Supervised Learning (Assignment 1)

Qiyu Wu

[Qwu346@gatech.edu](mailto:Qwu346@gatech.edu)

**Abstract**

This is an analysis which use five supervised learning algorithms (decision trees, neural networks, boosting, support vector machines) to train and predict on two different data sets are used.

**Introduction**

For each of the algorithm, it starts with the default classifier and then tune at least two hyperparameters as trying to improve the accuracy. Sklearn library is heavily used to avoid implementing algorithms so the work can focus on training the data, observe the

Difference and explore these five algorithms.

**Dataset**

Two different data sets are chosen for this project. The first data sets is the Titanic – Machine Learning from Disaster. The goal is to predict a passenger can survive or not based on the 11 features (columns). It contains 890 rows and 12 features. The second data set is Iris Flower Dataset. The target using this dataset is to predict the types of a flower based on its feature.

One major reason that I chose these two datasets is that their targets are classification with a few values, which suits the nature of all five algorithms thus would be inclined to get a good performance.

**Procedure**

Data preprocessing is necessary before trying to fit a model. After reading file into kernel, some irrelevant columns would be drop. Some are obvious by the name, for example, “Name” and “ID”, some are dropped after considering correlation etc.

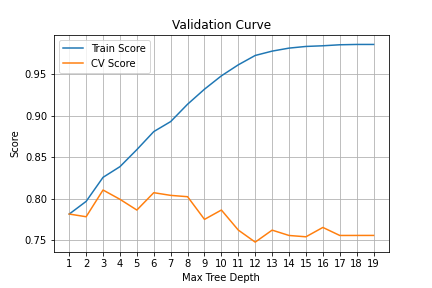
Both of the datasets will be split into training set and test set of a ratio of 70 : 30. Training set will be used to train the five models, while the test set will be used to test the performance of the models after tuning hyperparameters. The experiment started with models with hyperparameters with default values. At least two hyperparameters would be tuned separately for each model, and a grid search would be used to find the optimal hyperparameters using the validation curves got from previous tuning. Learning curve is plotted in order to see how the optimal model perform against different sizes of sample.

**Five Supervised Learning Models**

**Decision Trees**

The two hyperparameters chosen for decision trees are “max\_depth” (max depth of the decision tree) and “ccp\_alpha” (complexity parameter used for pruning).

The Titanic Dataset

Chart, line chart

Description automatically generated

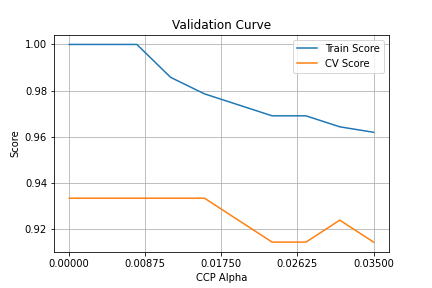
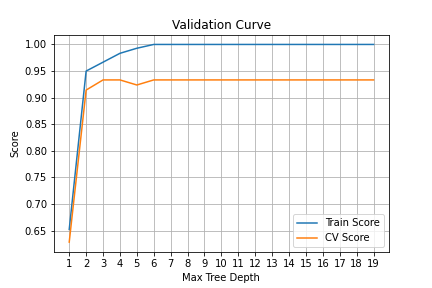
Chart, line chart

Description automatically generated

Through grid search, performance is best when “max\_depth” is 4 and “ccp\_alpha” is 0.0078, which matches what can be observed in the two validation curve plots above. This decision tree reached 0.84 in accuracy in test set.

Looking at the learning curve, we can say that the train score and cross-validation score converge when proceeding with more and more data. This is a good fit.

The Iris Dataset



Chart, line chart

Description automatically generated

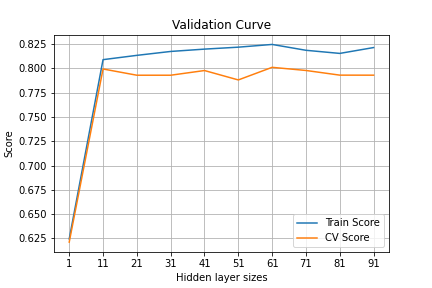
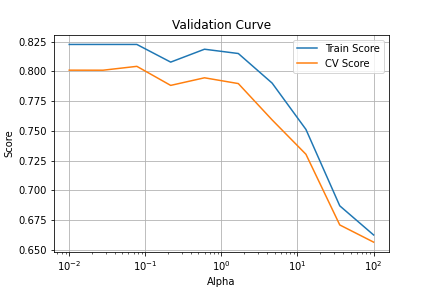
Through grid search, performance is best when “max\_depth” is 13 and “ccp\_alpha” is 0, which matches what can be observed in the two validation curve plots above. This decision tree reached 1 in accuracy in test set.

Looking at the learning curve, the train score is 1 as the iris data set is well trained for a lot of classification algorithm implemented in sklearn and it is used in the example code for decision trees. This is a good fit as a convergence is shown.

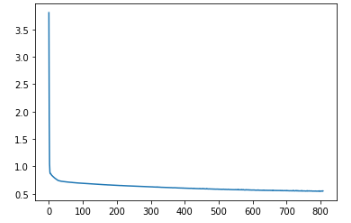
**Neural Networks**

The two hyperparameters chosen for neural networks are “alpha” (strength of the L2 regularization term) and “hidden\_layer\_sizes” which represent the number of neurons in the hidden layer that do the calculation. “sgd” is set for solver as gradient descent is used to optimize.

The Titanic Dataset



Chart, line chart

Description automatically generated

Through grid search, performance is best when “alpha” is 0.5994842503189409

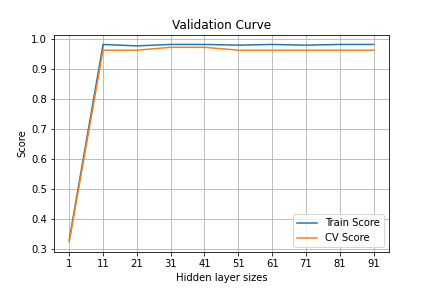
and “hidden\_layer\_sizes” is 91, which is not obvious from the two above validation curve plots and thus shows the necessity of grid search. This neural network reached 0.81 in accuracy in test set.

According to this learning curve, the reason why the train score started at a high score as 0.9 is likely a overfitting due to the limited size, thus when size increased the score tended to be stable and stabilized at around 0.81. The sudden drop means that the sample from 10% to 20% might contain a lot of noise which lead to poor score. Thus this is a good fit overall.

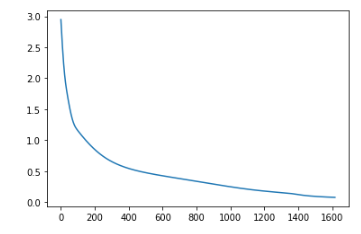
Looking at the loss curve, can see that the loss was lower, the precision got higher when analyzing more data.

The Iris Dataset

Chart, line chart

Description automatically generated

Chart, line chart

Description automatically generated

Through grid search, performance is best when “alpha” is 0.01

and “hidden\_layer\_sizes” is 11, which is not obvious from the two above validation curve plots and thus shows the necessity of grid search. This neural network reached 1.0 in accuracy in test set.

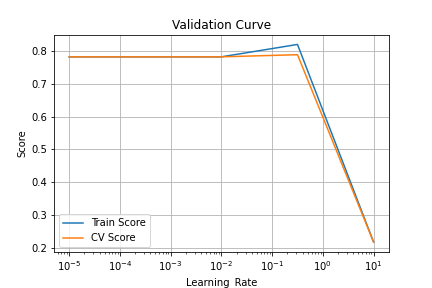
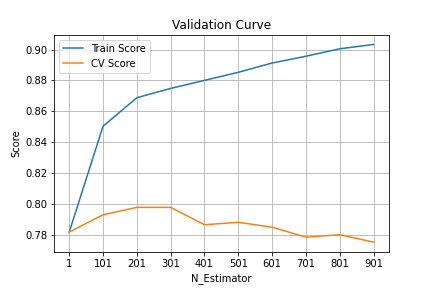
According to this learning curve, train score started at a high score and quickly stabilize. As for the cross validation score, it started low but quickly reached a high score. Since it converge very quickly and didn’t diverge, this is a good fit.

Looking at the loss curve, can see that the loss was lower, the precision got higher when analyzing more data.

**Boosting**

The two hyperparameters chosen for boosting are “n\_estimators” (the number of weak learners that later will be combined to conduct the computation) and “learning\_rate” which will shrink the weights of each sub classifier. The classifier used is AdaBoostClassifier.

The Titanic Dataset



Chart, line chart

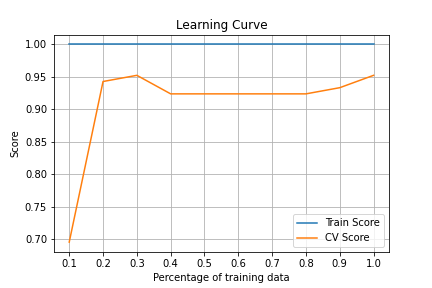
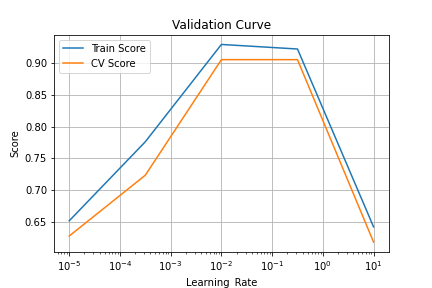
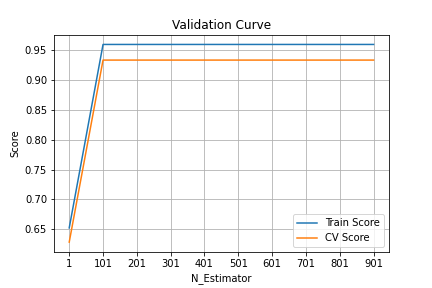
Description automatically generated

Through grid search, performance is best when “n\_estimators” is 101

and “learning\_rate” is 0.32 , which matches what can be observed in the two validation curve plots above. This decision tree reached 0.81 in accuracy in test set.

According to this learning curve, train score and cross validation score converge overall, and this represented this is a good fit. Since the there was no sign of closing the gap since 0.8, it means that more data would not help with this model after this point.

The Iris Dataset



Through grid search, performance is best when “n\_estimators” is 601

and “learning\_rate” is 0.00031622776601683794

, which matches what can be observed in the two validation curve plots above. This decision tree reached 1 in accuracy in test set.

According to this learning curve, the train score stabilized at one from the beginning to the end, same as using decision trees. And the cv scores rose to converge with the train score.

**Support Vector Machines**

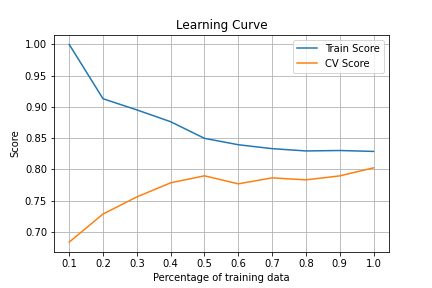
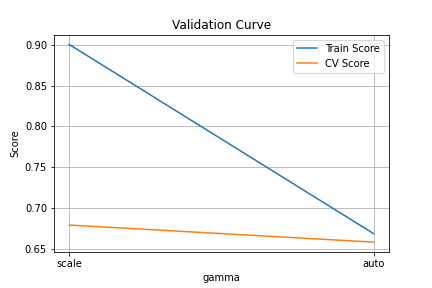
Unlike other models, there are three hyperparameters. The most crucial hyperparameter is “kernel” as if the transformation is incorrect, the model can have very poor performance. The other two that chosen are “C” and “gamma”, these two are correlated thus need to be tuned together.

The Titanic Dataset

Chart, line chart

Description automatically generatedChart, line chart

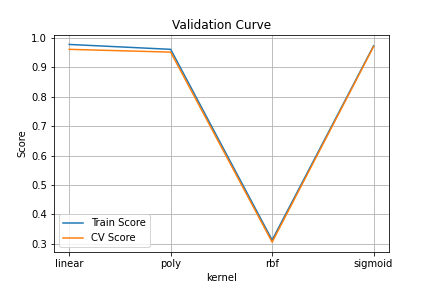
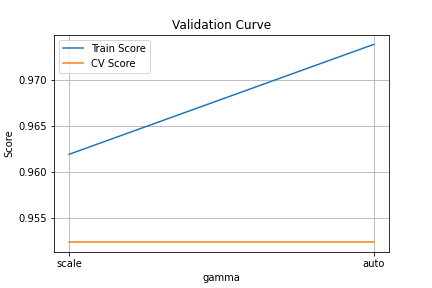
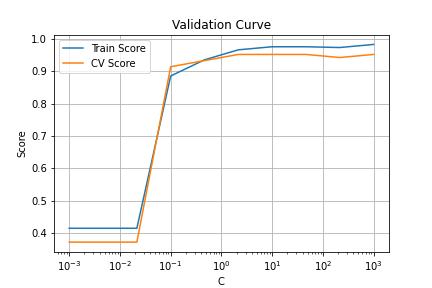
Description automatically generated



From the three validation curves above, it can observed there are no trend for “kernel” and “gamma” as the range of them are distinct values respectively. As for C, it showed an upward trend.

According to this learning curve, the train score and the cv scores rose to converge with the train score. It can be concluded that SVM is a good fit for the Titanic Dataset.

The Iris Dataset

 Chart, line chart

Description automatically generated

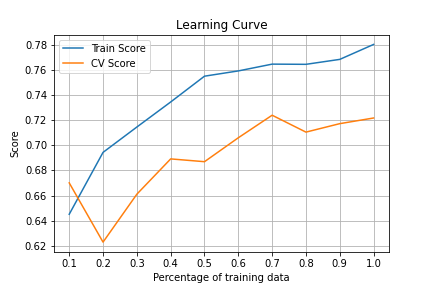
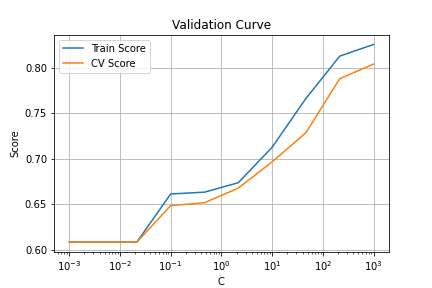
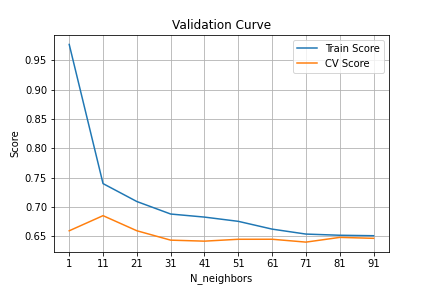
Compared to the validation curves for the Titanic datasets. It can be seen that there are no pattern for the hyperparameters which values are like enum, the choice based on the dataset.

In the plot of this learning curve, the train score and cv score converge at around 60% and then started drifting away, it showed a sign of overfitting that the model trained by the training set caught some noise which would not help with the prediction in the testing set.

**Nearest Neighbors**

The two hyperparameters chosen for boosting are “n\_neighbors” (the number of neighbors chosen that will be used to help predict the classification) and “p” (power parameter).

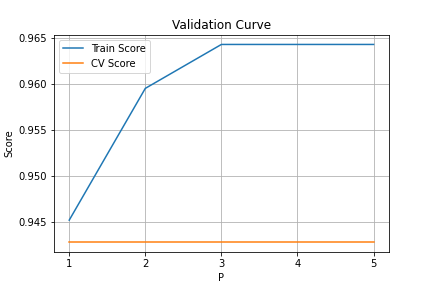
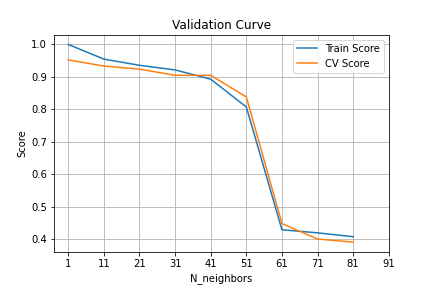
Titanic Dataset

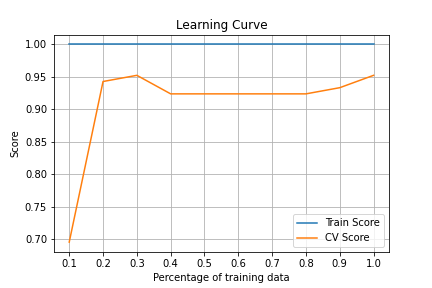


Through grid search, performance is best when “n\_neighbors” is 11 and “p” is 1, which matches what can be observed in the two validation curve plots above. This decision tree reached 0.75.

It can be seen from the learning curve that the train score and cross validation score didn’t converge, the gap between the two maintained the same regardless the percentage of training data, so this is a underfit.

Iris Dataset





Through grid search, performance is best when “n\_neighbors” is 1 and “p” is 2, which matches what can be observed in the two validation curve plots above. This decision tree reached 1 in accuracy in test set as all the previous models.

According to this learning curve, the train score stabilized at one from the beginning to the end, same as using decision trees and boosting. And the cv scores rose to converge with the train score. It can be concluded that K-NN is a good fit for the Iris Dataset.

**Wall Clock**

Time\_to\_train means time taken to fit the training sets after tuning hyperparameters. Time\_to\_predict means time taken to predict the target in the testing set. The two tables below show the time for the Titanic dataset and Iris dataset respectively.

|  |  |  |
| --- | --- | --- |
| Titanic Dataset | Time\_to\_train (s) | Time\_to\_predict (s) |
| Decision Tree | 5.221476316452026 | 0.0020127296447753906 |
| Neural Networks | 468.6052122116089 | 0.003567934036254883 |
| Boosting | 195.35589289665222 | 0.02073931694030761 |
| Support Vector Machines | 69.66563510894775 | 0.004541635513305664 |
| K-nearest neighbors | 3.1166388988494873 | 0.011260271072387695 |

|  |  |  |
| --- | --- | --- |
| Iris Dataset | Time\_to\_train (s) | Time\_to\_predict (s) |
| Decision Tree | 3.9491939544677734 | 0.0023555755615234375 |
| Neural Networks | 232.6334846019745 | 0.0026993751525878906 |
| Boosting | 167.96679973602295 | 0.10438156127929688 |
| Support Vector Machines | 2.591108560562134 | 0.002271890640258789 |
| K-nearest neighbors | 1.2812893390655518 | 0.004324436187744141 |

It can be seen that neural networks model had the highest time\_to\_train in both of the two datasets as it uses the back propagation technique to generalize the pattern, however, its time\_to\_predict is relatively good among all five models. Decision tree and k-nearest neighbours have low time\_to\_train because the mechanism behind are relatively easy and require less steps.

When comparing the time regarding the two datasets, the time taken for the Titanic dataset was longer as it has more data to fit and predict.

**Conclusion**

Overall, the accuracy takes a big role in choosing a model. It can be said that Support Vector Machines performed the best for both of the datasets as it took little time to train while taking in more data, and the time complexity while predicting for new data is also good. Loss curve analysis is necessary when doing neural network analysis as gradient descent is used behind the scene and loss is calcaluated every step in the bgack propagation.