Predictive Maintenance of Machine Data Analysis

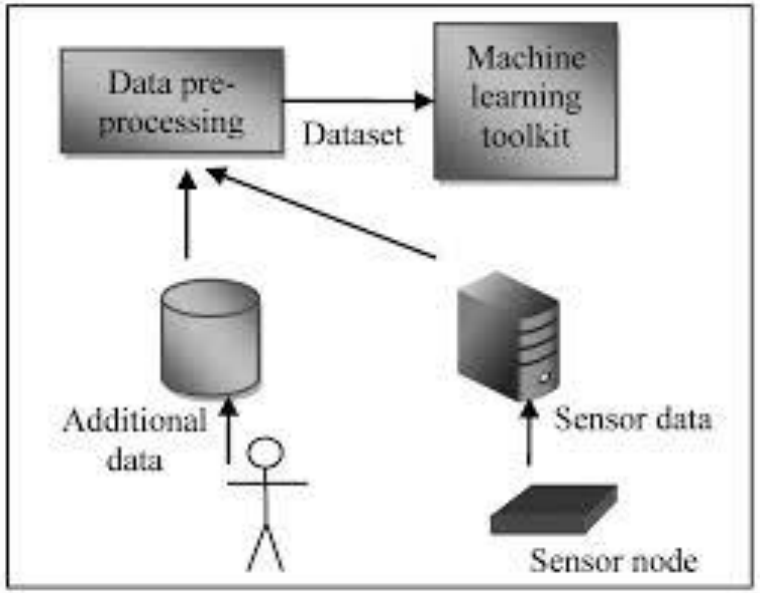
Under the supervision of – Professor Akhilesh Kumar (ISE Dept)

IIT Kharagpur

Project done and presented by- Shubham Sonal
- 16NA10028

About the project:

Machine condition and operation mode analysis is predictive maintenance tool for safety measure of machine system.



Given Dataset:

1. Size of dataset- (168697, 6)
2. 6 Columns in of dataset- (Timestamp, Sensor 1, Sensor 2, Sensor 3, Sensor 4, Label)
3. Total class label- 7 (seven different mode of operation)

A sample of provided dataset:

	Timestamp	Sensor 1	Sensor 2	Sensor 3	Sensor 4	Label
0	21-02-2020 12:35	2.79	2.78	1159.0	9.0	High Speed Normal
1	21-02-2020 12:35	2.84	2.78	1159.0	9.0	High Speed Normal
2	21-02-2020 12:35	2.82	2.80	1159.0	9.0	High Speed Normal
3	21-02-2020 12:35	2.81	2.80	1159.0	9.0	High Speed Normal
4	21-02-2020 12:35	2.82	2.76	1159.0	9.0	High Speed Normal

Number of datasets in each class (modes of operation):

High Speed Normal	102696
Medium Speed Normal	65833
Low Speed High Vibration	126
Low Speed Normal	31
Medium Speed High Vibration	5
High Speed High Current	4
High Speed High Vibration	2

Correlation matrix:

	Sensor 1	Sensor 2	Sensor 3	Sensor 4
Sensor 1	1.000000	0.992006	0.424558	0.918957
Sensor 2	0.992006	1.000000	0.460436	0.921139
Sensor 3	0.424558	0.460436	1.000000	0.480065
Sensor 4	0.918957	0.921139	0.480065	1.000000

Data Pre-processing

- The data provided was very precise to an extent there were no such outliers found in the data.
- In order to visualize the data we need to reduce its dimension to 2d for better visualization.
- Also, variation in data is too much so data must be standardized.

PCA for dimensionality reduction

- PCA calculates the new projection of given dataset and the new axis are based on the standard deviation of the variable.

- So, any variable with a high standard deviation will have a higher weight for the calculation of axis than a variable with a low standard deviation.

In python we can import a library decomposition and specify number of components.

Result of PCA: explained variance ratio of 79.35% and 17.94% in first and second principle components respectively

```
from sklearn.preprocessing import StandardScaler
std_data=StandardScaler().fit_transform(kmeandata)
kmeandata=pd.DataFrame(std_data, columns=columns)
from sklearn.decomposition import PCA
from sklearn import decomposition
pcaa = PCA(n_components=2)
X = pcaa.fit_transform(kmeandata)
print(pcaa.explained_variance_ratio_)
print(X)
print(X.shape)
```

```
[0.79349494 0.17942786]
[[ 0.90274319  0.61192656]
 [ 0.94695975  0.63196488]
 [ 0.94764448  0.63063565]
 ...
 [-3.10429783  3.6781379 ]
 [-3.10464019  3.67880252]
 [-3.10429783  3.6781379 ]]
(168697, 2)
```

K-means clustering

- K-means is a centroid-based algorithm, or a distance-based algorithm, where we calculate the distances to assign a point to a cluster. In K-Means, each cluster is associated with a centroid.

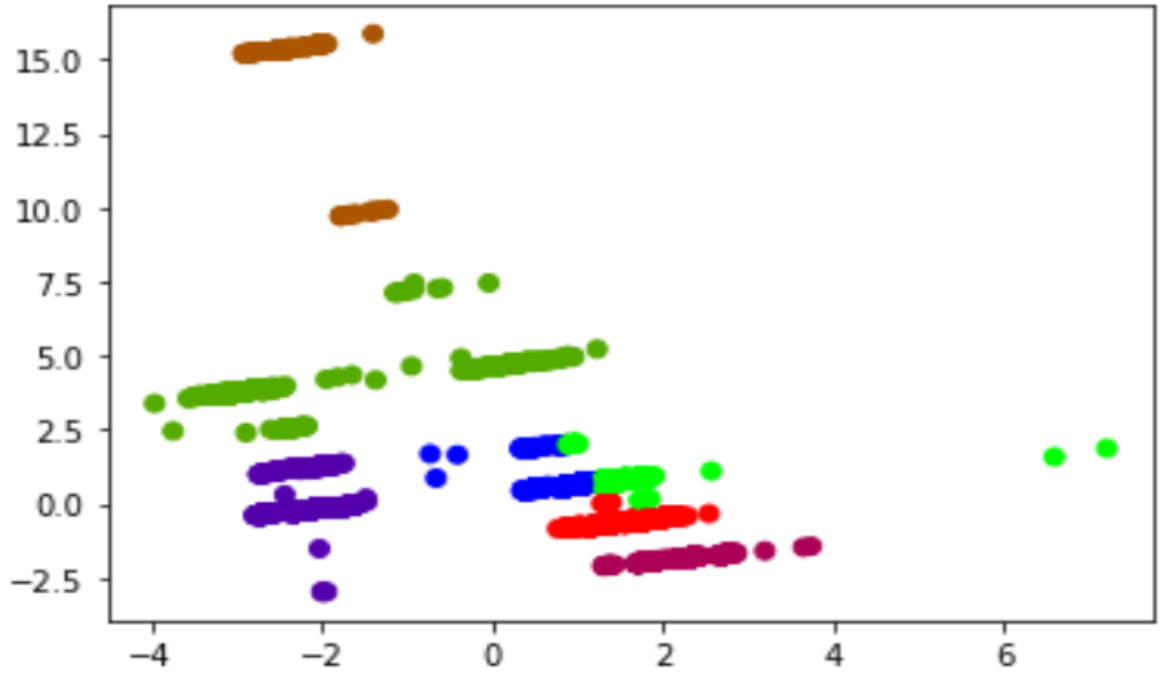
- Since our data is now ready for K-means clustering we apply k-means algorithm with k=7 (number of cluster)

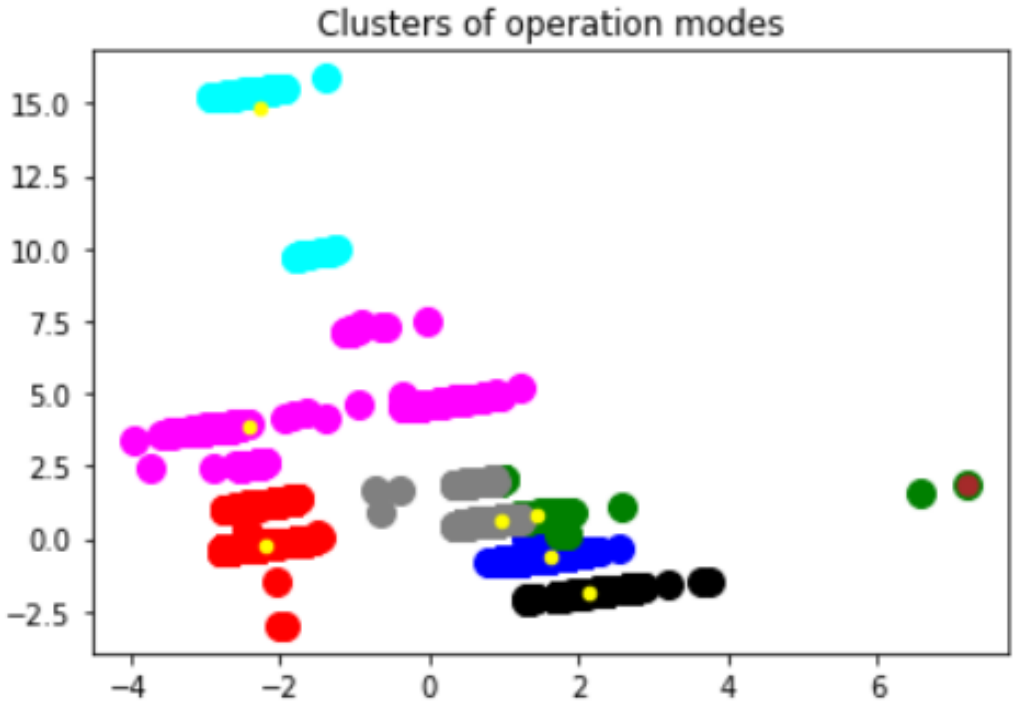
In [23]:
```
from sklearn.cluster import KMeans
k=7 #number of cluster
kmeans=KMeans(n_clusters=k)
kmeans.fit(X)
```

Out[23]:
```
KMeans(n_clusters=7)
```

In [24]:
```
plt.scatter(X[:,0],X[:,1],c=kmeans.labels_,cmap='brg')
```

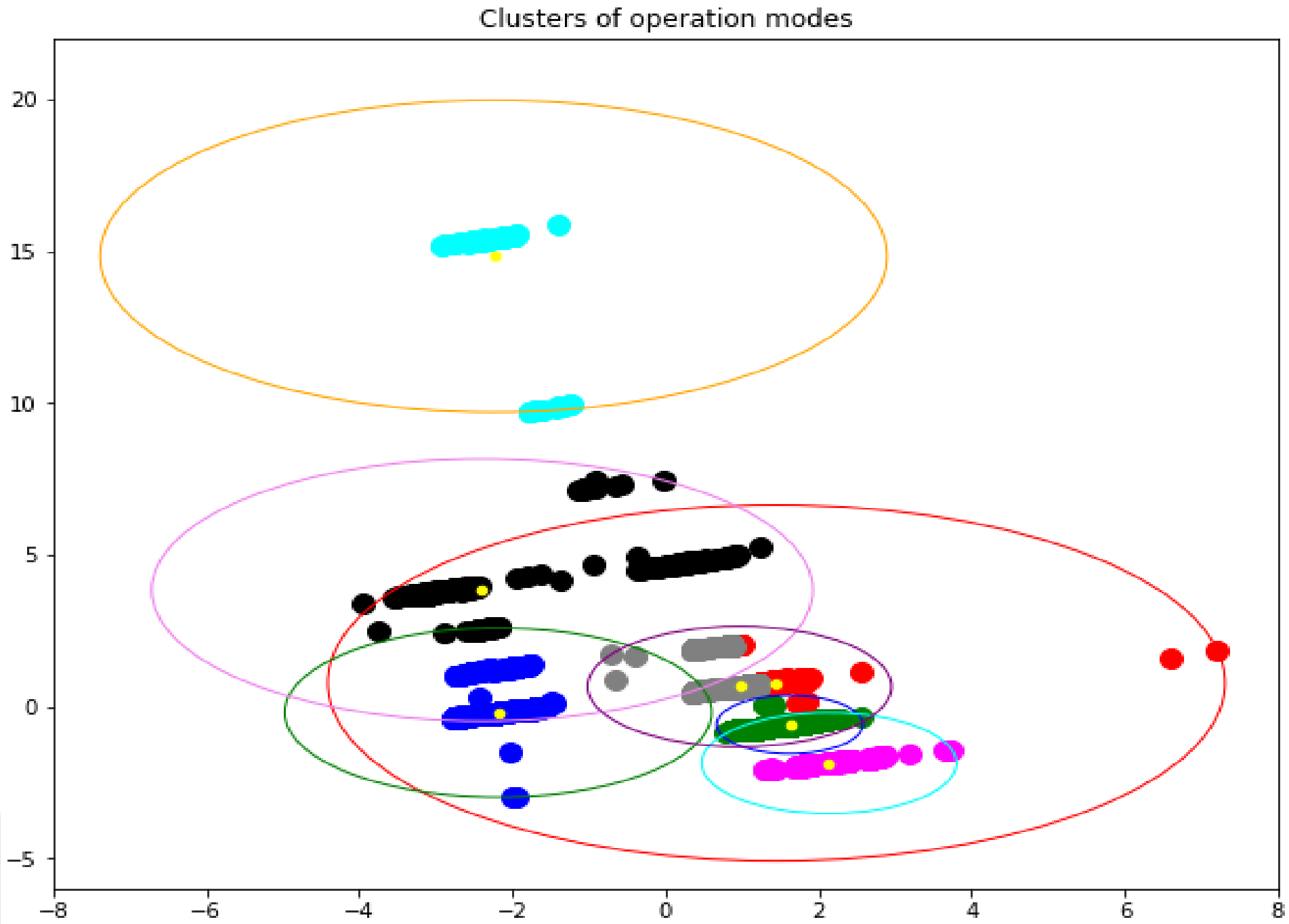
Out[24]:
```
<matplotlib.collections.PathCollection at 0x13251f10eb0>
```





Here yellow points represent the centre of each cluster in the above plot.

we can find the range of each cluster by finding the distance of the data which is at maximum distance from the centroid of that cluster.


Clusters of operation modes
20
15
10
5
0
−5
−8
−6
−4
−2
0
2
4
6
8

Testing Model

We must get the new unseen instances ready first which include all these steps:

- We must first standardize the data using the mean and standard deviation value we have from the past dataset.

- Then using the loading factor, we can get from PCA we applied while training we can transform the new data into 2d dataset.

PCA loadings-

PC1- [0.54351435, 0.54774608, 0.34075225, 0.53707941]

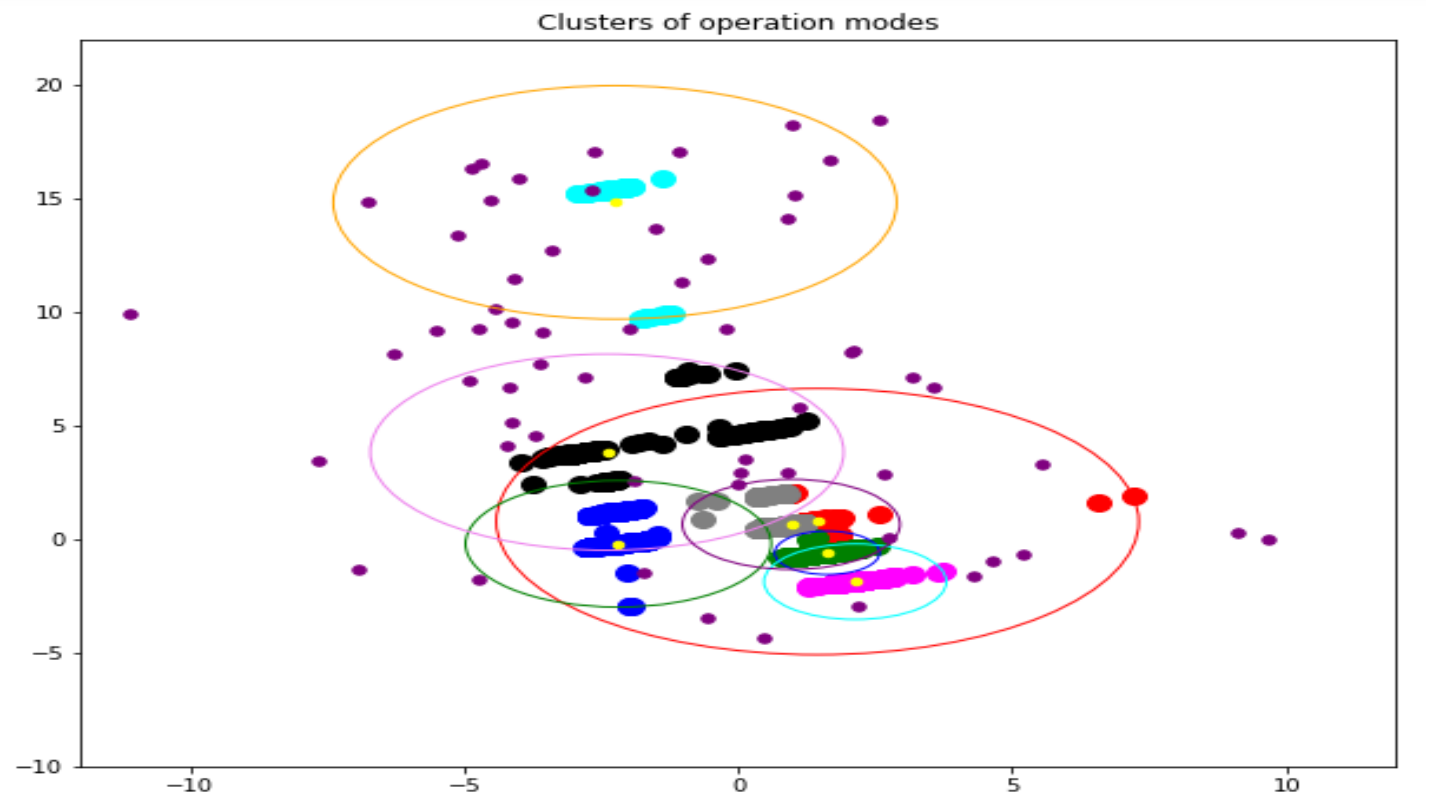
PC2- [0.24631304, 0.19934737, -0.93772127, 0.14237042]

So, each sensor data is multiplied by corresponding factor and added.

Then we get PC1 and PC2 for new dataset.

- Now this new dataset is checked weather it belongs to any of the existing or not. If not than the dataset is flagged and stored in another table.

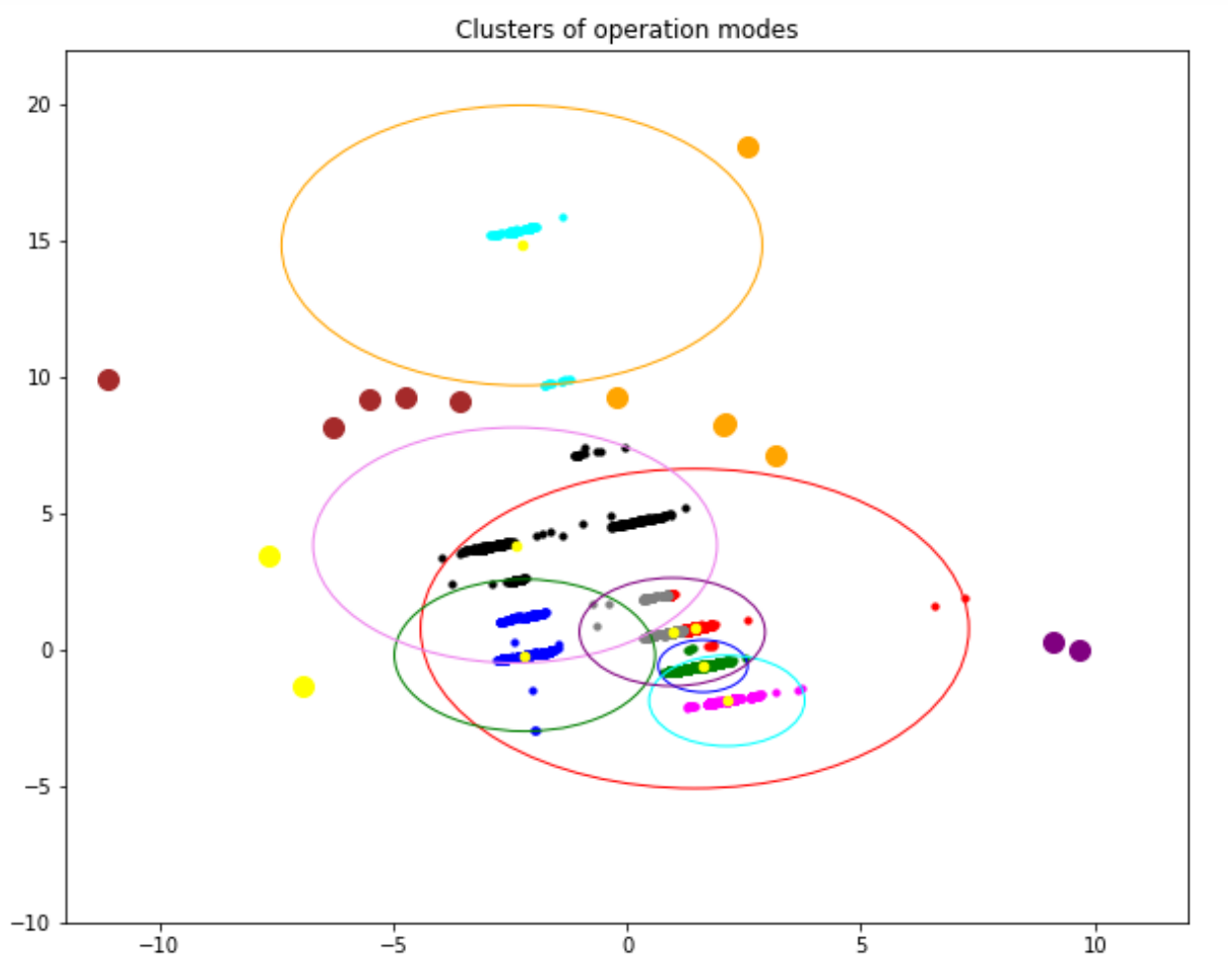
- Now if we have many flagged datasets, we again apply k-means clustering algorithm in it.



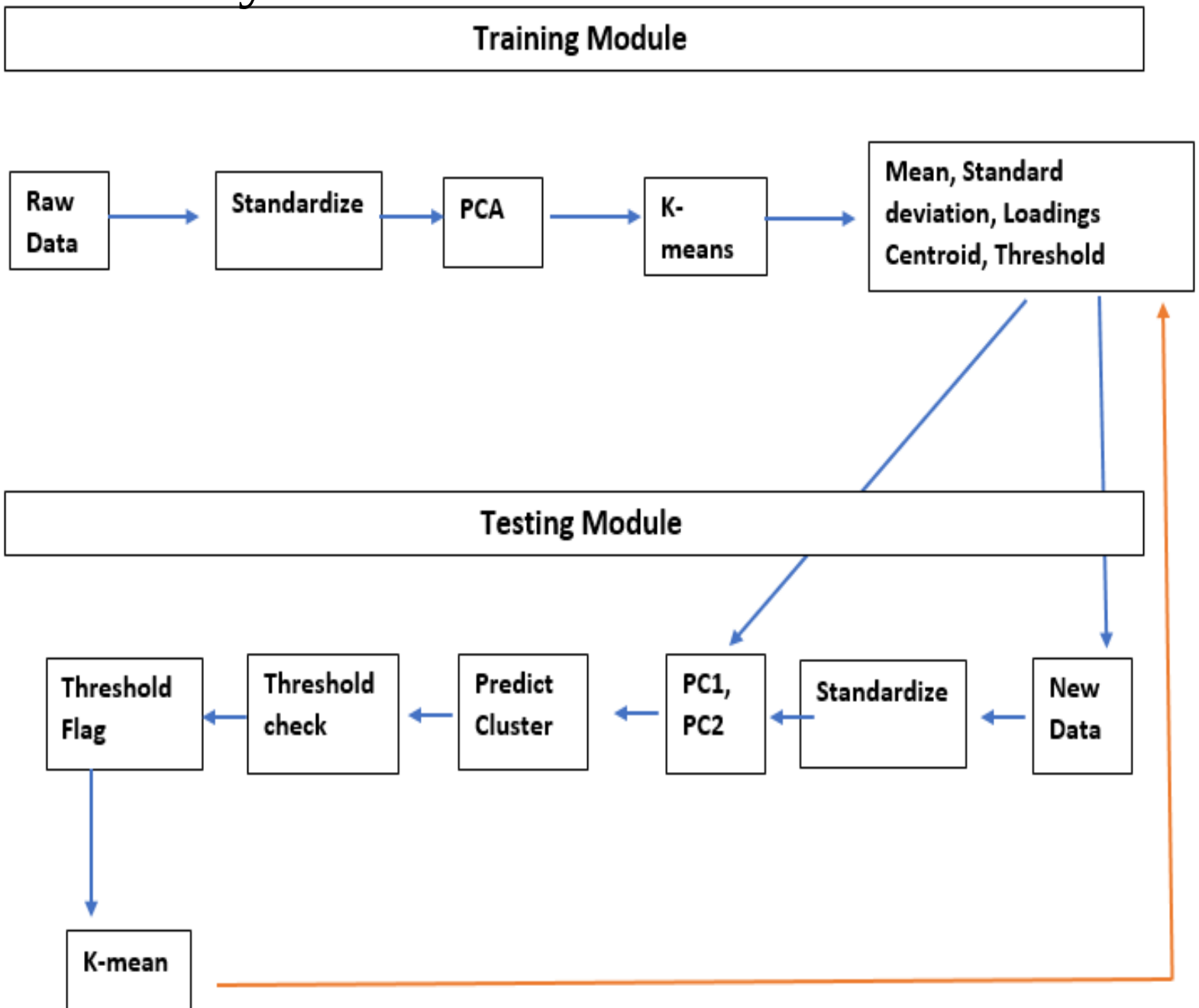
New incoming data set:

	sensor1	sensor2	sensor3	sensor4	flagged	cluster
0	3	1	729	9	1	4
1	7	4	980	7	0	None
2	8	2	676	10	1	4
3	5	1	1131	4	1	1
4	4	1	583	2	0	None
5	2	1	938	6	1	3
6	5	4	590	2	1	4
7	6	1	995	4	1	8
8	8	3	619	2	1	4
9	7	3	725	10	1	4
10	1	2	621	0	0	None

Result: After the flagging, the data points which come out of the range are clustered in new groups.



Summary


Training Module
Raw Data
Standardize
PCA
K-means
Mean, Standard deviation, Loadings Centroid, Threshold
Testing Module
Threshold Flag
Threshold check
Predict Cluster
PC1, PC2
Standardize
New Data
K-mean

THANK YOU