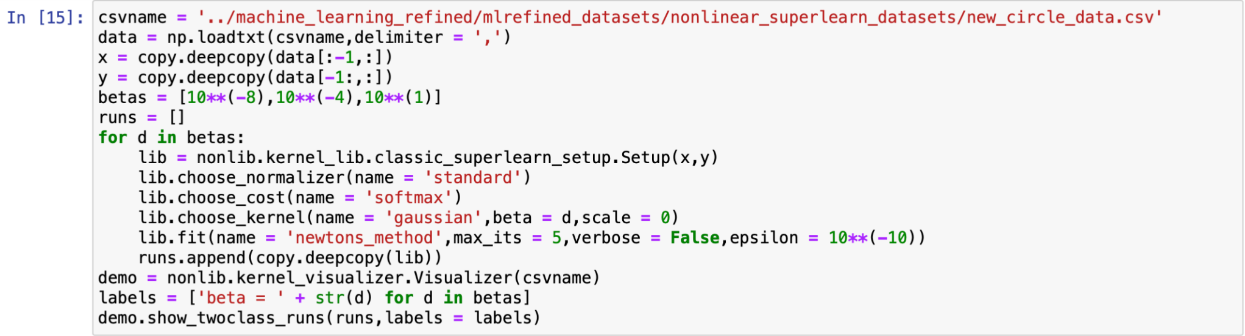
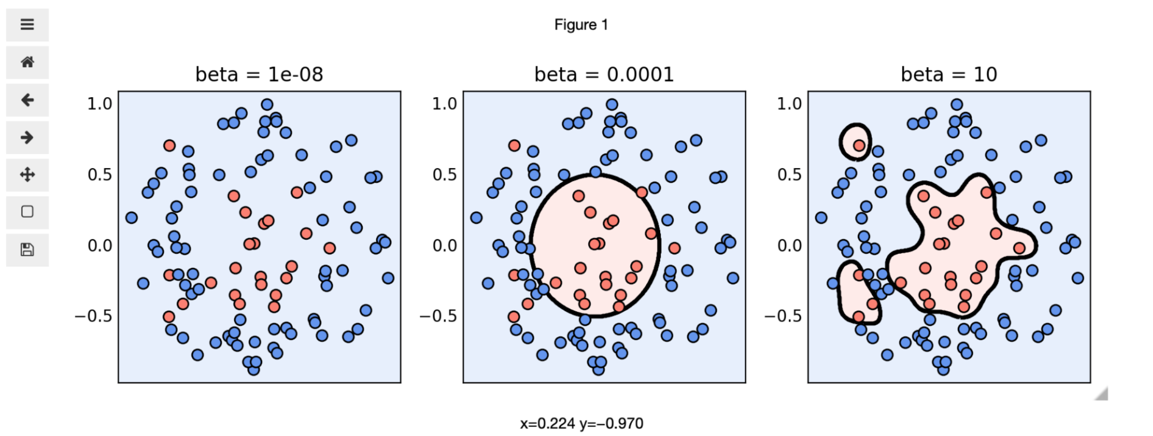
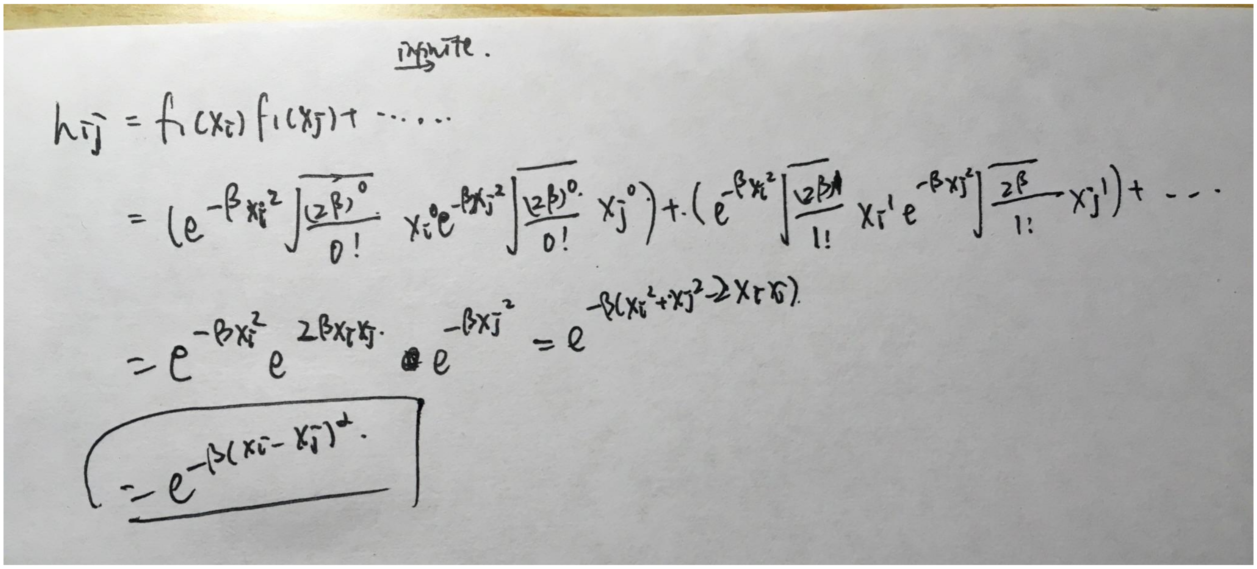
12.7



```
In [15]: csvname = '../machine_learning_refined/mlrefined_datasets/nonlinear_superlearn_datasets/new_circle_data.csv'
         data = np.loadtxt(csvname,delimiter = ',')
         x = copy.deepcopy(data[:-1,:])
         y = copy.deepcopy(data[-1:,:])
         betas = [10**(-8),10**(-4),10**(1)]
         runs = []
         for d in betas:
             lib = nonlib.kernel_lib.classic_superlearn_setup.Setup(x,y)
             lib.choose_normalizer(name = 'standard')
             lib.choose_cost(name = 'softmax')
             lib.choose_kernel(name = 'gaussian',beta = d,scale = 0)
             lib.fit(name = 'newtons_method',max_its = 5,verbose = False,epsilon = 10**(-10))
             runs.append(copy.deepcopy(lib))
         demo = nonlib.kernel_visualizer.Visualizer(csvname)
         labels = ['beta = ' + str(d) for d in betas]
         demo.show_twoclass_runs(runs,labels = labels)
```



12.10



infinite.

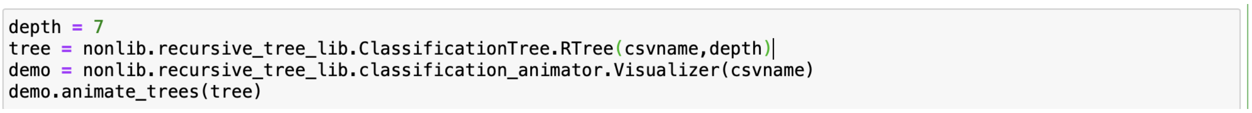
$$h_{ij} = f_i(x_i) f_i(x_j) + \cdots\cdots$$

$$= \left(e^{-\beta x_i^2}\sqrt{\frac{(2\beta)^0}{0!}}\, x_i^0 e^{-\beta x_j^2}\sqrt{\frac{(2\beta)^0}{0!}}\, x_j^0\right) + \left(e^{-\beta x_i^2}\sqrt{\frac{(2\beta)}{1!}}\, x_i^1 e^{-\beta x_j^2}\sqrt{\frac{2\beta}{1!}}\, x_j^1\right) + \cdots$$

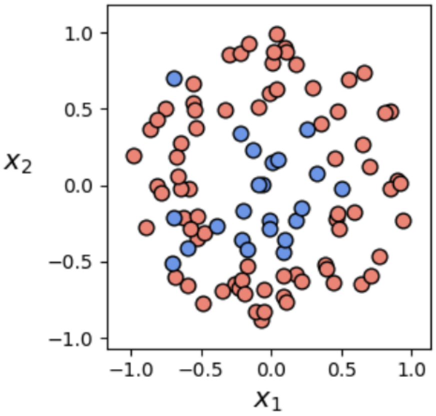
$$= e^{-\beta x_i^2} e^{2\beta x_i x_j} \cdot e^{-\beta x_j^2} = e^{-\beta(x_i^2+x_j^2-2x_i x_j)}$$

$$\boxed{= e^{-\beta(x_i-x_j)^2}.}$$

14.4

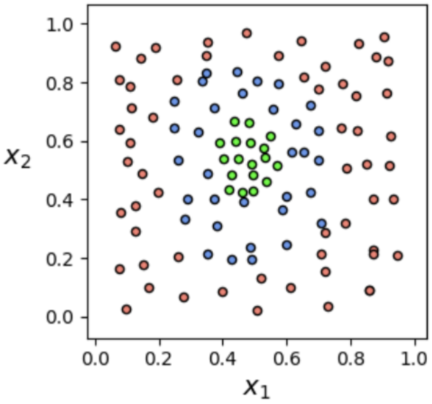


```
depth = 7
tree = nonlib.recursive_tree_lib.ClassificationTree.RTree(csvname,depth)
demo = nonlib.recursive_tree_lib.classification_animator.Visualizer(csvname)
demo.animate_trees(tree)
```



14.5

```
csvname = '../machine_learning_refined/mlrefined_datasets/nonlinear_superlearn_datasets/3_layercake_data.csv'
depth = 7
tree = nonlib.recursive_tree_lib.ClassificationTree.RTree(csvname,depth)
demo = nonlib.recursive_tree_lib.classification_animator.Visualizer(csvname)
demo.animate_trees(tree,pt_size = 20)
```



14.8

```
csvname = '../machine_learning_refined/mlrefined_datasets/nonlinear_superlearn_datasets/new_circle_data.csv'
trees = []
num_trees = 5
depth = 7
train_portion = 0.66
for i in range(num_trees):
tree = nonlib.recursive_tree_lib_crossval.ClassificationTree.RTree(csvname,de
pth,train_portion=train_portion)
trees.append(tree)
animator = nonlib.recursive_tree_lib_crossval.classification_ensembler.Visualizer
(csvname)
animator.show_runs(trees)
```