**Problem statement: Bike drop distance optimisation**

Using Python, please optimize for the following: In the df1, you have the bikes and its pickup locations. In df2, you have the drop locations. You have to assign a drop location from df2 to every bike present in df1 in such a way that it meets all the below conditions.

Here distance is haversine distance between two coordinates(pickup and drop).

Df1: Bike pickup locations (excel attached)

Df2: Bike drop locations (excel attached)

Constraints:

| **Hex with avg daily sessions/supply** | **Minimum no. of bikes should be drop** | **Maximum no. of bikes can be drop** |
| --- | --- | --- |
| >120 | 6 | 8 |
| [100 to 120) | 2 | 4 |
| [50 to 100) | 2 | 3 |
| [30 to 50) | 1 | 2 |
| [20 to 30) | 1 | 1 |
| [10 to 20) | 0 | 1 |
| <10 | 0 | 0 |

1. You have to always aim for the maximum number of bikes to be dropped in the respective destination hex but if the drop haversine distance is more than 4km you will have to assign a different drop hex to it.
2. Alway aim for the nearest drop point from df2 for each pickup bike in df1. Try to keep the haversine drop distance minimum as much as possible but it should be meeting contarints 1 & 2 first.
3. Priority order is in the order, like 1,2,3 are high priority and so on.
4. Overall( sum of the bike haversine drop distance/total drop locations), average haversine drop distance should be minimum but max it can go upto ~2km.
5. You have to assign drop location to each bike present in the bike pickup dataframe.
6. Please share the notebook(.ipynb) verison with HR.

To know more about hex please visit the website: <https://eng.uber.com/h3/>

Drop location is a hex\_9 which is an Uber’s Hexagonal Hierarchical Spatial Index. I have provided you the centroid of the drop hex\_9 so that you can calculate the haversine distance between the pickup coordinates and drop hex.