CS 7641: Machine Learning
Assignment 1: Supervised Learning

Author: Meghana Bhimasani
mbhimasani3@gatech.edu

Abstract

For this assignment, I trained, tested, and analyzed the performance of 5 supervised learning classifiers (Decision Trees, Neural Networks, Gradient Boosting, Support Vector Machines, k-Nearest Neighbors) on 2 classification problems.

Datasets:

Dataset 1: Online Shoppers Purchasing Intention

This dataset was found via the UCI Machine Learning Repository. The dataset consists of feature vectors belonging to 12,330 sessions. The dataset was formed so that each session would belong to a different user in a 1-year period to avoid any tendency to a specific campaign, special day, user profile, or period. The dataset consists of 10 numerical and 8 categorical attributes, with the 'Revenue' field acting as a class (target) label. 1,892 instances were positive class samples ending with shopping (i.e. Revenue equaled True), and 10,353 instances were negative class samples that did not end with shopping. Additionally, features are somewhat correlated. "Administrative", "Administrative Duration", "Informational", "Informational Duration", "Product Related" and "Product Related Duration" represent the number of different types of pages visited by the visitor in that session and total time spent in each of these page categories. The values of these features are derived from the URL information of the pages visited by the user and updated in real time when a user takes an action, e.g. moving from one page to another. The "Bounce Rate", "Exit Rate" and "Page Value" features represent the metrics measured by "Google Analytics" for each page in the e-commerce site. The value of "Bounce Rate" feature for a web page refers to the percentage of visitors who enter the site from that page and then leave ("bounce") without triggering any other requests to the analytics server during that session. The value of "Exit Rate" feature for a specific web page is calculated as for all pageviews to the page, the percentage that were the last in the session. I felt that this dataset was interesting specifically because of the number of correlated features and the imbalanced data.

Dataset 2: Synthetic Data

This dataset was synthetically generated using scikit's make_classification method. The dataset consisted of 3000 instances with 10 features and 1 class label. The data was evenly split with 1496 positive class samples and 5004 negative class samples. Features are not correlated.

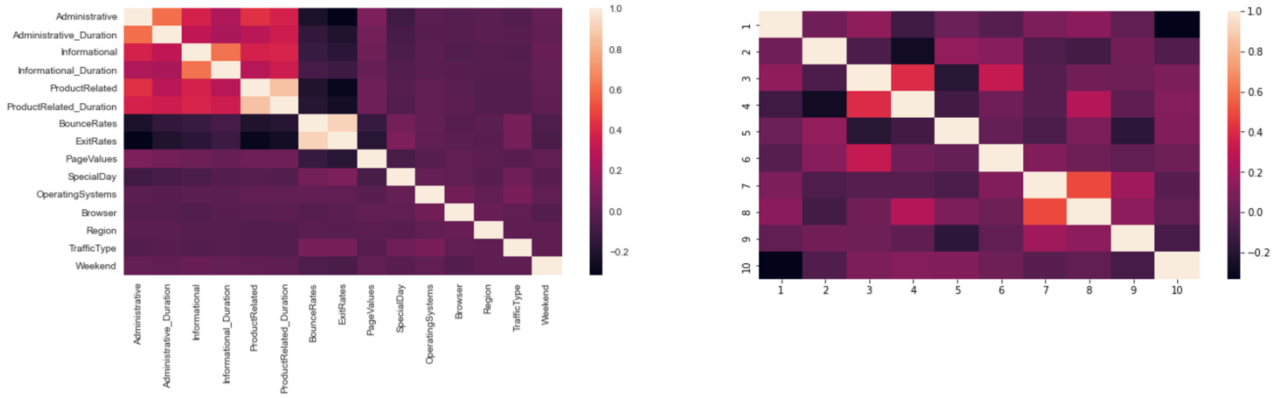

Figure 1. Heatmaps for dataset 1 and dataset 2 displaying lack of feature correlation

Classifiers:

For all classifiers, I set a random state of 142, used 5 folds when performing cross validation, and set training sizes as a range of values from 0.2 to 1.0 with a 0.1 step when creating learning curves.

1. Decision Tree (DT)

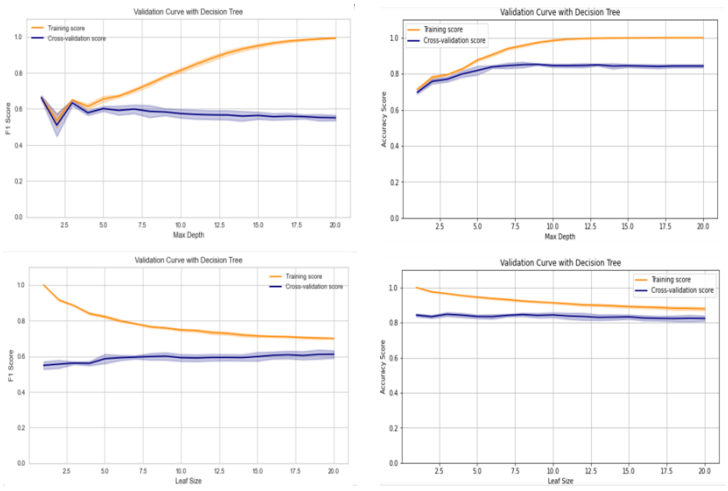

Figure 3. Cross Validation curves for dataset 1 (left column) and dataset 2 (right column)

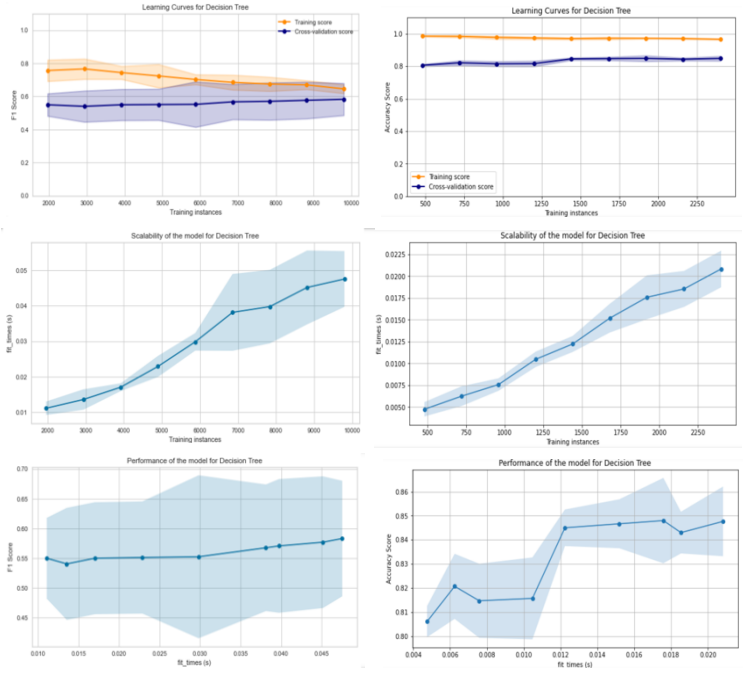

Figure 3. Learning rate curves, scalability, and performance curves for dataset 1 (left column) and dataset 2 (right column)

For the decision tree classifier, I tuned the hyperparameters "max depth" and "leaf size". The range of values tested for the "max depth" and "leaf size" hyperparameters for both datasets were 1 through 20 with a step size of 1.

In the cross validation curves for dataset 1, we can see that the DT classifier shows the ideal range of max depth at under 25 and the ideal range of leaf size at above 15. Overfitting is seen when max depth is above 25 or while leaf size below is 15. Through applying the GridSearchCV method from scikit learn to test optimal combinations of max depth and leaf size, it was determined that the optimal value of max depth was 5, while the optimal value of leaf size was 17. Additionally, the learning rate curve shows that both the validation score and the training score converge to a value that is *quite low* with increasing size of the training set. Thus, we will probably *not* benefit much from more training data. Finally, the Performance of the Decision Tree classifier was 0.893 with the optimized values for the max depth and leaf size hyperparameters.

In the cross validation curves for dataset 2, we can see that the DT classifier shows the ideal range of max depth at under 5 and the ideal range of leaf size at above 12. Overfitting is seen when max depth is above 5 or while leaf size below is 12. Through applying the GridSearchCV method from scikit learn to test optimal combinations of max depth and leaf size, it was determined that the optimal value of max depth was 9, while the optimal value of leaf size was 1. Additionally, in contrast with the learning rate curve for dataset 1, the learning rate curve for dataset 2 shows that both the validation score and the training score *only slightly* converge to a value with increasing size of the training set. Thus, we will probably benefit greatly from more training data. Finally, the Performance of the Decision Tree classifier was 0.852 with the optimized values for the max depth and leaf size hyperparameters.

2. Neural Networks (NN)

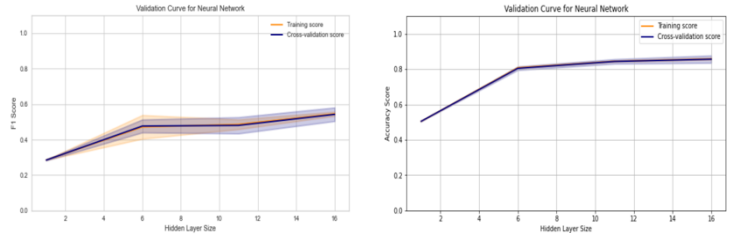

Figure 4. Cross Validation curves for dataset 1 (left column) and dataset 2 (right column)

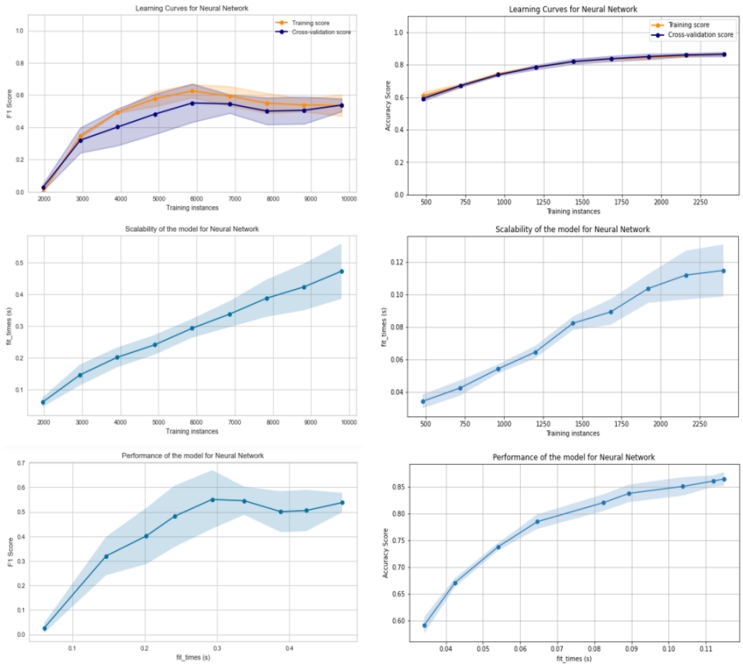

Figure 4. Learning rate curves, scalability, and performance curves for dataset 1 (left column) and dataset 2 (right column)

For the neural network classifier, I tuned the hyperparameters "hidden layer size". The range of values tested for the "hidden layer size" hyperparameters for both datasets were 1 through 20 with a step size of 5.

In the cross validation curves for dataset 1, we can see that the Neural Network classifier shows very little difference between the training score and the cross validation score when changing the value of the hidden layer size. This suggests that a different hyperparameter could have been tested and tuned to improve the neural networks performance. Through applying the GridSearchCV method from scikit learn to test optimal parameters, it was determined that the optimal value of hidden layer size was 6. Additionally, the learning rate curve shows that both the validation score and the training score converge to a value that is *much higher* with increasing size of the training set. Thus, we will probably benefit *greatly* from more training data. Finally, Performance of the Neural Network classifier was 0.879 with the optimized value for the hidden layer size hyperparameter.

In the cross validation curves for dataset 2, we can see that the Neural Network classifier shows very little difference between the training score and the cross validation score when changing the value of the hidden layer size. This suggests that a different hyperparameter could have been tested and tuned to improve the neural networks performance. Through applying the GridSearchCV method from scikit learn to test optimal parameters, it was determined that the that the optimal value of hidden layer size was 16. Additionally, similar to the learning rate curve for dataset 1, the learning rate curve for dataset 2 shows that both the validation score and the training score converge to a value that is *much higher* with increasing size of the training set. Thus, we will probably benefit *greatly* from more training data. Finally, the Performance of the Neural Network classifier was 0.856 with the optimized value for the hidden layer size hyperparameter.

3. Gradient Boosting (GBC)

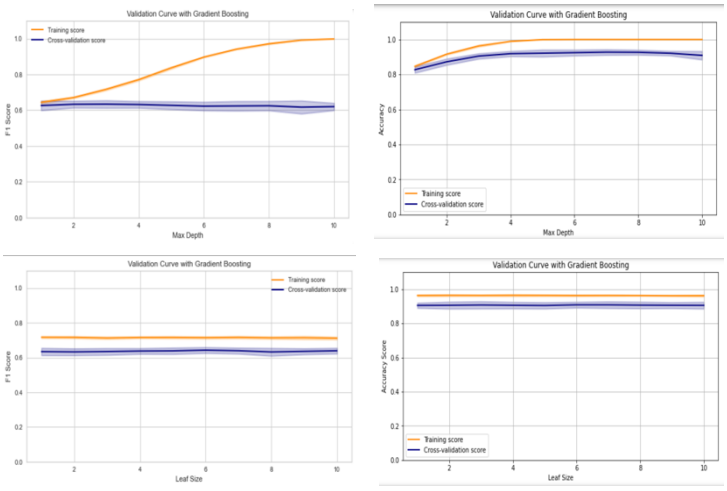

Figure 5. Cross Validation curves for dataset 1 (left column) and dataset 2 (right column)

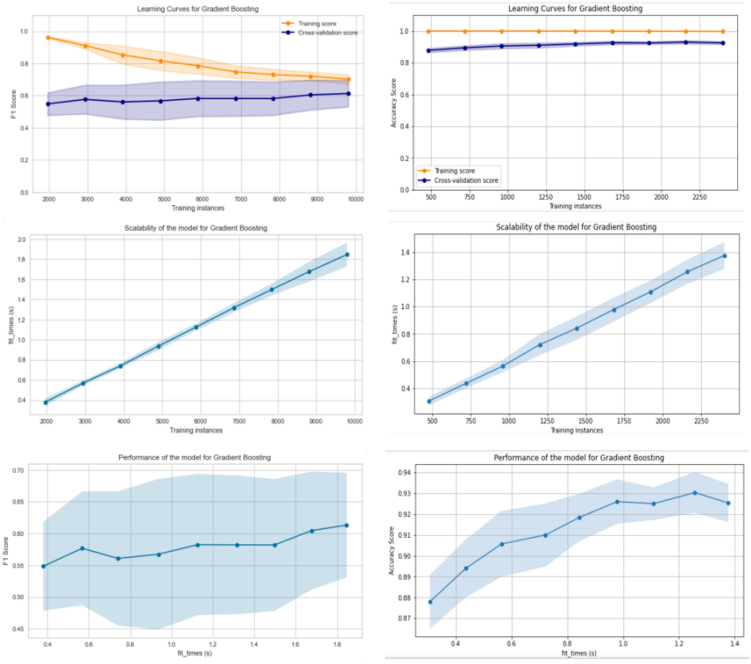

Figure 5. Learning rate curves, scalability, and performance curves for dataset 1 (left column) and dataset 2 (right column)

For the gradient boosting classifier, I tuned the hyperparameters "max depth" and "leaf size". The range of values tested for the "max depth" and "leaf size" hyperparameters for both datasets were 1 through 11 with a step size of 1 when creating the cross validation curves and 1 through 6 with a step size of 1 when creating the learning curves.

In the cross validation curves for dataset 1, we can see that the Gradient Boosting classifier shows the ideal range of max depth at under 2, while very little difference between the training score and the cross validation score when changing the value of the leaf size. Overfitting is seen when max depth is above 2. Through applying the GridSearchCV method from scikit learn to test optimal parameters, it was determined that the optimal value of max depth was 3, while the optimal value of leaf size was 5. Additionally, the learning rate curve shows that both the validation score and the training score converge to a value that is *quite low* with increasing size of the training set. Thus, we will probably *not* benefit much from more training data. Finally, the Performance of the Gradient Boosting classifier was 0.899 with the optimized values for the max depth and leaf size hyperparameters.

In the cross validation curves for dataset 2, we can see that the Gradient Boosting classifier shows the ideal range of max depth at under 4, while very little difference between the training score and the cross validation score when changing the value of the leaf size. Overfitting is seen when max depth is above 4. Through applying the GridSearchCV method from scikit learn to test optimal parameters, it was determined that the optimal value of max depth was 5, while the optimal value of leaf size was 4. Additionally, similar to the learning rate curve for dataset 1, the learning rate curve for dataset 2 shows that both the validation score and the training score converge to a value that is *higher* with increasing size of the training set. Thus, we will probably benefit *somewhat* from more training data. Finally, the Performance of the Gradient Boosting classifier was 0.925 with the optimized values for the max depth and leaf size hyperparameters.

4. k-Nearest Neighbors (kNN)

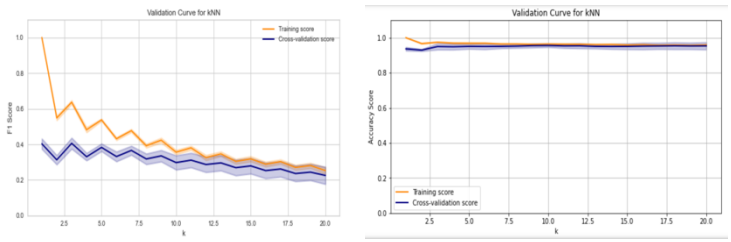

Figure 6. Cross Validation curves for dataset 1 (left column) and dataset 2 (right column)

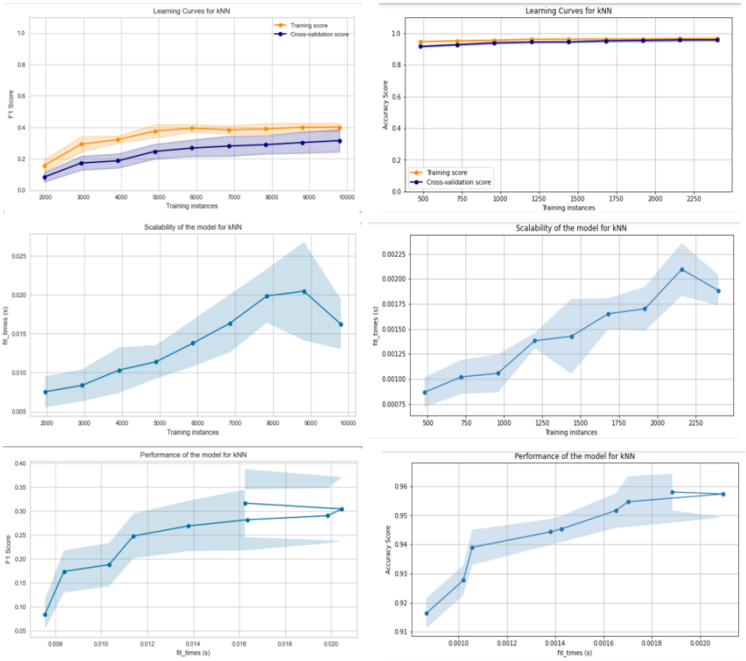

Figure 6. Learning rate curves, scalability, and performance curves for dataset 1 (left column) and dataset 2 (right column)

For the k-Nearest Neighbor classifier, I tuned the hyperparameter “n-neighbors”.

For dataset 1, the optimal value of n-Neighbors was 8. The Performance of the kNN classifier was 0.863 with the optimized value for the n_neighbors hyperparameters.

For dataset 2, the optimal value of n-Neighbors was 10. The Performance of the kNN classifier was 0.956 with the optimized value for the n_neighbors hyperparameters.

5. Support Vector Machines (SVM)

Citations:

Sakar, C.O., Polat, S.O., Katircioglu, M. et al. Real-time prediction of online shoppers' purchasing intention using multilayer perceptron and LSTM recurrent neural networks. Neural Comput & Applic 31, 6893–6908 (2019). https://doi.org/10.1007/s00521-018-3523-0

Pedregosa, Fabian, et al. "Scikit-Learn: Machine Learning in Python." Journal of Machine Learning Research, vol. 12, no. 85, 2011, pp. 2825–30.

Buitinck, Lars, et al. "API Design for Machine Learning Software: Experiences from the Scikit-Learn Project." ArXiv:1309.0238 [Cs], Sept. 2013. arXiv.org, http://arxiv.org/abs/1309.0238.

"3.4. Validation Curves: Plotting Scores to Evaluate Models." Scikit-Learn, https://scikit-learn/stable/modules/learning_curve.html. Accessed 27 Sept. 2021.