modified_final

May 1, 2018

```
In [3]: %pylab inline
```

```
Populating the interactive namespace from numpy and matplotlib
```

```
In [4]: from sklearn.manifold import TSNE
```

```
In [5]: # Loading the iris datset
        from sklearn.datasets import load_iris
        iris = load_iris()
```

```
In [8]: iris.keys()
```

```
Out[8]: dict_keys(['data', 'target', 'target_names', 'DESCR', 'feature_names'])
```

```
In [9]: iris.target_names
```

```
Out[9]: array(['setosa', 'versicolor', 'virginica'], dtype='<U10')
```

```
In [10]: iris.feature_names
```

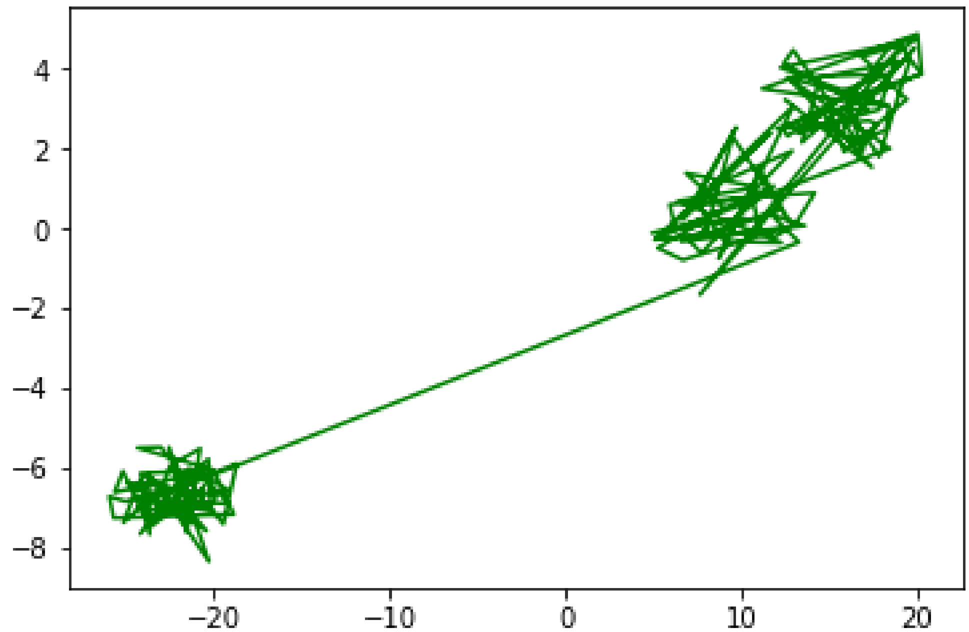
```
Out[10]: ['sepal length (cm)',
          'sepal width (cm)',
          'petal length (cm)',
          'petal width (cm)']
```

```
In [11]: iris.target
```

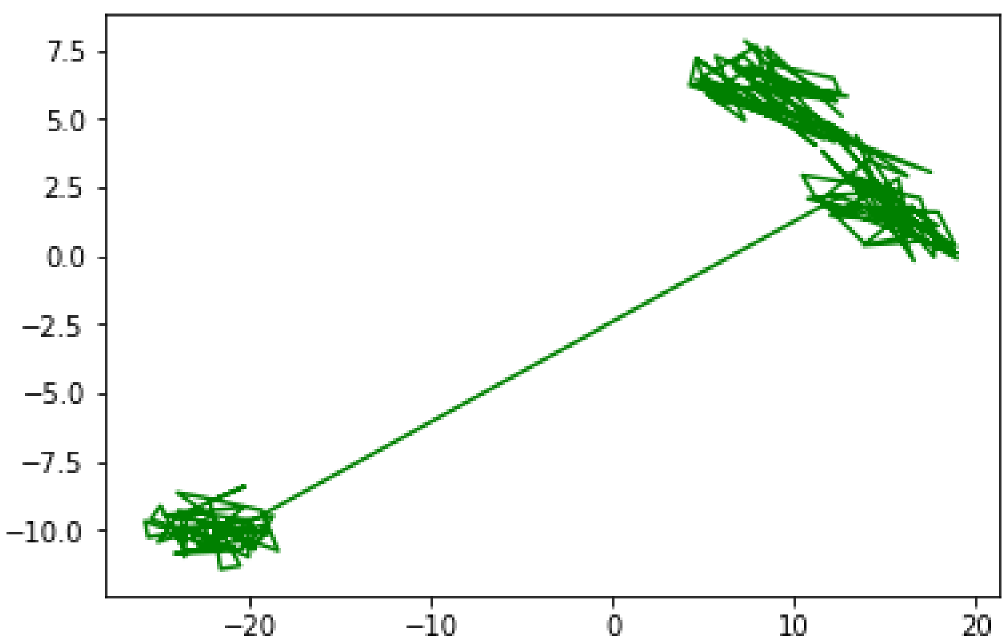
```
Out[11]: array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
                1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
                1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
                2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
                2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2])
```

```
In [28]: from sklearn.manifold import TSNE
         X = np.array([[0, 0, 0], [0, 1, 1], [1, 0, 1], [1, 1, 1]])
         # X_embedded = TSNE(n_components=2).fit_transform(iris.data)
         X_embedded = TSNE(learning_rate = 100).fit_transform(iris.data)
         X_embedded.shape
```

```
Out[28]: (150, 2)

In [29]: plt.plot( X_embedded[:,0], X_embedded[:,1] , 'g')

Out[29]: [<matplotlib.lines.Line2D at 0x166b2f28358>]
```

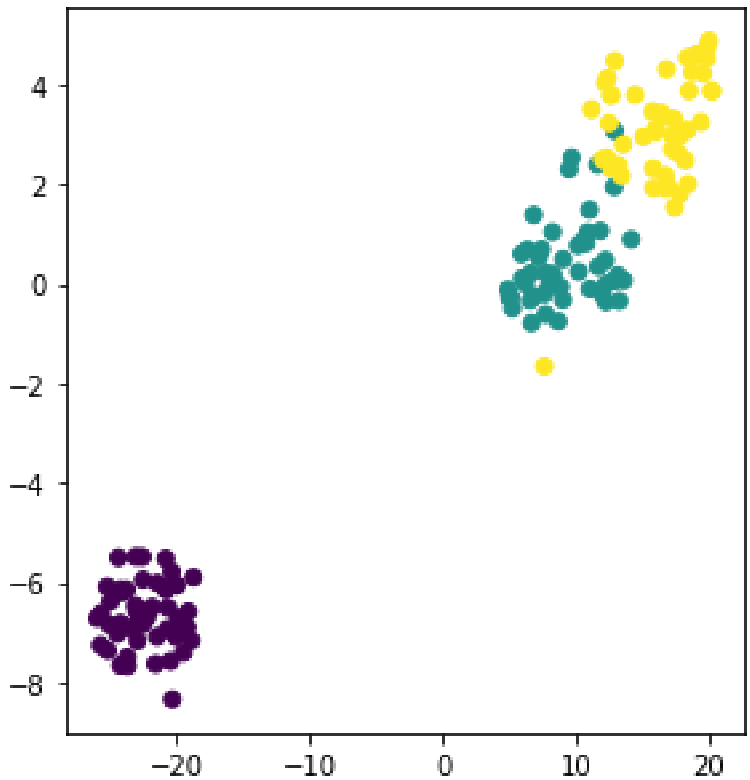


```
In [26]: plt.plot( X_embedded[:,0], X_embedded[:,1] , 'g')

Out[26]: [<matplotlib.lines.Line2D at 0x166b2ec8438>]
```



In [38]:
```
figure(figsize = (10,5) )
subplot(121)
scatter(X_embedded[:,0], X_embedded[:,1], c = iris.target  )
```

Out[38]: <matplotlib.collections.PathCollection at 0x166b326beb8>



In [46]:
```
x_min = X_embedded[:,0].min()
```

In [47]:
```
x_max = X_embedded[:,0].max()
```

In [48]:
```
y_min = X_embedded[:,1].min()
y_max = X_embedded[:,1].max()
```

In [49]:
```
Limts = [ [ x_min , x_max] , [y_min, y_max] ]
```

In [50]:
```
Limts
```

Out[50]: [[-25.9487, 20.243372], [-8.339147, 4.883639]]

In [89]:
```
def find_good_size(Limits,windowSize):
    x_min = Limits[0][0]
    x_max = Limits[0][1]
    y_min = Limits[1][0]
    y_max = Limits[1][1]
    from math import floor , ceil
    # Clean the matrix
    x_min = floor(x_min)
    x_max = ceil(x_max)
    y_min = floor(y_min)
    y_max = ceil(y_max)

    # Printing the cleaend up values.
    print(x_min,x_max,y_min, y_max)

    # Fitting the size to handle the windows
    a = (x_max - x_min) % windowSize
    b = (y_max - y_min) % windowSize

    if ( (x_max + a) % windowSize == 0):
        x_max = x_max + a
    else:
        x_max = x_max + (windowSize - a)

    if ( (y_max + b) % windowSize == 0):
        y_max = y_max + b
    else:
        y_max = y_max + (windowSize - b)

    print("You Can use this matrix now.")
    print("X_min=",x_min)
    print("X_max=",x_max)
    print("Y_min=",y_min)
    print("Y_max=",y_max)
    print("And the Matrix Shape:", (x_max - x_min) , " * ", (y_max - y_min))
    return [[x_min,x_max],[y_min,y_max]]
```

In [90]:
```
find_good_size(Limts,1)
```

```
-26 21 -9 5
You Can use this matrix now.
X_min= -26
X_max= 21
Y_min= -9
Y_max= 5
And the Matrix Shape: 47  *  14
```

```
Out[90]: [[-26, 21], [-9, 5]]

In [66]: find_good_size(Limts,10)

-26 21 -9 5
You Can use this matrix now.
X_min= -26
X_max= 24
Y_min= -9
Y_max= 11
And the Matrix Shape: 50  *  20


Out[66]: [[-26, 24], [-9, 11]]

In [91]: def tell_windows(Limts,windowSize):
             g = find_good_size(Limts,windowSize)
             # Going Row Wise:
             count = 0
             for i in range(g[0][0],g[0][1],windowSize):
                 for j in range(g[1][0],g[1][1],windowSize):
                     count += 1
                     print("<Window:",count,">","X:[", i, ",", i +windowSize , "]" , "Y:[", j,
             print("Total Count:",count)
             ans = (g[0][1]-g[0][0])*(g[1][1]-g[1][0]) // (windowSize ** 2)
             print("Expected Count:",ans)
             print("OKAY:", count == ans )

In [93]: tell_windows([[0,3],[0,3]],1)

0 3 0 3
You Can use this matrix now.
X_min= 0
X_max= 3
Y_min= 0
Y_max= 3
And the Matrix Shape: 3  *  3
<Window: 1 > X:[ 0 , 1 ] Y:[ 0 , 1 ]
<Window: 2 > X:[ 0 , 1 ] Y:[ 1 , 2 ]
<Window: 3 > X:[ 0 , 1 ] Y:[ 2 , 3 ]
<Window: 4 > X:[ 1 , 2 ] Y:[ 0 , 1 ]
<Window: 5 > X:[ 1 , 2 ] Y:[ 1 , 2 ]
<Window: 6 > X:[ 1 , 2 ] Y:[ 2 , 3 ]
<Window: 7 > X:[ 2 , 3 ] Y:[ 0 , 1 ]
<Window: 8 > X:[ 2 , 3 ] Y:[ 1 , 2 ]
<Window: 9 > X:[ 2 , 3 ] Y:[ 2 , 3 ]
Total Count: 9
Expected Count: 9
OKAY: True
```

In [94]:
```
tell_windows([[0,3],[0,3]],3)
```

```
0 3 0 3
You Can use this matrix now.
X_min= 0
X_max= 3
Y_min= 0
Y_max= 3
And the Matrix Shape: 3  *  3
<Window: 1 > X:[ 0 , 3 ] Y:[ 0 , 3 ]
Total Count: 1
Expected Count: 1
OKAY: True
```

In [95]:
```
tell_windows([[0,3],[0,3]],5)
```

```
0 3 0 3
You Can use this matrix now.
X_min= 0
X_max= 5
Y_min= 0
Y_max= 5
And the Matrix Shape: 5  *  5
<Window: 1 > X:[ 0 , 5 ] Y:[ 0 , 5 ]
Total Count: 1
Expected Count: 1
OKAY: True
```

In [77]:
```
Limts
```

Out[77]:
```
[[-25.9487, 20.243372], [-8.339147, 4.883639]]
```

In [103]:
```
# Okay So now We have a function which finds the windows we just now need to calcula
def calEntropy(window,label):
    'Takes: [x_min, x_max],[y_min, y_max]'
    'returns entropy of the window'
    x_min, x_max , y_min, y_max = window
    # Set up entropy to be zero
    en = 0
    from math import log
    c = [0,0,0]
    # Calculate the number of samples in this window.
    # X_embedded stores the data.
    for i in range(150):
        cx = X_embedded[i][0]
        cy = X_embedded[i][1]
        if x_min <= cx <=x_max and y_min <= cy <= y_max:
```

```
                print("Found a sample of class:",label[i])
                c[label[i]] += 1
        t = c[0] + c[1] + c[2]
        for i in c:
            if i!= 0:
                en += - ( i/t * log(i/t) )
        print("Found Entropy:",en)
        return en
```

```
In [109]: def tell_windows(Limts,windowSize,target):
              g = find_good_size(Limts,windowSize)
              # Going Row Wise:
              total_ent = 0
              count = 0
              for i in range(g[0][0],g[0][1],windowSize):
                  for j in range(g[1][0],g[1][1],windowSize):
                      count += 1
                      print("<Window:",count,">","X:[", i, ",", i +windowSize , "]" , "Y:[", j
                      total_ent += calEntropy([i,windowSize+i, j, windowSize+j], target)
              print("Total Count:",count)
              ans = (g[0][1]-g[0][0])*(g[1][1]-g[1][0]) // (windowSize ** 2)
              print("Expected Count:",ans)
              print("OKAY:", count == ans )
              print("-----Done----",total_ent)
              return total_ent
```

```
In [110]: tell_windows(Limts,3, add_noise(0))
```

```
Plotting the data with Noise =>  0  %
Done!
-26 21 -9 5
You Can use this matrix now.
X_min= -26
X_max= 22
Y_min= -9
Y_max= 6
And the Matrix Shape: 48  *  15
<Window: 1 > X:[ -26 , -23 ] Y:[ -9 , -6 ]
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
```

```
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found Entropy: 0.0
<Window: 2 > X:[ -26 , -23 ] Y:[ -6 , -3 ]
Found a sample of class: 0
Found a sample of class: 0
Found Entropy: 0.0
<Window: 3 > X:[ -26 , -23 ] Y:[ -3 , 0 ]
Found Entropy: 0
<Window: 4 > X:[ -26 , -23 ] Y:[ 0 , 3 ]
Found Entropy: 0
<Window: 5 > X:[ -26 , -23 ] Y:[ 3 , 6 ]
Found Entropy: 0
<Window: 6 > X:[ -23 , -20 ] Y:[ -9 , -6 ]
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found Entropy: 0.0
<Window: 7 > X:[ -23 , -20 ] Y:[ -6 , -3 ]
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found Entropy: 0.0
<Window: 8 > X:[ -23 , -20 ] Y:[ -3 , 0 ]
Found Entropy: 0
```

```
<Window: 9 > X:[ -23 , -20 ] Y:[ 0 , 3 ]
Found Entropy: 0
<Window: 10 > X:[ -23 , -20 ] Y:[ 3 , 6 ]
Found Entropy: 0
<Window: 11 > X:[ -20 , -17 ] Y:[ -9 , -6 ]
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found a sample of class: 0
Found Entropy: 0.0
<Window: 12 > X:[ -20 , -17 ] Y:[ -6 , -3 ]
Found a sample of class: 0
Found Entropy: 0.0
<Window: 13 > X:[ -20 , -17 ] Y:[ -3 , 0 ]
Found Entropy: 0
<Window: 14 > X:[ -20 , -17 ] Y:[ 0 , 3 ]
Found Entropy: 0
<Window: 15 > X:[ -20 , -17 ] Y:[ 3 , 6 ]
Found Entropy: 0
<Window: 16 > X:[ -17 , -14 ] Y:[ -9 , -6 ]
Found Entropy: 0
<Window: 17 > X:[ -17 , -14 ] Y:[ -6 , -3 ]
Found Entropy: 0
<Window: 18 > X:[ -17 , -14 ] Y:[ -3 , 0 ]
Found Entropy: 0
<Window: 19 > X:[ -17 , -14 ] Y:[ 0 , 3 ]
Found Entropy: 0
<Window: 20 > X:[ -17 , -14 ] Y:[ 3 , 6 ]
Found Entropy: 0
<Window: 21 > X:[ -14 , -11 ] Y:[ -9 , -6 ]
Found Entropy: 0
<Window: 22 > X:[ -14 , -11 ] Y:[ -6 , -3 ]
Found Entropy: 0
<Window: 23 > X:[ -14 , -11 ] Y:[ -3 , 0 ]
Found Entropy: 0
<Window: 24 > X:[ -14 , -11 ] Y:[ 0 , 3 ]
Found Entropy: 0
<Window: 25 > X:[ -14 , -11 ] Y:[ 3 , 6 ]
Found Entropy: 0
<Window: 26 > X:[ -11 , -8 ] Y:[ -9 , -6 ]
Found Entropy: 0
<Window: 27 > X:[ -11 , -8 ] Y:[ -6 , -3 ]
Found Entropy: 0
<Window: 28 > X:[ -11 , -8 ] Y:[ -3 , 0 ]
Found Entropy: 0
<Window: 29 > X:[ -11 , -8 ] Y:[ 0 , 3 ]
```

```
Found Entropy: 0
<Window: 30 > X:[ -11 , -8 ] Y:[ 3 , 6 ]
Found Entropy: 0
<Window: 31 > X:[ -8 , -5 ] Y:[ -9 , -6 ]
Found Entropy: 0
<Window: 32 > X:[ -8 , -5 ] Y:[ -6 , -3 ]
Found Entropy: 0
<Window: 33 > X:[ -8 , -5 ] Y:[ -3 , 0 ]
Found Entropy: 0
<Window: 34 > X:[ -8 , -5 ] Y:[ 0 , 3 ]
Found Entropy: 0
<Window: 35 > X:[ -8 , -5 ] Y:[ 3 , 6 ]
Found Entropy: 0
<Window: 36 > X:[ -5 , -2 ] Y:[ -9 , -6 ]
Found Entropy: 0
<Window: 37 > X:[ -5 , -2 ] Y:[ -6 , -3 ]
Found Entropy: 0
<Window: 38 > X:[ -5 , -2 ] Y:[ -3 , 0 ]
Found Entropy: 0
<Window: 39 > X:[ -5 , -2 ] Y:[ 0 , 3 ]
Found Entropy: 0
<Window: 40 > X:[ -5 , -2 ] Y:[ 3 , 6 ]
Found Entropy: 0
<Window: 41 > X:[ -2 , 1 ] Y:[ -9 , -6 ]
Found Entropy: 0
<Window: 42 > X:[ -2 , 1 ] Y:[ -6 , -3 ]
Found Entropy: 0
<Window: 43 > X:[ -2 , 1 ] Y:[ -3 , 0 ]
Found Entropy: 0
<Window: 44 > X:[ -2 , 1 ] Y:[ 0 , 3 ]
Found Entropy: 0
<Window: 45 > X:[ -2 , 1 ] Y:[ 3 , 6 ]
Found Entropy: 0
<Window: 46 > X:[ 1 , 4 ] Y:[ -9 , -6 ]
Found Entropy: 0
<Window: 47 > X:[ 1 , 4 ] Y:[ -6 , -3 ]
Found Entropy: 0
<Window: 48 > X:[ 1 , 4 ] Y:[ -3 , 0 ]
Found Entropy: 0
<Window: 49 > X:[ 1 , 4 ] Y:[ 0 , 3 ]
Found Entropy: 0
<Window: 50 > X:[ 1 , 4 ] Y:[ 3 , 6 ]
Found Entropy: 0
<Window: 51 > X:[ 4 , 7 ] Y:[ -9 , -6 ]
Found Entropy: 0
<Window: 52 > X:[ 4 , 7 ] Y:[ -6 , -3 ]
Found Entropy: 0
<Window: 53 > X:[ 4 , 7 ] Y:[ -3 , 0 ]
```

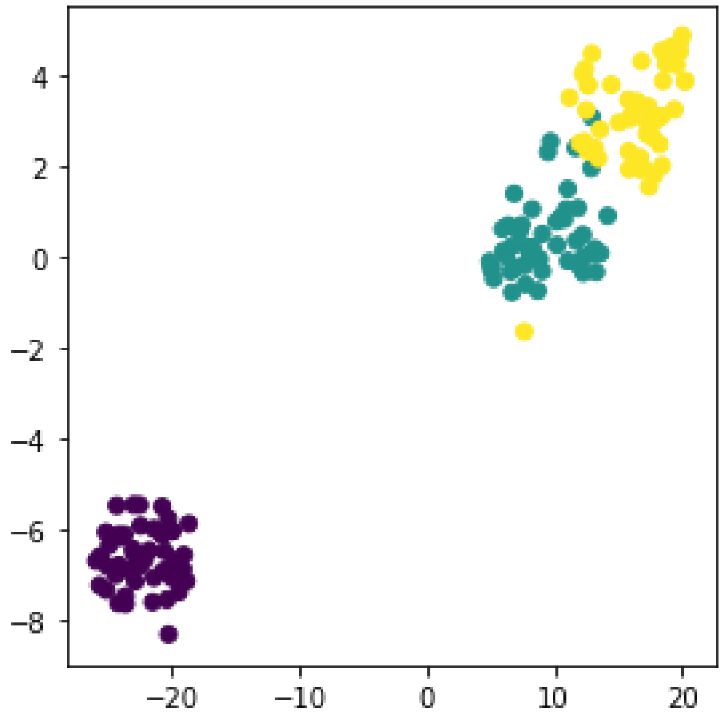
```
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found Entropy: 0.0
<Window: 54 > X:[ 4 , 7 ] Y:[ 0 , 3 ]
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found Entropy: 0.0
<Window: 55 > X:[ 4 , 7 ] Y:[ 3 , 6 ]
Found Entropy: 0
<Window: 56 > X:[ 7 , 10 ] Y:[ -9 , -6 ]
Found Entropy: 0
<Window: 57 > X:[ 7 , 10 ] Y:[ -6 , -3 ]
Found Entropy: 0
<Window: 58 > X:[ 7 , 10 ] Y:[ -3 , 0 ]
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 2
Found Entropy: 0.410116318288409
<Window: 59 > X:[ 7 , 10 ] Y:[ 0 , 3 ]
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found Entropy: 0.0
<Window: 60 > X:[ 7 , 10 ] Y:[ 3 , 6 ]
Found Entropy: 0
<Window: 61 > X:[ 10 , 13 ] Y:[ -9 , -6 ]
Found Entropy: 0
<Window: 62 > X:[ 10 , 13 ] Y:[ -6 , -3 ]
```

```
Found Entropy: 0
<Window: 63 > X:[ 10 , 13 ] Y:[ -3 , 0 ]
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found Entropy: 0.0
<Window: 64 > X:[ 10 , 13 ] Y:[ 0 , 3 ]
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found Entropy: 0.48257756517701206
<Window: 65 > X:[ 10 , 13 ] Y:[ 3 , 6 ]
Found a sample of class: 1
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found Entropy: 0.37677016125643675
<Window: 66 > X:[ 13 , 16 ] Y:[ -9 , -6 ]
Found Entropy: 0
<Window: 67 > X:[ 13 , 16 ] Y:[ -6 , -3 ]
Found Entropy: 0
<Window: 68 > X:[ 13 , 16 ] Y:[ -3 , 0 ]
Found a sample of class: 1
Found Entropy: 0.0
<Window: 69 > X:[ 13 , 16 ] Y:[ 0 , 3 ]
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 1
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
```

```
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found Entropy: 0.6365141682948128
<Window: 70 > X:[ 13 , 16 ] Y:[ 3 , 6 ]
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found Entropy: 0.0
<Window: 71 > X:[ 16 , 19 ] Y:[ -9 , -6 ]
Found Entropy: 0
<Window: 72 > X:[ 16 , 19 ] Y:[ -6 , -3 ]
Found Entropy: 0
<Window: 73 > X:[ 16 , 19 ] Y:[ -3 , 0 ]
Found Entropy: 0
<Window: 74 > X:[ 16 , 19 ] Y:[ 0 , 3 ]
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found Entropy: 0.0
<Window: 75 > X:[ 16 , 19 ] Y:[ 3 , 6 ]
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found Entropy: 0.0
<Window: 76 > X:[ 19 , 22 ] Y:[ -9 , -6 ]
Found Entropy: 0
<Window: 77 > X:[ 19 , 22 ] Y:[ -6 , -3 ]
Found Entropy: 0
<Window: 78 > X:[ 19 , 22 ] Y:[ -3 , 0 ]
Found Entropy: 0
<Window: 79 > X:[ 19 , 22 ] Y:[ 0 , 3 ]
Found Entropy: 0
```

```
<Window: 80 > X:[ 19 , 22 ] Y:[ 3 , 6 ]
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found a sample of class: 2
Found Entropy: 0.0
Total Count: 80
Expected Count: 80
OKAY: True
-----Done---- 1.9059782130166705
```

Out[110]: 1.9059782130166705



In [113]:
```
def add_noise(percentage):
    'Adds the given percentage of noise to the iris data and returns the new data'
    # Original
    target = [0 for i in range(50)] + [ 1 for i in range(50)] + [ 2 for i in range(5
    # 0   - 50   : class 0
```

```
            # 50 - 100   : class 1
            # 100 - 150  : class 2
            # We will change the labels in a class label randomly
            import random
            # Offset for percentage => (window size)* precentage
            offset = 50 * percentage // 100
            for i in range(0, 0 + offset ):
                target[i] = random.choice([1,2])
            for i in range(50, 50 + offset):
                target[i] = random.choice([0,2])
            for i in range(100,100 + offset):
                target[i] = random.choice([0,1])

            # Got the new labeled data
            # Now we can plot the tsne with this.
            #print("Plotting the data with Noise => ", percentage , " %")
            figure( figsize = (20,10))
            subplot(242)
            scatter(X_embedded[:,0], X_embedded[:,1], c = target)
            print("Done!")
            return target
```

```
In [115]: def main(windowSize, noise):
              return tell_windows(Limts,windowSize,add_noise(noise))
```

```
In [ ]: xxxx = []
        yyyy = []
        for i in range(0,100,10):
            xxxx.append(i)
            yyyy.append(main(3,i))
```

```
In [118]: xxxx
```

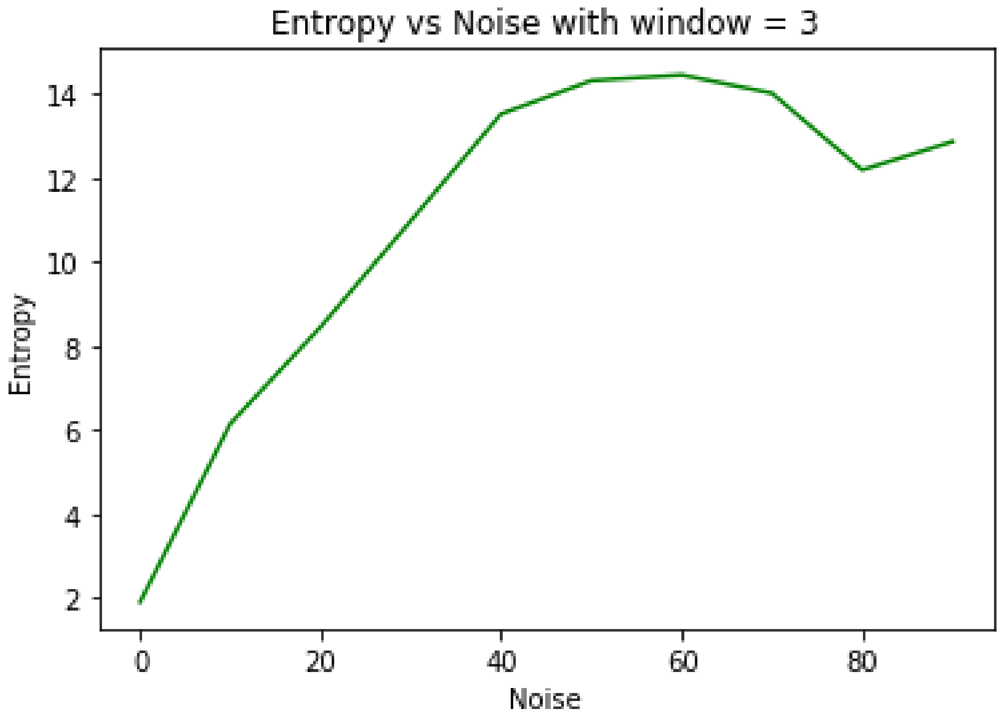
```
Out[118]: [0, 10, 20, 30, 40, 50, 60, 70, 80, 90]
```

```
In [119]: yyyy
```

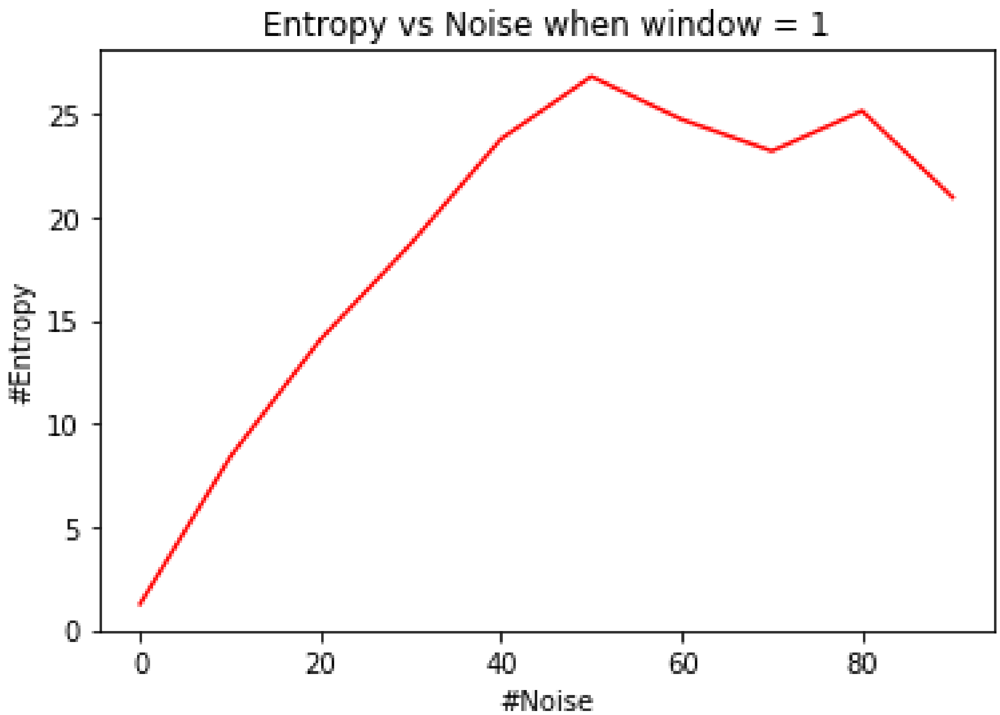
```
Out[119]: [1.9059782130166705,
           5.833869503703472,
           8.146882998185937,
           10.507330121040468,
           13.927450011307299,
           14.301825684532538,
           12.619612365784652,
           15.087570013154586,
           13.689747163330022,
           13.465812334559693]
```

```
In [120]: plt.plot(x,y,'g')
          plt.xlabel('Noise')
```

```
plt.ylabel('Entropy')
plt.title('Entropy vs Noise with window = 3')
plt.show()
```




```
In [ ]: xxxx = []
        yyyy = []
        for i in range(0,100,10):
            xxxx.append(i)
            yyyy.append(main(1,i))

In [122]: plt.plot(xxxx,yyyy,'r')
          plt.xlabel('#Noise')
          plt.ylabel('#Entropy')
          plt.title('Entropy vs Noise when window = 1')
          plt.show()
```



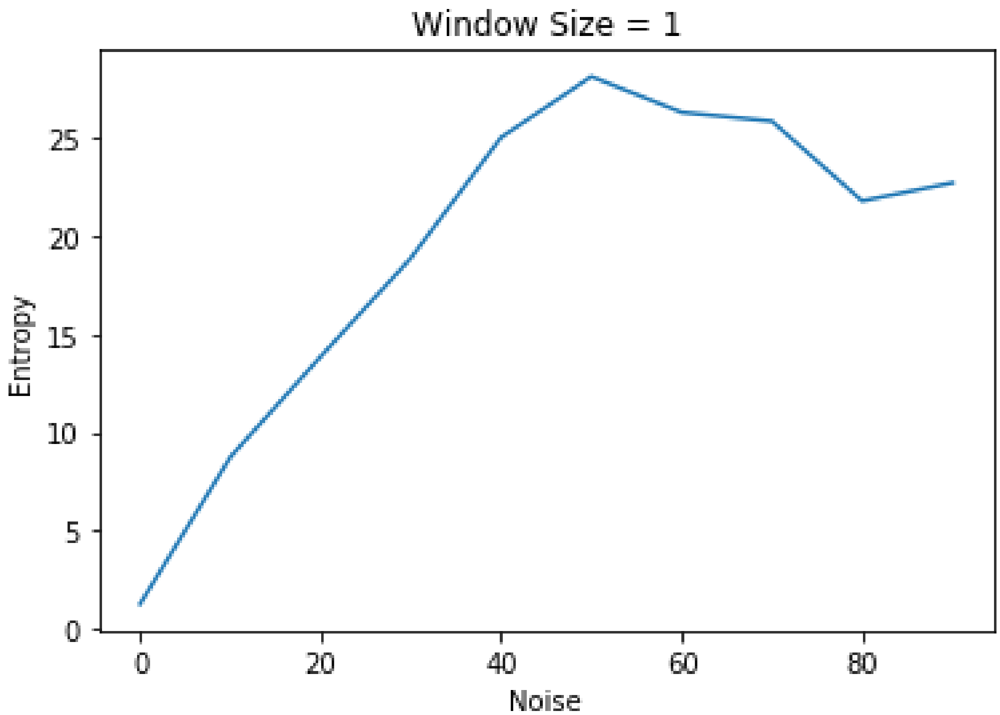

```
In [ ]: dataSet= [ [[],[]], [[],[]], [[],[]], [[],[]] , [[],[]], [[],[]] ]
        for window in [1,3,5]:
            for i in range(0,100,10):
                dataSet[window][0].append(i)
                dataSet[window][1].append(main(window,i))

In [126]: dataSet

Out[126]: [[[], []],
           [[0, 10, 20, 30, 40, 50, 60, 70, 80, 90],
            [1.2554823251787535,
             8.73851878067532,
             13.853681503172105,
             18.878635726235473,
             25.02940503282032,
             28.127269243771856,
             26.30719326841124,
             25.858124148322695,
             21.792007983531576,
             22.716655925390782]],
           [[], []],
           [[0, 10, 20, 30, 40, 50, 60, 70, 80, 90],
            [1.9059782130166705,
             5.906832364815045,
```

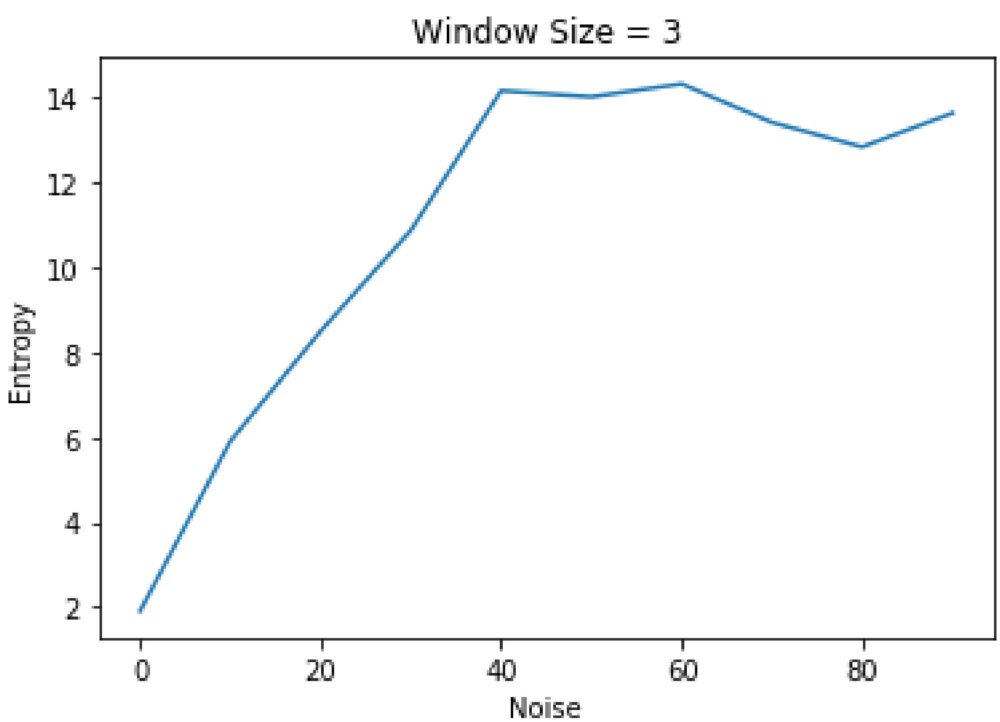
```
              8.501479237313205,
              10.879963858860972,
              14.168158576926993,
              14.028571586277163,
              14.32908036737314,
              13.426159812900783,
              12.847992308871694,
              13.64677921343577]],
            [[], []],
            [[0, 10, 20, 30, 40, 50, 60, 70, 80, 90],
             [0.8377336454040922,
              3.1059009623027247,
              4.348245698116212,
              6.66788264811653,
              7.452498242994488,
              7.6894883073824065,
              7.321481995767062,
              7.438085109160104,
              6.891550087350982,
              7.177780753797773]]]
```

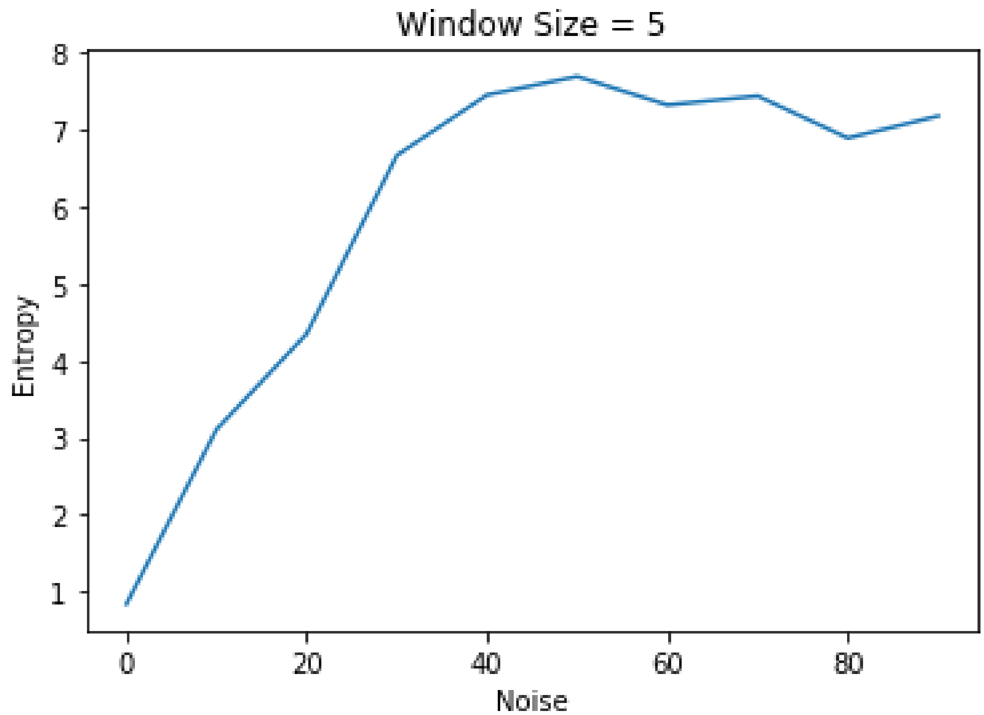
In [131]:
```
print("The final plots are")
def plotter(i):
    plt.plot(dataSet[i][0],dataSet[i][1])
    plt.title('Window Size = ' + str(i))
    plt.xlabel('Noise')
    plt.ylabel('Entropy')
```

```
The final plots are
```

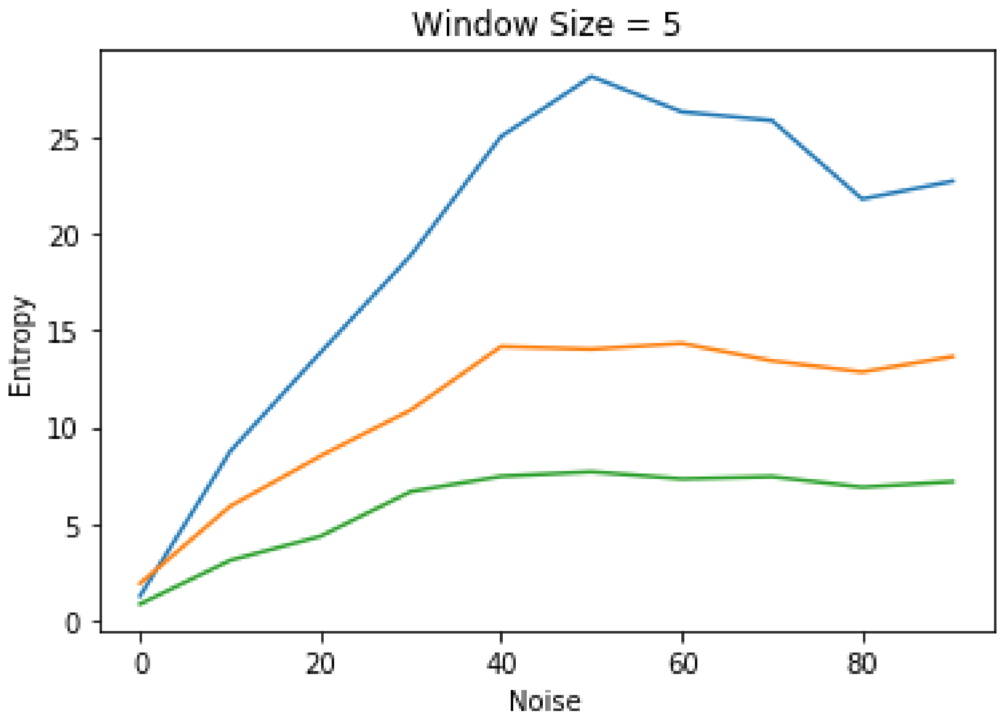
In [130]:
```
plotter(1)
plotter(3)
plotter(5)
```


Window Size = 1
Entropy
Noise
0
5
10
15
20
25
0
20
40
60
80


Window Size = 3
Entropy
Noise
2
4
6
8
10
12
14
0
20
40
60
80




```
In [132]: plotter(1)
          plotter(3)
          plotter(5)
          plt.show()
```




```
In [133]: # Done !
```