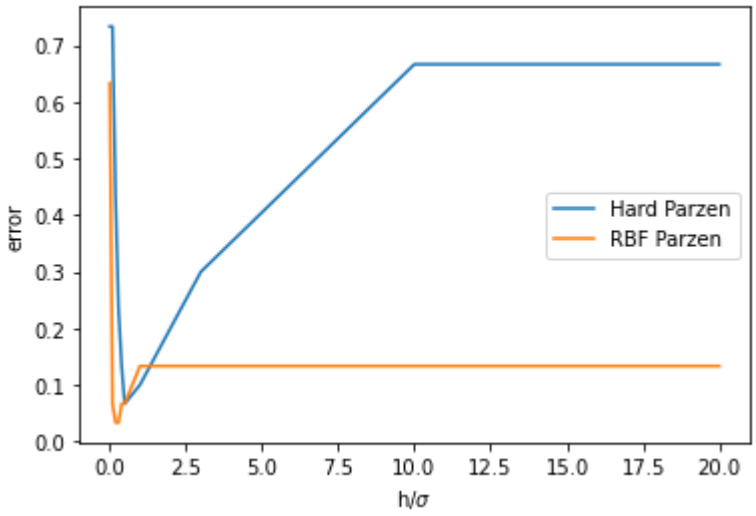
5)



For both Parzens, there is a value of h or σ at which error is minimum and the model performs best. For very small values and very large values of h or σ the error is high showing that the model performance is bad for those values. Compared to RBF Parzen, Hard Parzen seems to be more sensitive to change in h as the error is much higher for large values of h compared to large values of σ in RBF Parzen. Overall, it seems RBF Parzen performs better than Hard Parzen since it has a lower minima.

7)

i) Hard Parzen:
Training = O(1)
Because it simply loads the training data.

Evaluation:
If number of features = N
No of train examples = n_train
No of test examples = n_test

For each test example:
Distance function = O(N * n_train) is the costliest operation

Net time complexity = O(N * n_train * n_test)

ii) RBF Parzen:

Training = O(1)

Because it simply loads the training data.

Evaluation:

If number of features = N

No of train examples = n_train

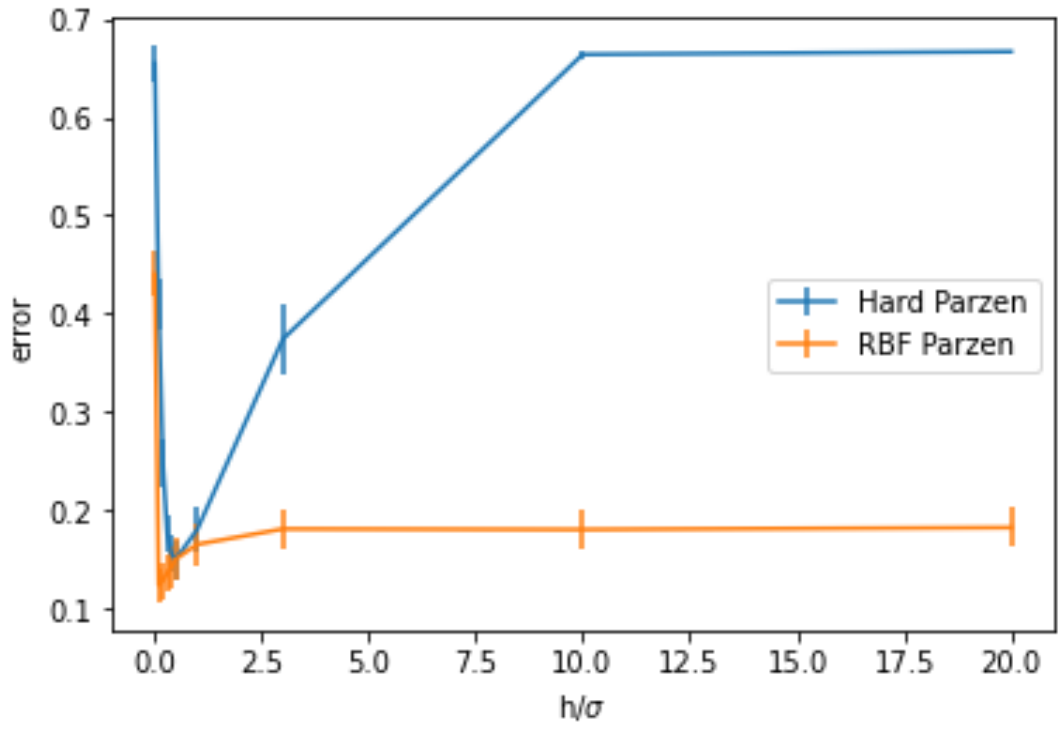
No of test examples = n_test

RBF functions has same time complexity of Distance function since it calls Distance function internally and there are no other more complex operations in RBF.

Net time complexity - O(N * n_train * n_test)

For both methods, time complexity is independent of the hyperparameters h and σ, since in both, we have to calculate kernel function value between each of the train example and each of the test example irrespective of the hyperparameters.

9)



The results are very similar to the previous experiment, showing that random projections don't affect the original distribution of data in the context of learning.