**Anomaly Detection in Public Street Lighting using Clustering Algorithms**

**Requirements**

* Data set is converted from JSON to Spark DataFrame by using **PySpark=3.2.1**.
* To implement the clustering algorithm, Spark DataFrame is converted into the Pandas DataFrame and scikit-learn is exploited.
* Code is implemented by using PyCharm IDE.

**Running code classes in Anomaly-Detection project**

Project repository contains data set folder with name “PELL\_Data\_with\_Synthetic\_Anomalies” and implementation files for K-means, DBSCAN, OPTICS, and synthetic anomaly injection.

**K-means Model**

Run main.py and uncomment

kmeans = KmeansAnomalyDetection()

kmeans.Kmeans\_anomaly(no of clusters)

**DBSCAN Model**

Run main.py and uncomment

dbscan = DBSCANAnomalyDetection()

dbscan.dbscan\_anomaly(eps\_value, min\_sample\_value)

**OPTICS Model**

Run main.py and uncomment

optics = OPTICSAonamyDetection()

optics.optics\_anomaly(eps\_value)

**Artificial Anomalies**

In order to evaluate the performance of cluster algorithms for anomaly detection from street lighting data, we introduced six anomalous scenarios w.r.t. street lighting domain. By following these domain specific scenarios, synthetic anomalies are automatically inserted into the original data set. This helps to determine the ability of clustering algorithms to detect anomalies by using some standard evaluation metrices.

The file name “[synthetic\_anomaly\_injection.py](https://github.com/MubashirAliCheema/Anomaly-Detection/blob/main/synthetic_anomaly_injection.py)” contains the implementation for all the six scenarios to automatically insert synthetic anomalies.

Run main.py and uncomment whatever scenario you want to invoke.