating to 1