Demo

February 27, 2019

0.1 Demo and Test

In this part, we test the Convext Hull 2D for some cases

```
In [1]: from ConvexHull import ConvexHull2D
        import numpy as np
        model = ConvexHull2D()
```

0.1.1 Test One: Invalid. input

```
In [2]: points = None
        model(points)
```

```
No valid convex hull is found! Please provide more than 3 unique points
```

0.1.2 Test Two: Invalid. input

```
In [3]: points = np.array([])
        model(points)
```

```
No valid convex hull is found! Please provide more than 3 unique points
```

0.1.3 Test Three: Invalid. input (insufficent amount of points)

```
In [12]: points = np.array([[1,2], [-4.9, 0.73]])
         model(points)
```

```
No valid convex hull is found! Please provide more than 3 unique points
```

0.1.4 Test Four: Invalid. input (points are actually on the same line)

```
In [13]: points = np.array([[1,2], [1, 3], [1,4]])
         convex_hull = model(points)
         print ("\nThe vertices of convex hull are\n")
         print (convex_hull)
```

```
<class 'Exception'>
('The input points are located on the same line. No convex hull is found!',)
<class 'Exception'>
('Not enough points are found for convex hull. Please check your input and other information',)

The vertices of convex hull are

None
```

0.1.5 Test Five: Simple case

```
In [5]: points = np.array([[0,0], [1,2], [-4.9, 0.73]])
        model(points)
```

```
Out[5]: array([[-4.9 ,  0.73],
               [ 0.  ,  0.  ],
               [ 1.  ,  2.  ]])
```

0.1.6 Test Six: General case

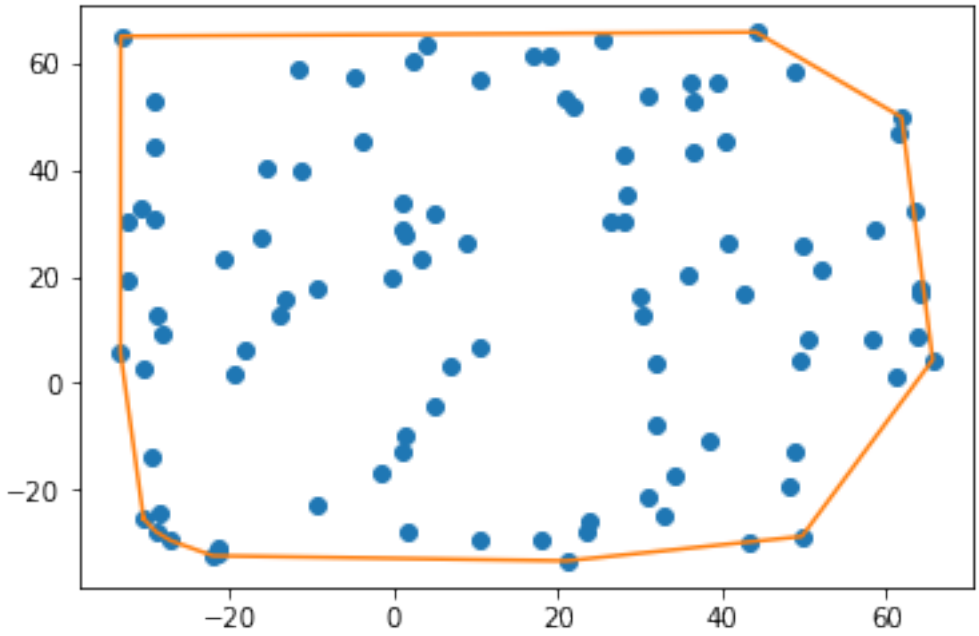
```
In [10]: import matplotlib.pyplot as plt

         points = np.random.random((100,2)) * 100 #draw from normal distribution and scale it
         points -= np.random.random() * 100 # test points with positive and negative coordinat
         plt.plot(points[:,0], points[:,1], 'o')

         convex_hull = model(points)
         print ("The vertices of convex hull are")
         print (convex_hull)

         #Close the hull during plot by making the head and tail as the same point
         convex_hull = np.vstack([convex_hull, convex_hull[0]])
         plt.plot(convex_hull[:,0], convex_hull[:,1])
         plt.show()
```

```
The vertices of convex hull are
[[-30.34478464 -25.49434852]
 [-28.80982052 -27.97168969]
 [-27.03556445 -29.62912708]
 [-21.92438044 -32.46525418]
 [ 21.12174928 -33.42353813]
 [ 49.75244411 -28.84468783]
 [ 65.68061666   4.33814033]
 [ 61.89855911  49.91768006]
 [ 44.16177304  65.88208679]
 [-33.12044019  65.10176138]
 [-33.15281795   5.67249395]]
```



0.1.7 Test Seven: Given a test set of points, check if they are inside the convex hull

```
In [7]: import matplotlib.pyplot as plt

        points = np.random.random((100,2)) * 100 #draw from normal distribution and scale it
        points -= np.random.random() * 100 # test points with positive and negative coordinate
        test_points = np.random.random((10,2)) * 100

        plt.plot(points[:,0], points[:,1], 'o')
        plt.plot(test_points[:,0], test_points[:,1], 'ro')
        convex_hull = model(points)
        print ("The vertices of convex hull are\n")
        print (convex_hull)

        #Close the hull during plot by making the head and tail as the same point
        convex_hull = np.vstack([convex_hull, convex_hull[0]])
        plt.plot(convex_hull[:,0], convex_hull[:,1])

        isInside = model.isInside(test_points)
        print ("\n")
        print ("Testing the model with random test set")
        for point, status in zip(test_points, isInside):
            print("point: ", point, " status: ", status )

        plt.show()
```

```
The vertices of convex hull are

[[-24.43333595 -10.16025703]
 [-18.28298976 -18.63656776]
 [-14.7729698  -21.6422585 ]
 [  1.32096596 -23.79991096]
 [ 56.0300138  -23.88969908]
 [ 70.7858042  -20.71260849]
 [ 74.88493107  39.03853099]
 [ 70.62610472  73.48114179]
 [  4.64657323  74.66695145]
 [-23.28574506  68.39594002]
 [-24.08897784  57.55102279]]


Testing the model with random test set
point:  [71.93988592 55.35013509]  status:  True
point:  [ 6.29908119 26.5605536 ]  status:  True
point:  [66.47244996 61.7487539 ]  status:  True
point:  [ 5.38516047 18.65902789]  status:  True
point:  [15.64034429  6.01978347]  status:  True
point:  [58.91889175 13.6917658 ]  status:  True
point:  [16.38792149 64.03859936]  status:  True
point:  [20.88248563 41.23138062]  status:  True
point:  [34.69632593 71.56028955]  status:  True
point:  [ 6.76041777 87.9044422 ]  status:  False
```