```
pip install pyod
```

```
Collecting pyod
  Downloading pyod-0.9.4.tar.gz (111 kB)
     |████████████████████████████████| 111 kB 28.3 MB/s
Requirement already satisfied: joblib in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: matplotlib in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: numpy>=1.13 in /usr/local/lib/python3.7/dist-pa
Requirement already satisfied: numba>=0.35 in /usr/local/lib/python3.7/dist-pa
Requirement already satisfied: scipy>=1.3.1 in /usr/local/lib/python3.7/dist-p
Requirement already satisfied: scikit_learn>=0.20.0 in /usr/local/lib/python3.
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: statsmodels in /usr/local/lib/python3.7/dist-pa
Requirement already satisfied: llvmlite<0.35,>=0.34.0.dev0 in /usr/local/lib/p
Requirement already satisfied: setuptools in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.7/d
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /us
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.7/dist-p
Requirement already satisfied: patsy>=0.4.0 in /usr/local/lib/python3.7/dist-p
Requirement already satisfied: pandas>=0.19 in /usr/local/lib/python3.7/dist-p
Requirement already satisfied: pytz>=2017.2 in /usr/local/lib/python3.7/dist-p
Building wheels for collected packages: pyod
  Building wheel for pyod (setup.py) ... done
  Created wheel for pyod: filename=pyod-0.9.4-py3-none-any.whl size=130726 sha
  Stored in directory: /root/.cache/pip/wheels/50/32/84/d64079e8bf087c1b231c9b
Successfully built pyod
Installing collected packages: pyod
Successfully installed pyod-0.9.4
```

```
import numpy as np
import pandas as pd
```

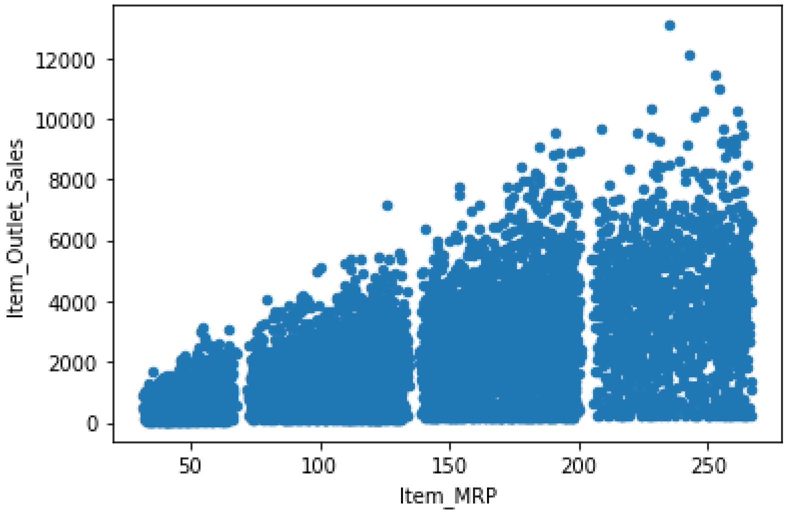
```
data=pd.read_csv('BigMart.csv')
```

```
data.head()
data.shape
```

```
(8523, 12)
```

```
import matplotlib.pyplot as plt
data.plot.scatter('Item_MRP','Item_Outlet_Sales')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f41d0a8ec50>
```



```
from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler(feature_range=(0,1))
data[['Item_MRP','Item_Outlet_Sales']] = scaler.fit_transform(data[['Item_MRP','Ite
```

```
data[['Item_MRP','Item_Outlet_Sales']].head()
```

	Item_MRP	Item_Outlet_Sales
0	0.927507	0.283587
1	0.072068	0.031419
2	0.468288	0.158115
3	0.640093	0.053555
4	0.095805	0.073651

```
X1=data['Item_MRP'].values.reshape(-1,1)
X2=data['Item_Outlet_Sales'].values.reshape(-1,1)
```

```
X=np.concatenate((X1,X2),axis=1)
```

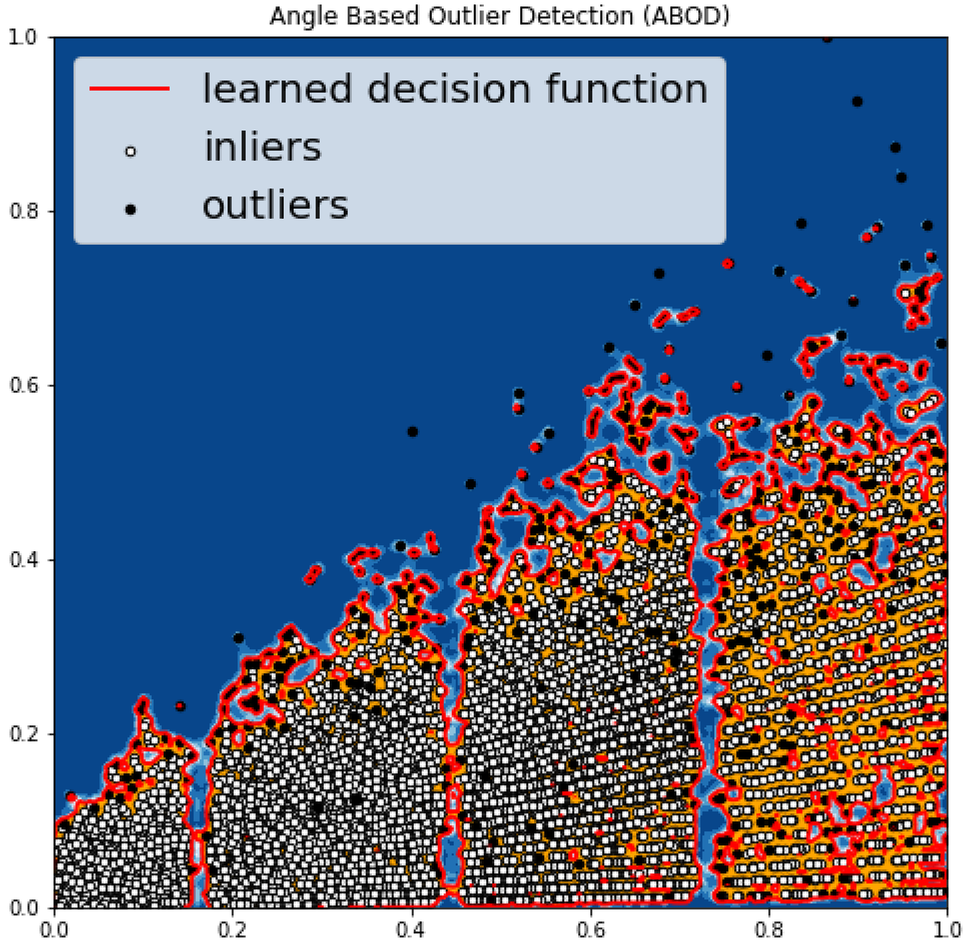
```
X
```

```
array([[0.92750715, 0.28358666],
       [0.0720684 , 0.03141895],
       [0.46828841, 0.15811486],
       ...,
       [0.22849221, 0.08885035],
       [0.30493925, 0.13883505],
       [0.18750976, 0.05610527]])
```

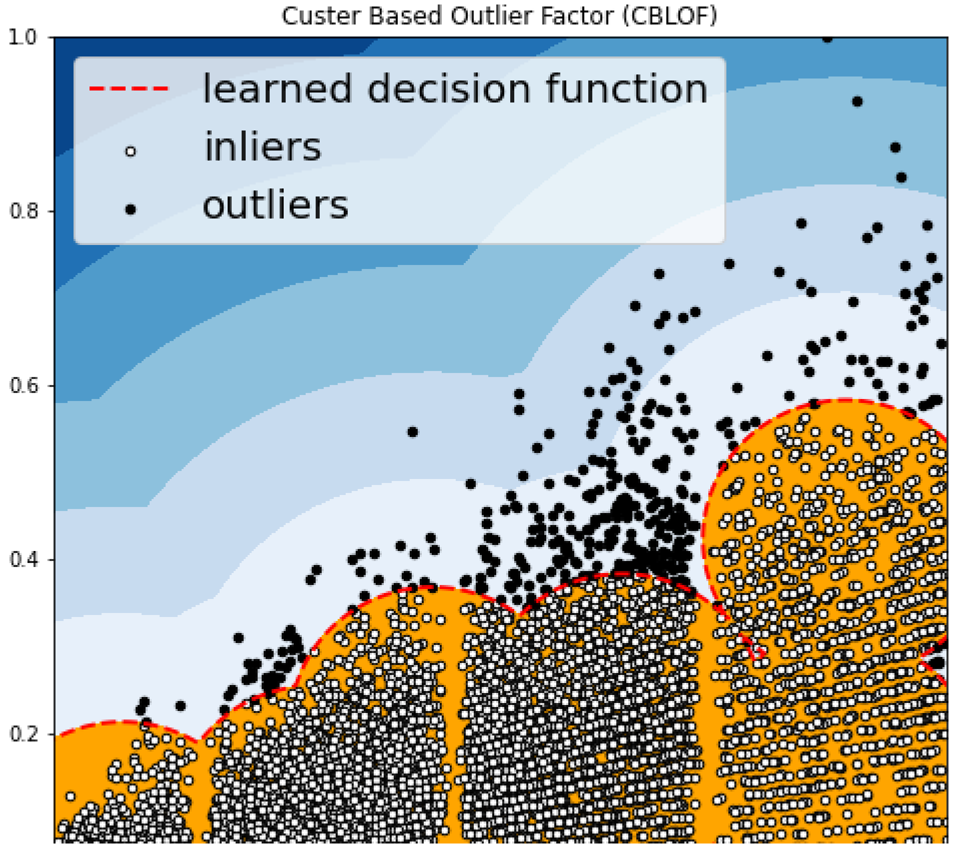
```
#import all the models
from pyod.models.abod import ABOD #(angle based outlier detection )
from pyod.models.cblof import CBLOF # cluster based
from pyod.models.hbos import HBOS
from pyod.models.lof import LOF
from pyod.models.mcd import MCD


random_state=np.random.RandomState(21)
outlier_fraction=0.05
classifiers={
    'Angle Based Outlier Detection (ABOD)':ABOD(contamination=outlier_fraction),
    'Custer Based Outlier Factor (CBLOF)': CBLOF(contamination=outlier_fraction, ch
    'Histogram Based Outlier Detection (HBOS)': HBOS(contamination=outlier_fraction
    'Local Outlier Factor(LOF)': LOF(n_neighbors=35,contamination=outlier_fraction)
    'Minimum Covariance Detection (MCD)': MCD(contamination=outlier_fraction,random
}


#creating a meshgrid for visualisation
xx,yy=np.meshgrid(np.linspace(0,1,200),np.linspace(0,1,200))



from scipy import stats
import matplotlib.pyplot as plt
import matplotlib.font_manager


from scipy import stats

import matplotlib.font_manager




for i, (clf_name,clf) in enumerate(classifiers.items()):

  #fitting the model

  clf.fit(X)

  #predict anomaly score

  scores_pred = clf.decision_function(X)*-1

  #prediction of a datapoint category outlier or inlier

  y_pred=clf.predict(X)

  n_inliers = len(y_pred)-np.count_nonzero(y_pred)
```

```
n_outliers = np.count_nonzero(y_pred==1)



plt.figure(figsize=(8,8))



dfx=data

dfx['outlier']=y_pred.tolist()




IX1 = np.array(dfx['Item_MRP'][dfx['outlier']==0]).reshape(-1,1)

IX2 = np.array(dfx['Item_Outlet_Sales'][dfx['outlier']==0]).reshape(-1,1)



OX1 = np.array(dfx['Item_MRP'][dfx['outlier']==1]).reshape(-1,1)

OX2 = np.array(dfx['Item_Outlet_Sales'][dfx['outlier']==1]).reshape(-1,1)
print('OUTLIERS : ', n_outliers,'INLIERS : ', n_inliers, clf_name)
threshold= stats.scoreatpercentile(scores_pred,100*outlier_fraction)
Z=clf.decision_function(np.c_[xx.ravel(),yy.ravel()])*-1
Z=Z.reshape(xx.shape)
plt.contourf(xx,yy,Z,levels=np.linspace(Z.min(), threshold,7),cmap=plt.cm.Blues_r
a=plt.contour(xx,yy,Z, levels=[threshold], linewidths=2, colors='red')
plt.contourf(xx,yy,Z, levels=[threshold,Z.max()], colors='orange')
b=plt.scatter(IX1,IX2, c='white', s=20,edgecolor='k')
c=plt.scatter(OX1,OX2, c='black', s=20,edgecolor='k')
plt.axis('tight')
plt.legend([a.collections[0],b,c],
           ['learned decision function', 'inliers', 'outliers'],
           prop=matplotlib.font_manager.FontProperties(size=20),
           loc=2)

plt.xlim((0,1))
plt.ylim((0,1))
plt.title(clf_name)
```

```
OUTLIERS :  447 INLIERS :  8076 Angle Based Outlier Detection (ABOD)
OUTLIERS :  427 INLIERS :  8096 Custer Based Outlier Factor (CBLOF)
OUTLIERS :  501 INLIERS :  8022 Histogram Based Outlier Detection (HBOS)
OUTLIERS :  392 INLIERS :  8131 Local Outlier Factor(LOF)
OUTLIERS :  427 INLIERS :  8096 Minimum Covariance Detection (MCD)
```



Angle Based Outlier Detection (ABOD)



Custer Based Outlier Factor (CBLOF)