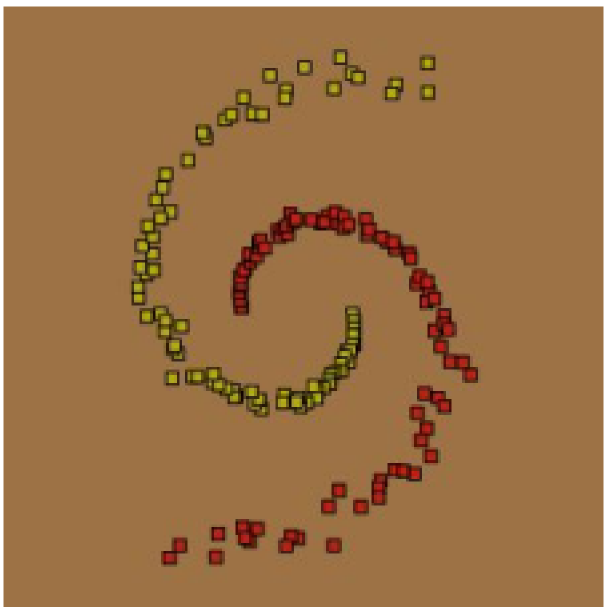
Sheet 02 : Computer Vision 2

Submitted By :
Kajal Puri, s6kapuri@uni-bonn.de, 3305398
Pallab Das, s6padass@uni-bonn.de, 3214085

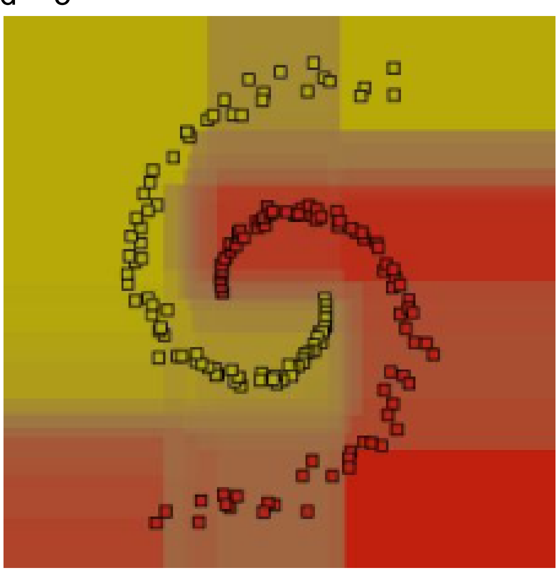
5. a) The depth of the trees.

For example_n2 file where we have 2 classes, we kept all the arguments with their default values but changed the value of /d argument to make observations.

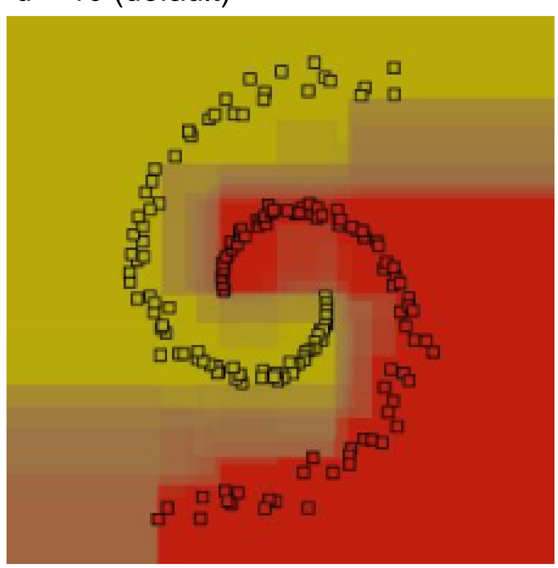
d=1



d = 5

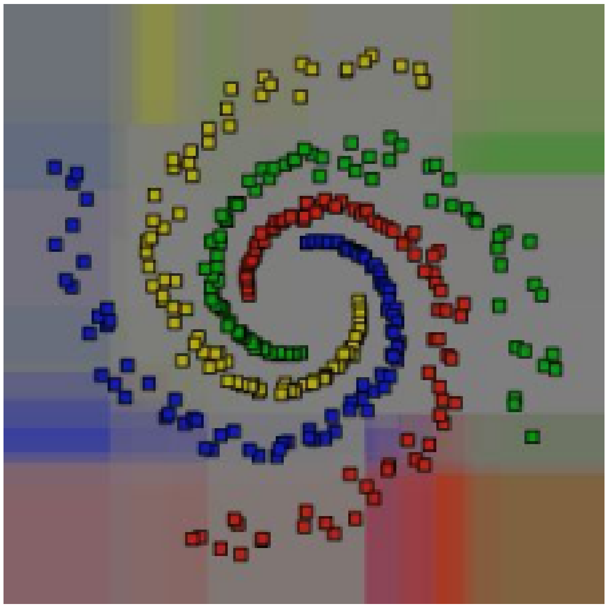


d = 10 (default)

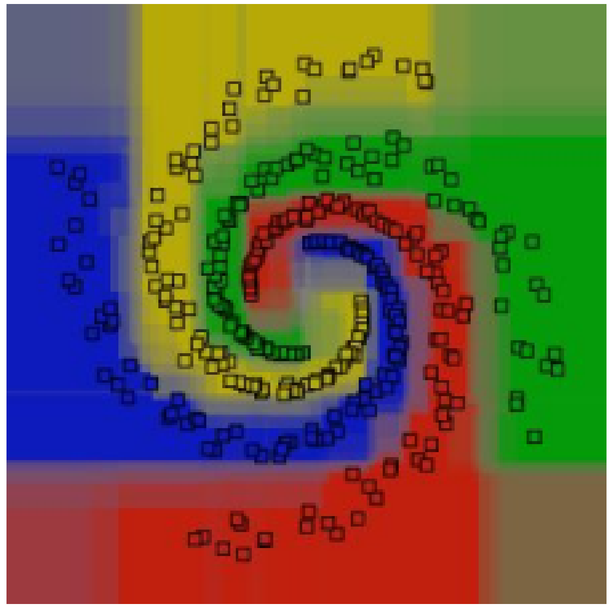


For example_n4 file where we have 4 classes, we kept all the arguments with their default values but changed the value of /d argument to make observations.

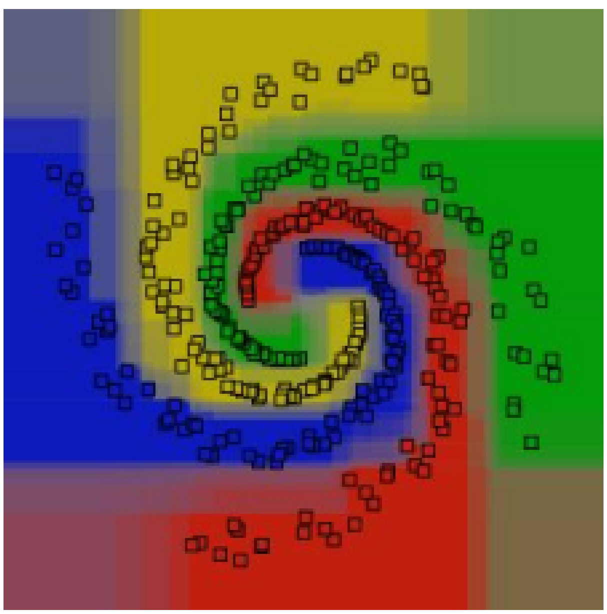
D = 5



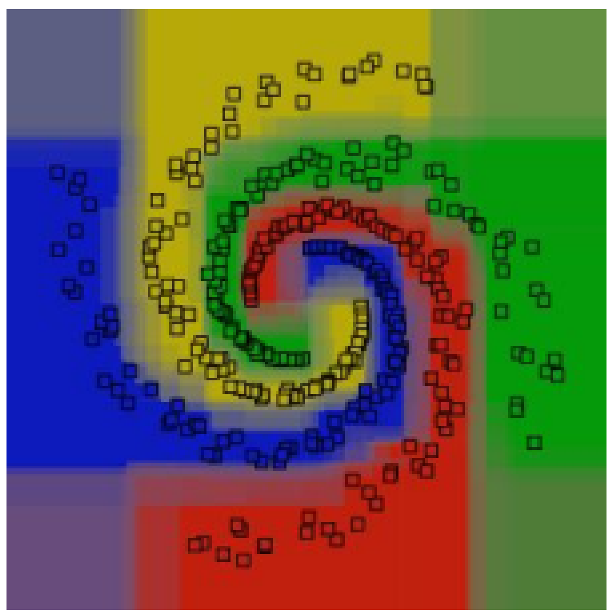
D = 10



D = 15



D = 20

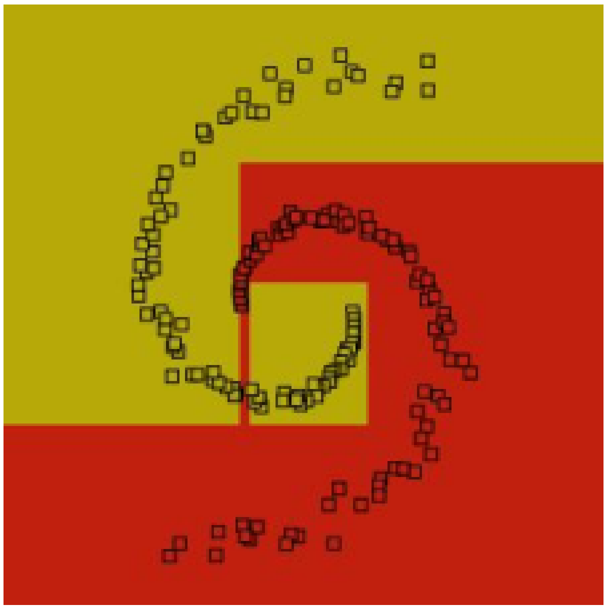


Observation : Notice that as the depth increases, we tend to get very strangely shaped classification regions. For example, at a depth of fifteen, the shape of the curves are gaining more confidence in classification. It's clear that this is not because of intrinsic data distribution but a result of the particular sampling or noise properties of the data. That is, this decision tree is clearly over-fitting our data (D = 20). Same as, very low depth value is causing underfitting.

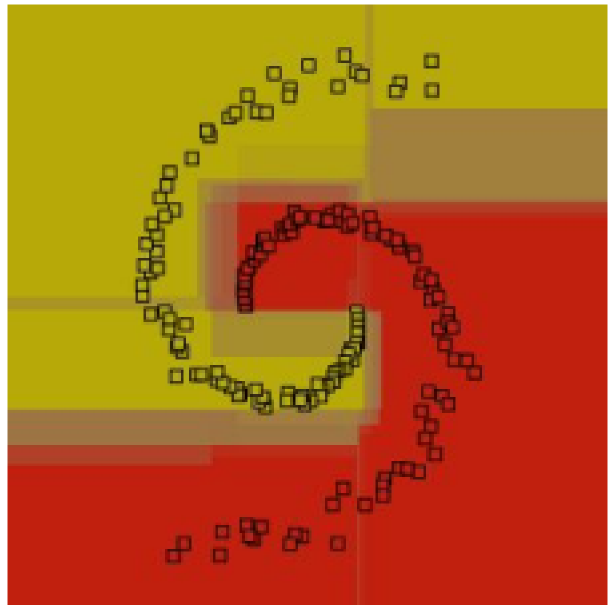
b) The number of trees in a forest.

For example_n2 file where we have 2 classes, we kept all the arguments with their default values but changed the value of /t argument to make observations.

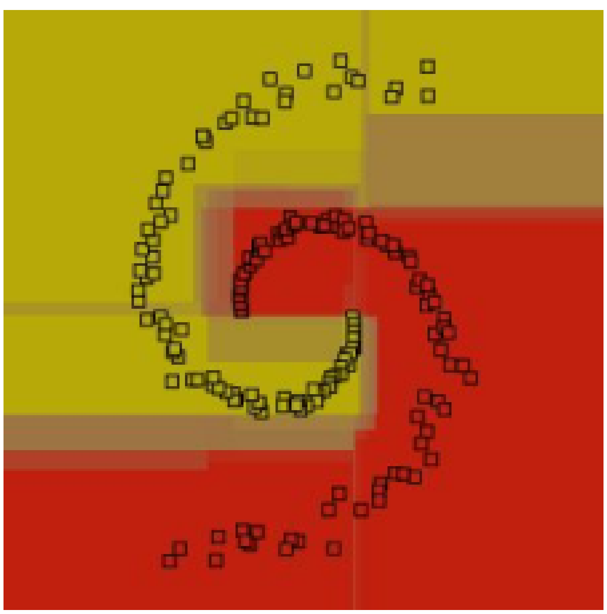
T = 1



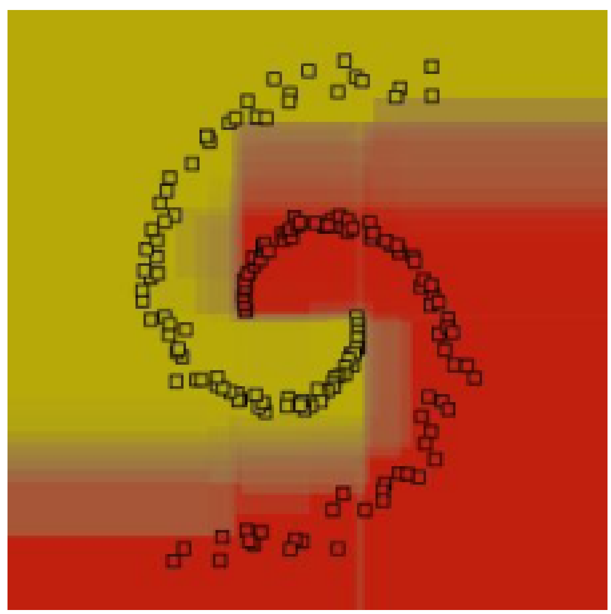
T = 2



T = 5

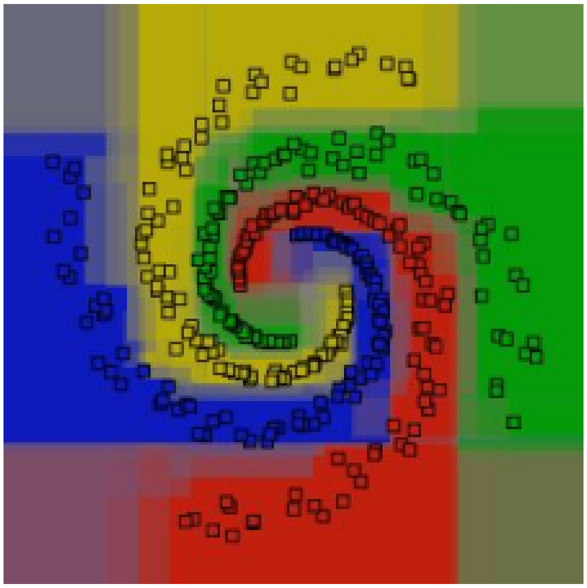


T = 15

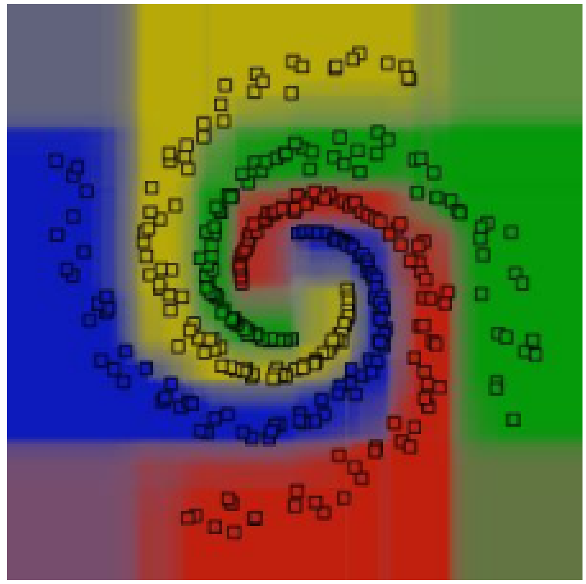


For example_n4 file where we have 2 classes, we kept all the arguments with their default values but changed the value of /t argument to make observations.

T = 1



T = 50



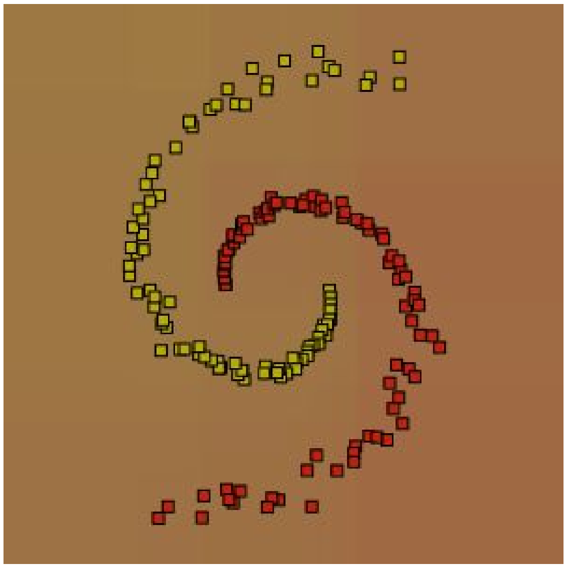
Observation :

We observed that more trees yielded better results i.e. the boundaries are much smoother. However, the improvement decreases as the number of trees increases, i.e. at a certain point the benefit in prediction performance from learning more trees was lower than the cost in computation time for learning these additional trees.

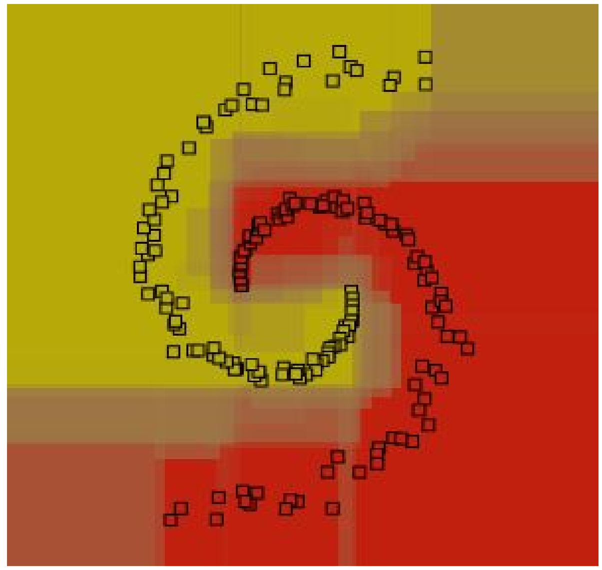
c) The number of candidate feature response functions per split node.

For example_n2 file where we have 2 classes, we kept all the arguments with their default values but changed the value of /t argument to make observations.

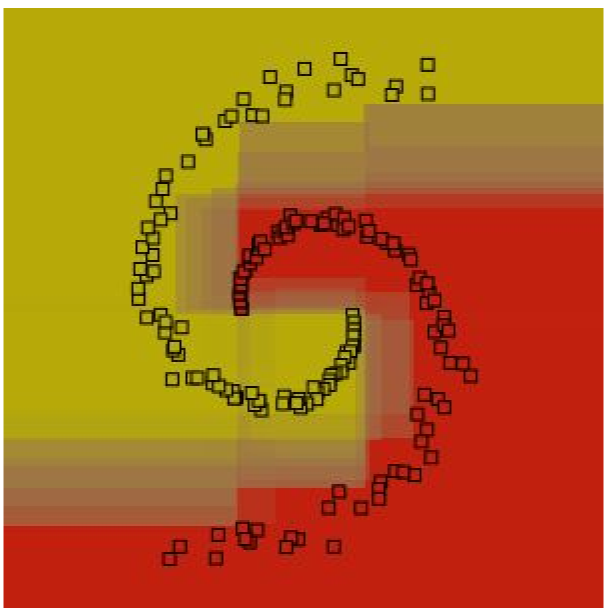
F = 1



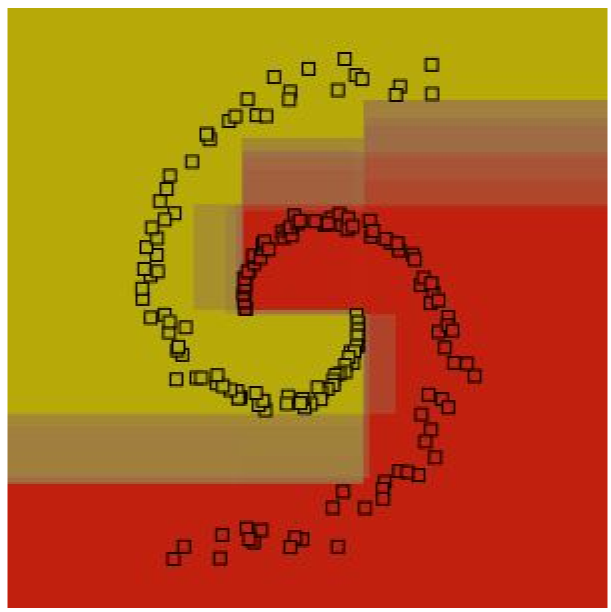
F = 5



F = 20

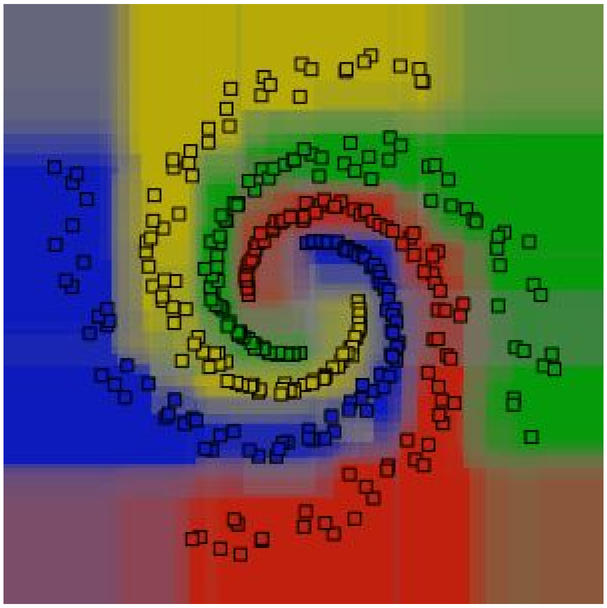


F = 100

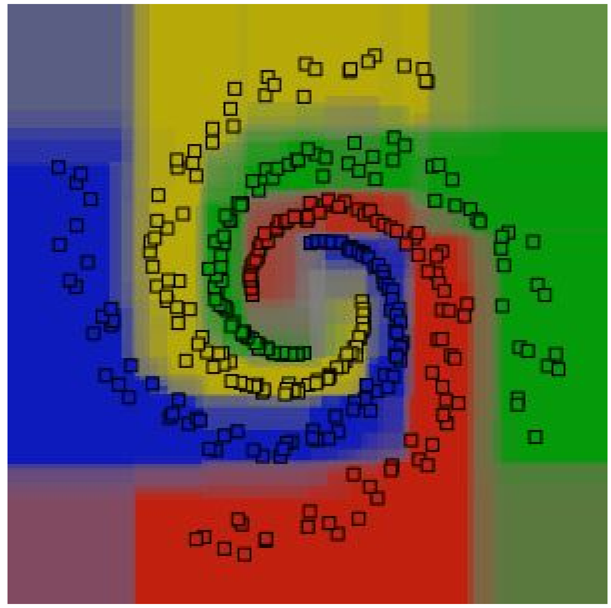


For example_n4 file where we have 4 classes, we kept all the arguments with their default values but changed the value of /t argument to make observations.

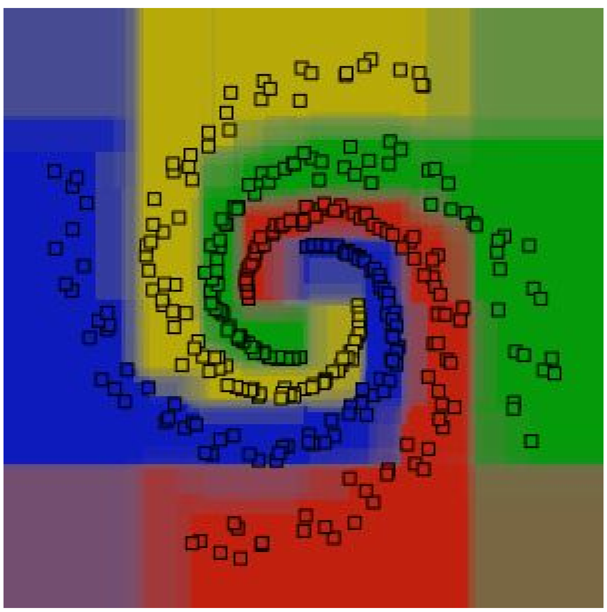
F = 5



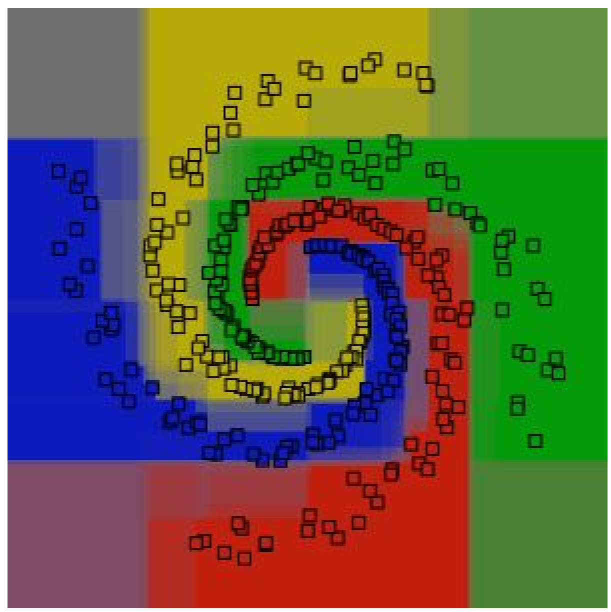
F = 10



F = 20



F = 100



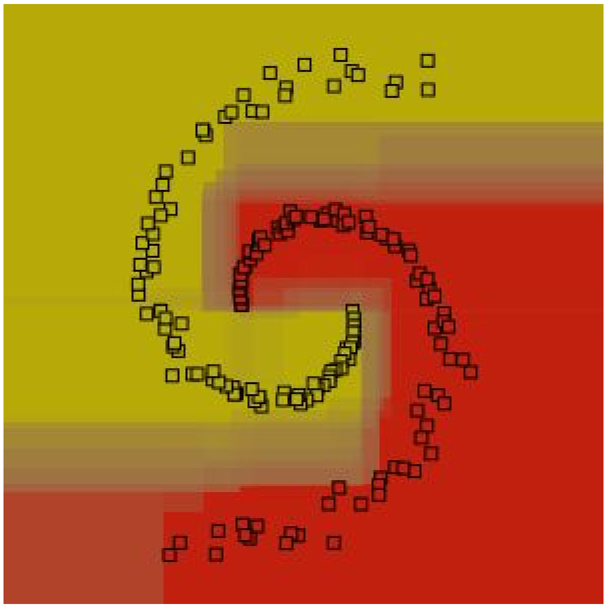
Observation :

Increasing the number of candidate features response functions per split node gives better generalization for the classes i.e. smooth boundaries but upto a certain extent, after this point it starts to overfit due to extreme model capacity.

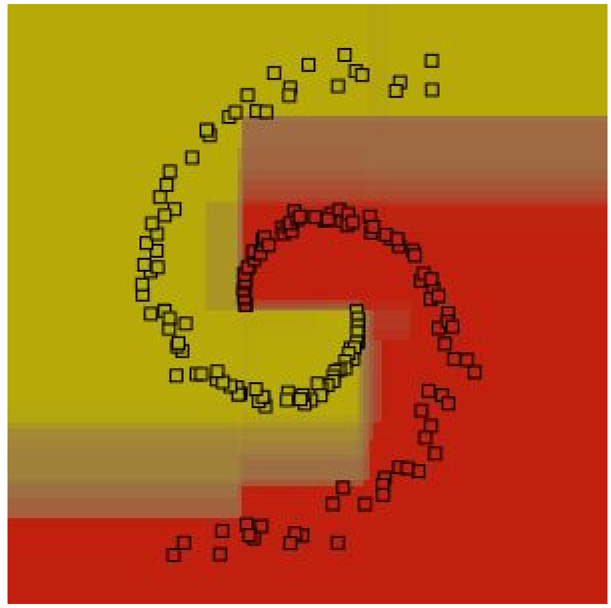
d) The number of candidate thresholds per feature response function.

For example_n2 file where we have 2 classes, we kept all the arguments with their default values but changed the value of /l argument to make observations.

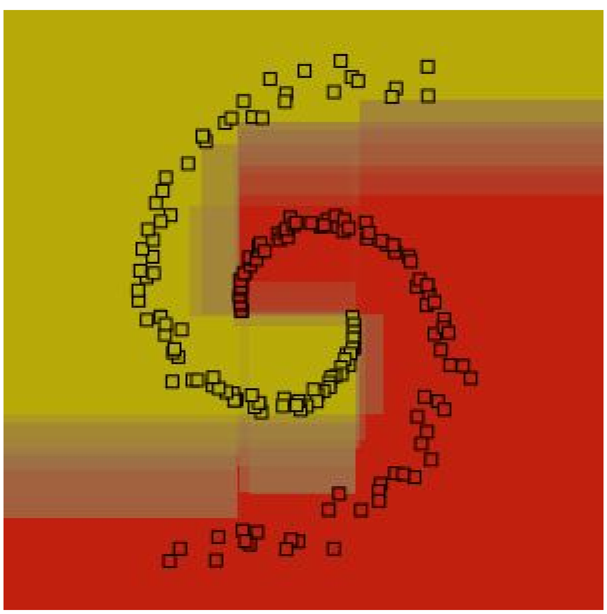
L = 1



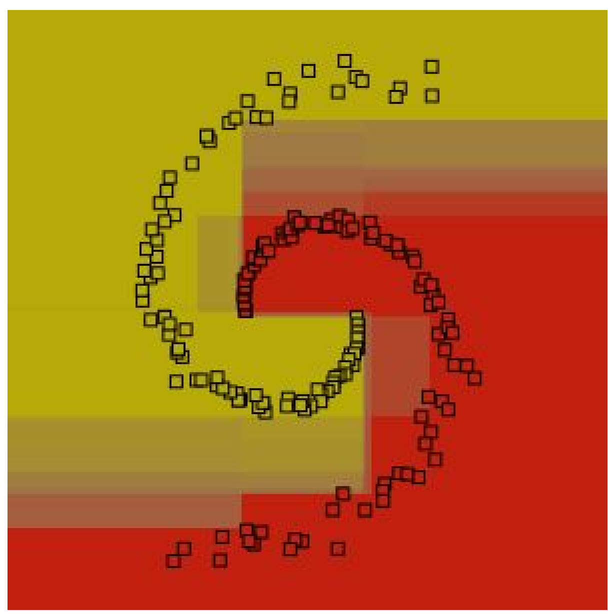
L = 5



L = 10

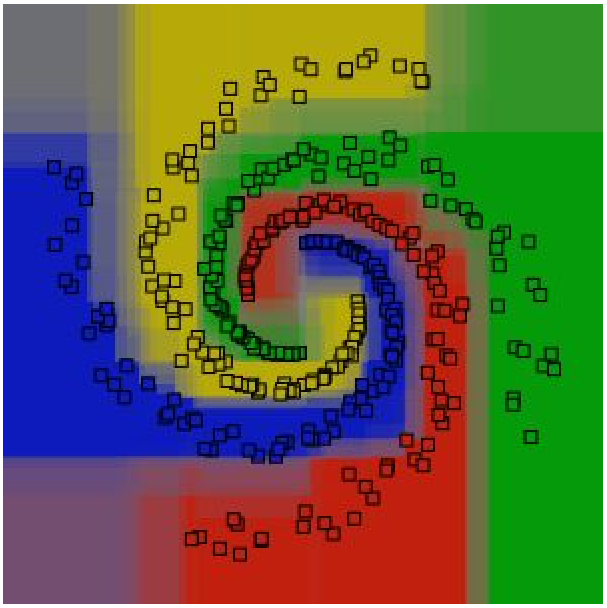


L = 20

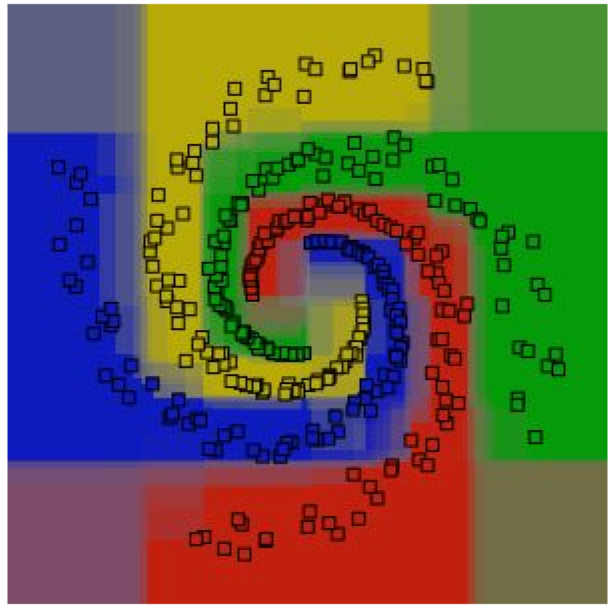


For example_n4 file where we have 4 classes, we kept all the arguments with their default values but changed the value of /l argument to make observations.

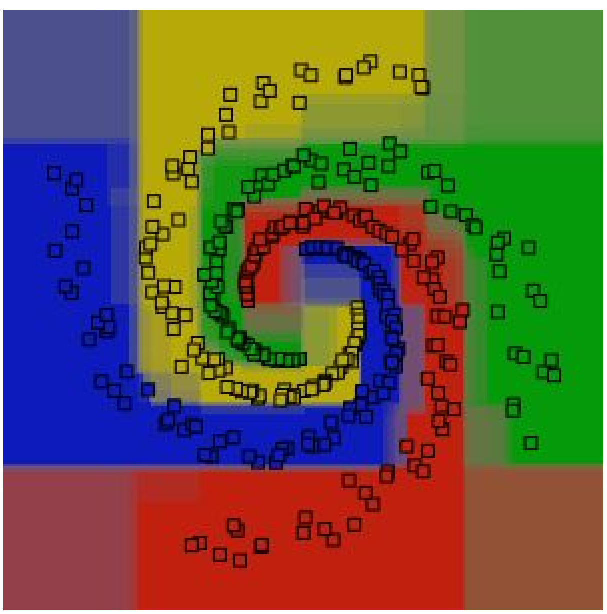
L = 1



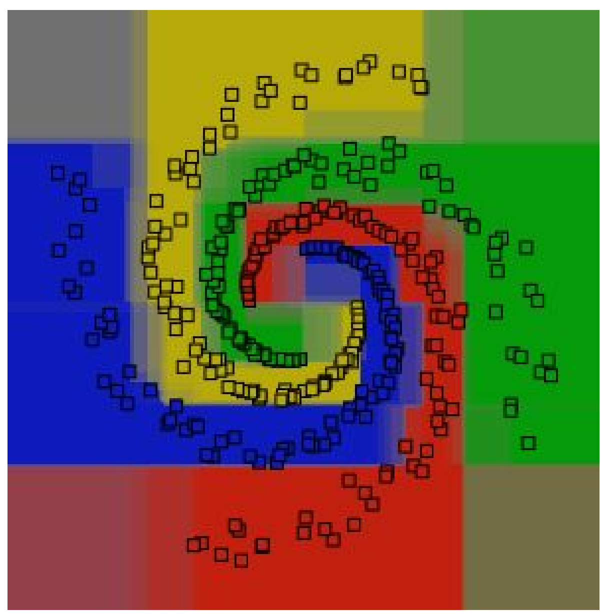
L = 5



L = 10



L = 20



Observation :

After a certain value it begins to overfit the data and the memory consumption gets higher so a small value like 1-5 seems optimal for 2-4 multiclass examples. In conclusion, the lesser value of candidate threshold per feature gives smooth boundaries and higher values produces lesser generalised results.