박상욱 2020710425
길이만 교수님
Machine_Learning
2020년 6월 17일

Classification of Ionoshpere Data Set

1. Input and Functional Structure

Describe the input and functional structure of the machine learning model for the ionosphere data set.

1.1. Input

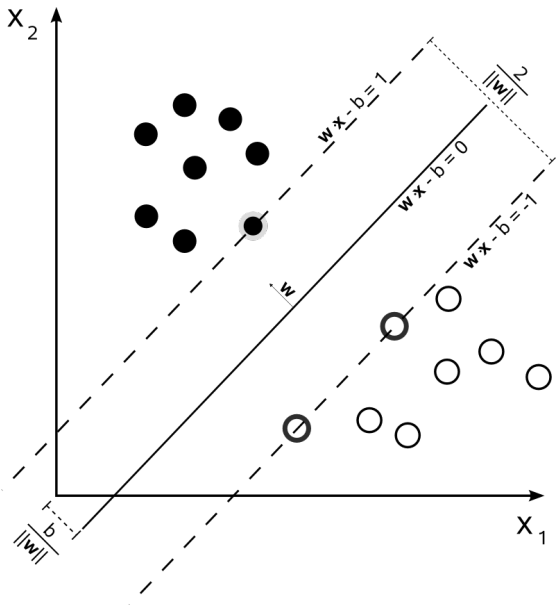
Ionosphere Datasets consists of arrays which contains antennas' data. Each row of the datafile represents 34 floating number and 1 character showing classes. There are two classes, 'g' for good and 'b' for bad. Total Number of instances is 351. Each floating number of instances is continuous feature except last elements of the instance.

In data.py file, there is a function called get_data(path): which read data from the file (if it does not exist, download it through internet request) and then divide each instances into data and label.

1.2. Functional Structure

I used 5 different machine learning models for training and evolution which consists SVM, Decision Tree, Random Forest, Ada Boost, and lastly CNN.

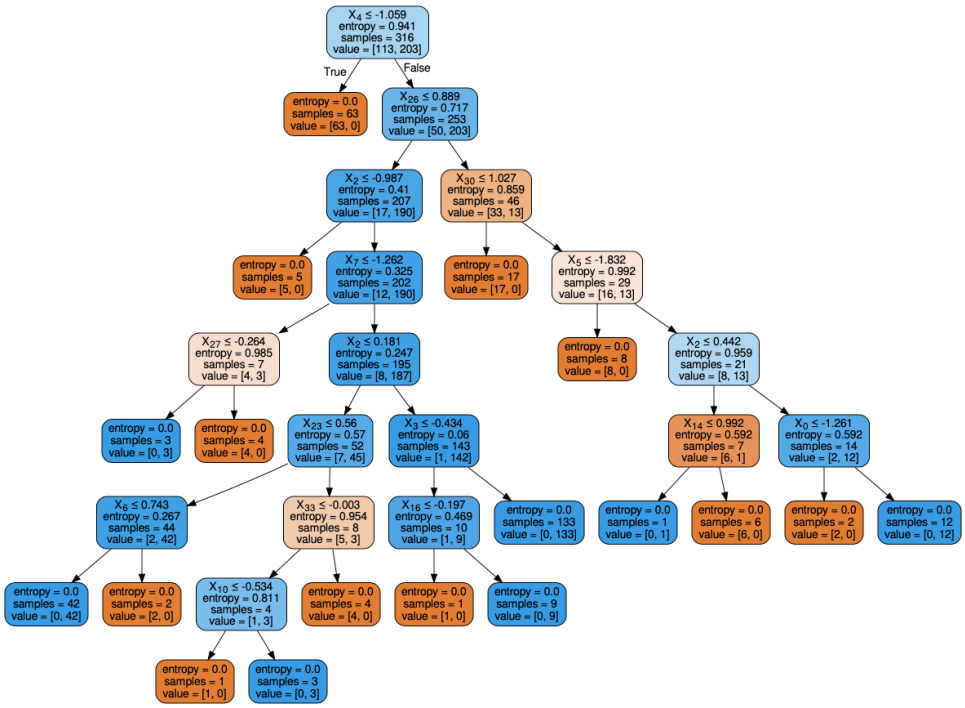
1. SVM



Support Vector Machine finds separating hyperplane which divides data into two different classes having minimum margin. Above figure shows toy example of SVM. The dotted line represents margin plane, upper one is plus-margin and lower one is minus-margin. The total margin is the distance between both plus and minus margin. Vectors which help determine margin size is called support vector. I used SVC module in skelarn.svm library. For training I used three different kernels; rbf, linear, sigmoid. Each performances are measured and will be analyzed in section 2 and 3.

2. Decision Tree

   The decision Trees are a non-parametric supervised learning method which can be used for not only classification but also regression. As task of assignment is classification, I used Decision Tree for classification task. I used 3 different max_depths when training Decision Tree. Below figure shows the actual tree structure of Decision Tree model which trained on Ionosphere model. I used DecisionTreeClassifier module of sklearn library for code implementation



3. Random Forest

   Random forest consists of multiple decision trees. In Random Forest, multiple decision trees operates as ensemble. Each trees are uncorrelated and operate as a one committee so that it will perform better than single decision tree. In this work I used different number of estimator which represents number of decision tree. I used RandomForestClassifier module of sklearn library for code implementation.

4. Ada Boost

   This model is also an ensemble-based classifier. Namely, there are multiple weak classifier to make strong classifier. This model leverages error samples of previous weak classifier when training current model to focus on hard samples. As I

diversified the number of trees when training Random forest, I also used multiple estimators, weak classifiers, for training Ada Boost model. Each estimator is Decision Tree Classifier. I used AdaBoostClassifier module of sklearn library for code implementation.

5. CNN

   I used CNN(Convolutional Neural Network) for the Deep Learning Model. This model uses convoluted feature when training. My model consists of 2 of 1-dimensional convolutional layer which use relu activation. Output logins are calculated by linear layer.

2. Experiments

2.1. Learning Parameters

Describe how and why learning parameters were determined for the selected learning model

Different learning parameters are used for training of each models. Below shows parameters.

1. SVM
   1. Kernel : rbf, linear, sigmoid
2. Decision Tree
   1. Criterion : gini, entropy
   2. max_depth : 10, 25, 50, None
3. Random Forest
   1. Number of estimator : 10, 50, 100, 300
4. Ada Boost
   1. Number of estimator : 10, 50, 100, 300
5. CNN
   1. Learning rate : 0.1, 0.01, 0.001
   2. Epoch : 50, 100, 300

2.2. Evaluation

Using the methods of evaluation of the learning models described above, the measures of evaluation for each learning model are summarized.

Every evaluation metrics are from sklearn.metrics module. The whole datasets are divided into 10 folds consisting 9 training data and 1 testing data. Evaluation proceeds at testing phase. This training and testing with 10 folds is repeated 10 times. There are 4 evaluation metrics; Accuracy, Precision, Recall, F1 score. Accuracy shows how many times a model is correctly performed. This metric looks like the best way to measure the model performances. However accuracy does not consider domain bias of the dataset. The precision is calculated by true positive over sum of true positive and false positive. This means that how many times model predicts positive outputs correctly. Another measurement is recall which is calculated by true positive over sum of true positive and false negative. Here, false negative means model predicted False but the actual value is True, meaning correctness with the respect to data. Lastly, F1 score is combination of precision and recall. Below table shows total results of each model using accuracy, precision, recall, f1 evaluation metric.

Table 1. Results

Model	No	acc	precision	recall	f1
1-1 SVM	1	0.9425	0.9336	0.9822	0.9566
	2	0.884	0.871	0.9659	0.9148
	3	0.861	0.847	0.9623	0.8995
1-2 Decision Tree	1	0.8841	0.9058	0.9169	0.9099
	2	0.8846	0.9096	0.9143	0.9105
	3	0.8831	0.9041	0.9171	0.9092
	4	0.8813	0.9059	0.9138	0.9078
	5	0.8901	0.9109	0.9227	0.915
	6	0.8887	0.9082	0.9226	0.9138
	7	0.8852	0.9056	0.9191	0.911
	8	0.8858	0.9044	0.9222	0.9117
2-1 Random Forest	1	0.9248	0.9373	0.948	0.9416
	2	0.933	0.9358	0.964	0.9487
	3	0.9333	0.9346	0.9653	0.9489
	4	0.9331	0.9344	0.9652	0.9486
2-2 Ada Boost	1	0.91	0.9135	0.9515	0.9312
	2	0.9273	0.924	0.968	0.9448
	3	0.9291	0.9273	0.9675	0.946
	4	0.9277	0.9242	0.9694	0.9453
3 CNN	1	0.9564	0.9572	0.9778	0.9666
	2	0.9271	0.9218	0.9714	0.945
	3	0.7219	0.7067	0.9867	0.8214
	4	0.9544	0.9572	0.9743	0.9648
	5	0.9468	0.9394	0.9822	0.9597
	6	0.7903	0.7644	0.9875	0.8597
	7	0.9516	0.9497	0.9778	0.9629
	8	0.9624	0.9628	0.9803	0.971
	9	0.898	0.8899	0.9639	0.9242

Orange row shows best f1 score of each model

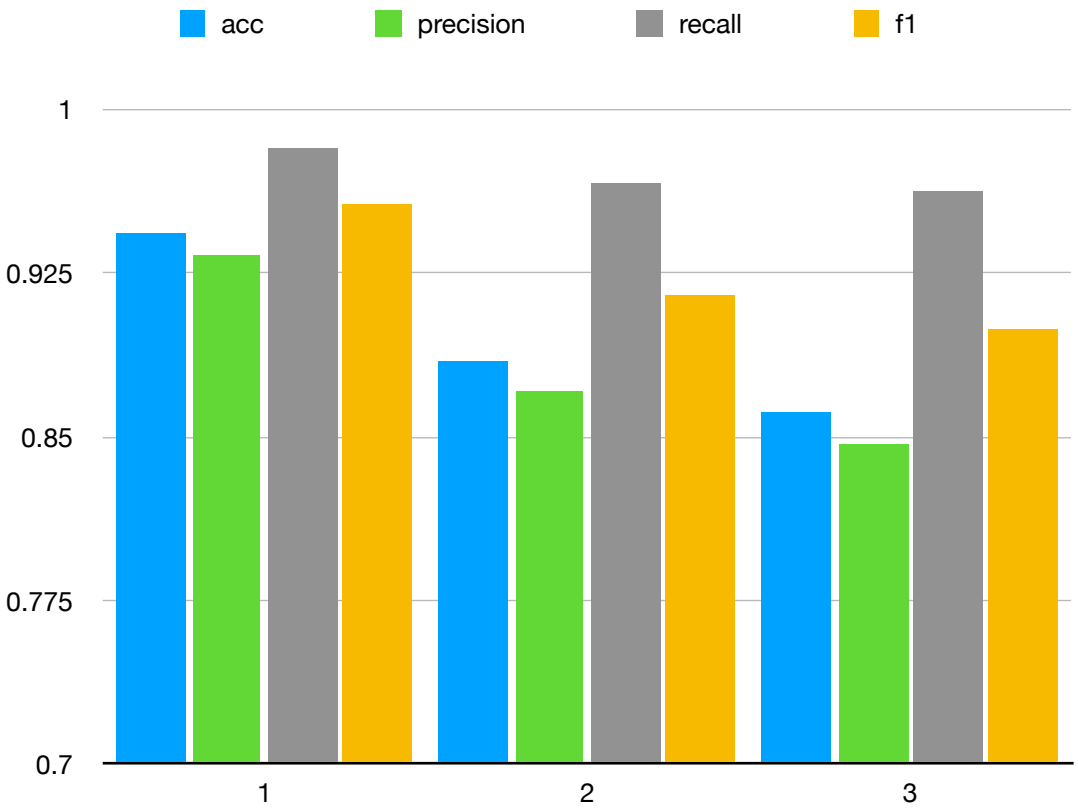
Table 2. Best Score of each models

MODEL	ACCURACY	PRECISION	RECALL	F1
SVM	0.9425	0.9336	0.9822	0.9566
Decision Tree	0.8887	0.9082	0.9226	0.9138
Random Forest	0.9333	0.9346	0.9653	0.9489
Ada Boost	0.9291	0.9273	0.9675	0.946
CNN	0.9544	0.9572	0.9743	0.9648

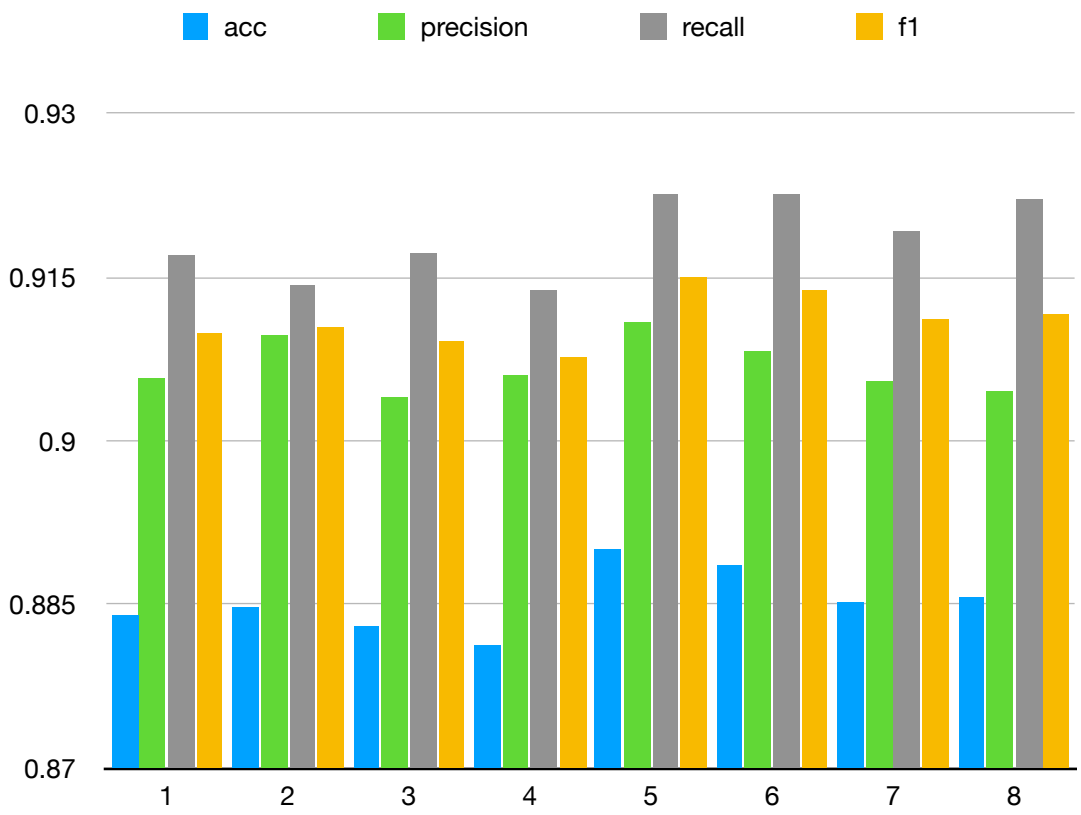
2.3. Compare models

Above table shows best f1 score of each model. It can be found that CNN shows the best performance among 5 models. As CNN is the latest deep learning method, it shows best performance with almost every evaluation metrics, except recall. The SVM model has the best recall score, which is 0.9822. Moreover even if SVM is old and simple method comparing to the others, it ranks second place. As Random Forest and Ada Boost is ensemble model using decision tree as base estimators, shows better performance than decision tree. Between Random Forest and Ada Boost, Random Forest has better results.

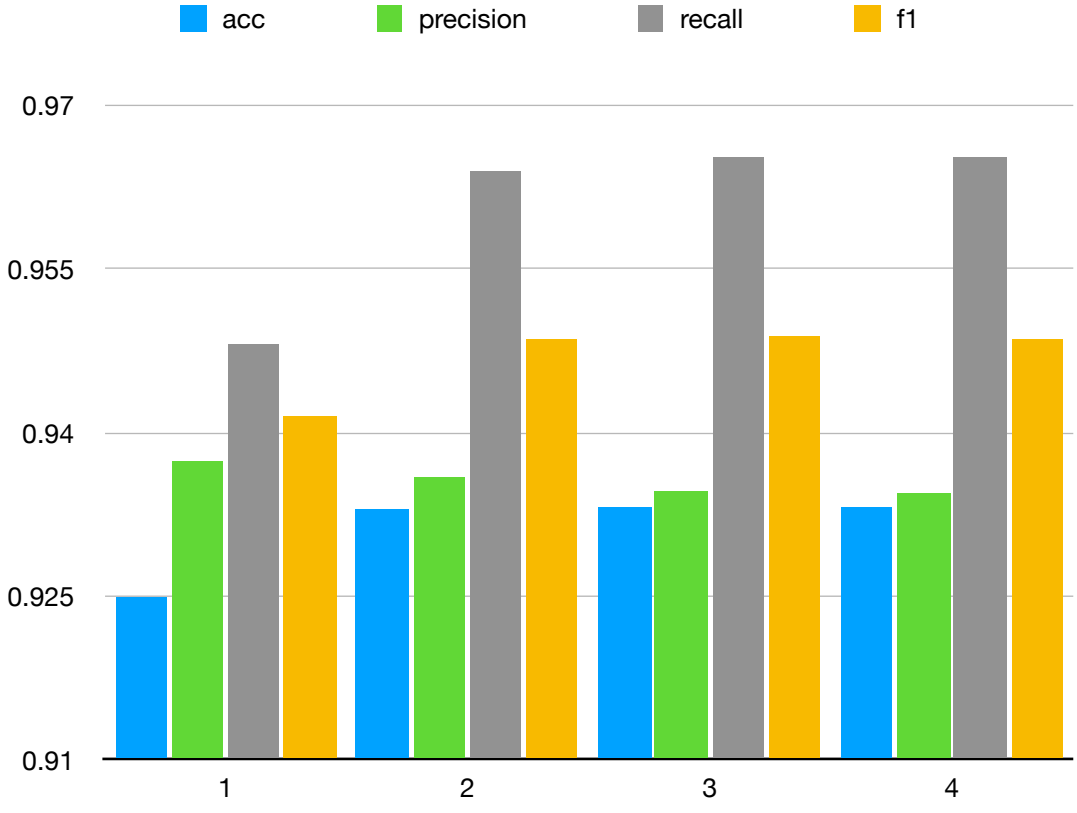
The learning parameter comparison can be found through below figures.



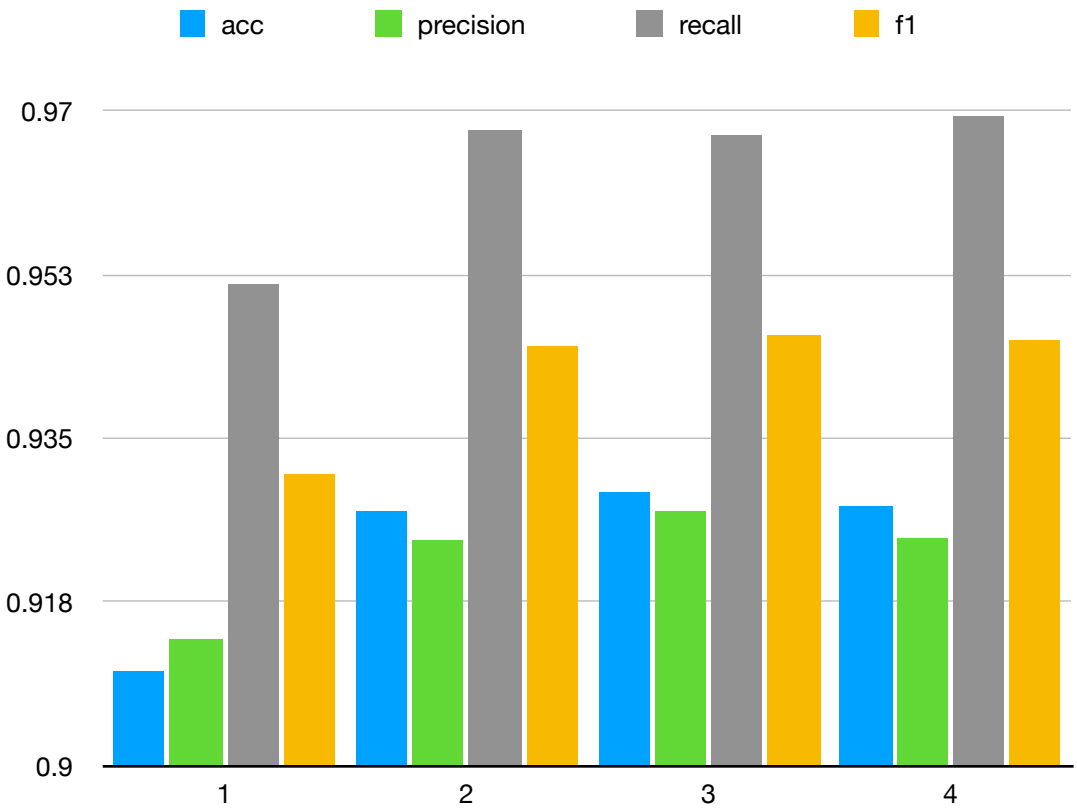
SVM results using different kernels. 1 uses rbf, 2 uses linear, 3 uses sigmoid. Using rbf kernel shows best performance on SVM.



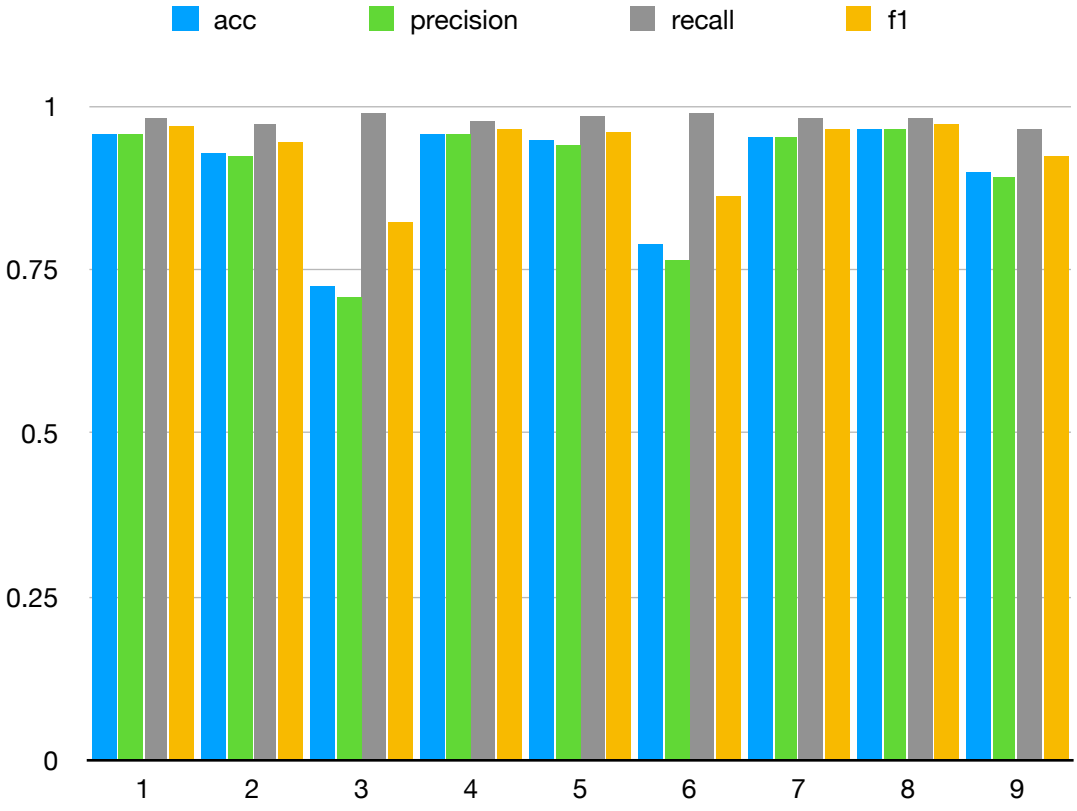
Decision Tree using different criterion and max depth of tree. 1 uses gini index and 10 depth. 2 uses gini index and 25 depths. 3 uses gini index and 50 depth. 4 uses gini index and doesn't have max depth. 5 uses entropy criterion and 10 depth. 6 uses entropy and 25 depths. 7 uses entropy and 50 depths. Lastly 8 uses entropy and no max depth.
Entropy with 10 max depths shows best performance with every evaluation metric. Apparently entropy calculation is the best criterion for the decision tree, and proper depth is 10 because every model which have 10 depth shows best performance.



This figure shows evaluation of different number of estimators of Random Forest. 10, 50, 100, 300 estimators are used for the evaluation. There are large gab between 10 and 50 estimators. However the increase tend to get low after 50 estimators. Consequently, there is no need to use too many estimators when using Random Forest



Similar to Random Forest, Ada Boost uses 4 different number of estimators. 10, 50, 100, 300 for each model. It shows same phenomena like Random Forest. There is no performance increase after 50 estimators.



Lastly, for CNN I diversified learning rate and training epoch. Learning rates are 0.1, 0.01, 0.001, and epochs are 50, 100, 300. Proper learning rate. CNN model with 0.1 learning rate and 50 epoch has shown best score. 0.001 learning rate shows lowest performance between every epoch diversion. 300 epoch shows Lowest performance. It could be interpreted as there are over fitting problem with epoch which is over 100.

3. References

SVM
https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

Decision Tree
https://scikit-learn.org/stable/modules/tree.html

RandomForest
https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

Adaboost
https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html