Supervised Learning Assignment

Chase Brooks

dbrooks43@gatech.edu

1 DATASET OVERVIEW

1.1 Email Spam Dataset

The email spam dataset is a collection of 4.6k emails classified as either legitimate or spam. The data consists of various work frequencies commonly used in spam emails such as “free” and “money” as well as indicators of emails likely to be legitimate, such as coming from a .edu email address.

This dataset is particularly interesting, because everyone who owns an email address has suffered from barrages of spam email. Email providers have worked hard to develop spam filters, and analyzing email contents for evidence of spam can lead to powerful spam prediction capabilities. This dataset shows that analyzing email contents on their own can yield great results, but adding in additional information such as sender IP and DNS information as well as user reported spam flags will likely improve this dataset’s predictive capability.
Link to dataset: https://www.openml.org/d/44

1.2 Churn Dataset

The churn dataset contains 5k customer account records with various fields that indicate customer tenure, usage, and area with the aim of predicting whether or not a customer will churn. In subscription-based businesses, churn is a critical component of maintaining profitability. If a company is able to determine someone will terminate their account before they actually do so, the company has the opportunity to intervene and hopefully retain that customer. Many companies employ complex protocols to monitor churn and ensure customer satisfaction, and this dataset can give us exposure into determining how a customer’s behavior can be used to predict whether or not they will churn.
Link to dataset: https://www.openml.org/d/40701

2 DECISION TREE

2.1 Decision Tree Pruning

In an effort to reduce tree complexity and overfitting, I incorporated two forms of decision tree pruning: limiting maximum depth and cost complexity pruning. In the spam dataset, overfitting can clearly be seen when the max tree depth passes 20, since training score continues to increase as testing score decreases. Both datasets show a flattening of training scores near 1.0 as maximum depth increases, which further indicates overfitting. Cost complexity pruning is a method of simplifying a decision tree by pruning the node with the poorest predictive performance. Sci-Kit Learn provides a parameter called ccp_alpha (CCP Alpha) that allows us to easily test a decision tree’s performance at various CCP Alpha levels to decide how much pruning we want to do.

2.2 Decision Tree Hyperparameter Tuning and Performance

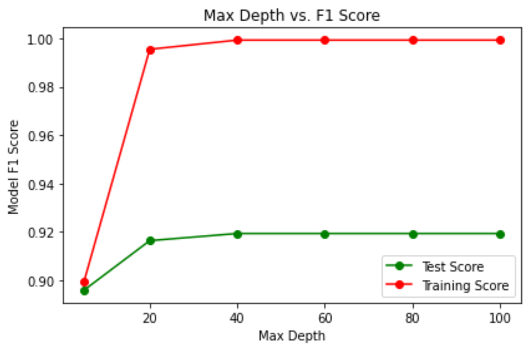
I used Sci-Kit Learn’s GridSearchCV method to search over a finite list of max depth and CCP Alpha values and produce the highest score. The best hyperparameters and scores for each dataset are as follows:

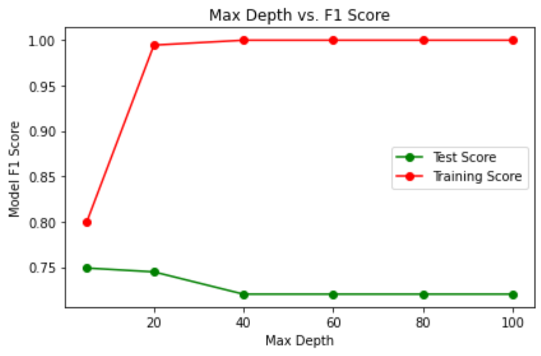
Dataset	Max Depth	CCP Alpha	Train Score	Test Score
Spam	5	0.01	88.8%	88.5%
Churn	5	0.01	92%	92%

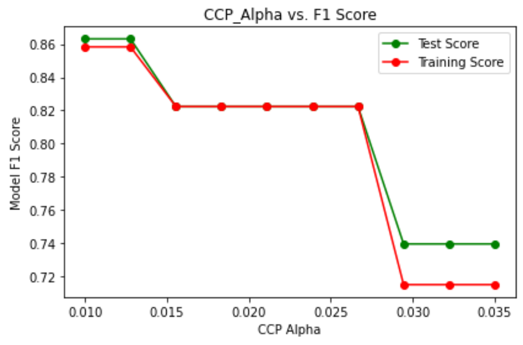
2.3 Decision Tree Validation Curves

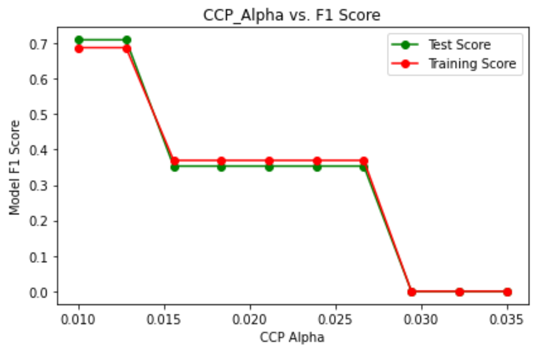
Spam Dataset:

Churn Dataset:





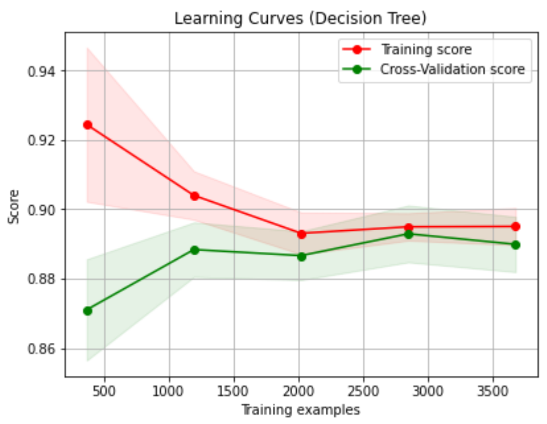




2.4 Decision Tree Learning Curves

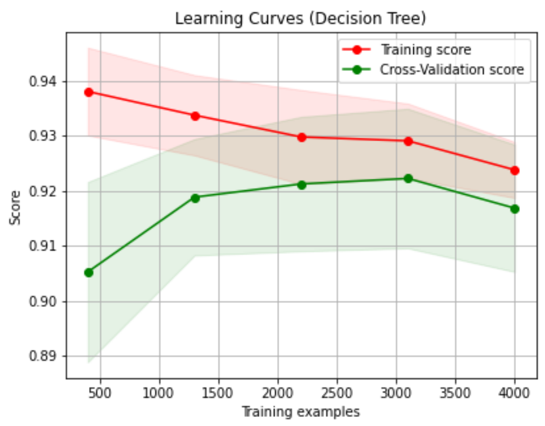
Once I determined the best combination of Max Depth and CCP Alpha, I held out 20% of samples as a test set for final score calculations. On the remaining 80% of data, I implemented a 5-fold cross-validation to generate learning curves. The decision tree for the spam dataset begins to generalize around 1200 samples.

Spam Dataset:



3

Churn Dataset:



NEURAL NETWORK

Each dataset used Sci-Kit Learn's MLPClassifer with the Adam solver and a ReLU activation function to build a feedforward, multi-layer perceptron neural network.

3.1 Neural Network Hyperparameter Tuning and Performance

I again used Sci-Kit Learn's GridSearchCV method to search over a list of hidden layer sizes and learning rates to produce the highest score. The best hyperparameters and scores for each dataset are as follows:

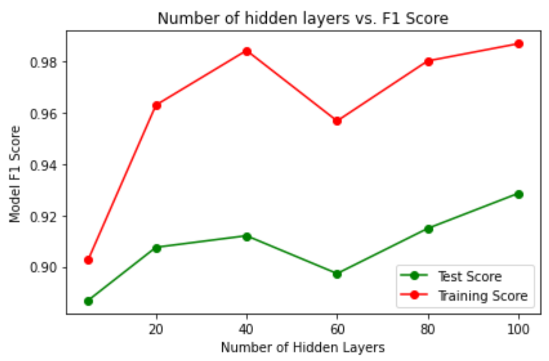
Dataset	Learning Rate	Hidden Layers	Train Score	Test Score
Spam	0.1	60	93%	93%
Churn	0.5	100	86%	85%

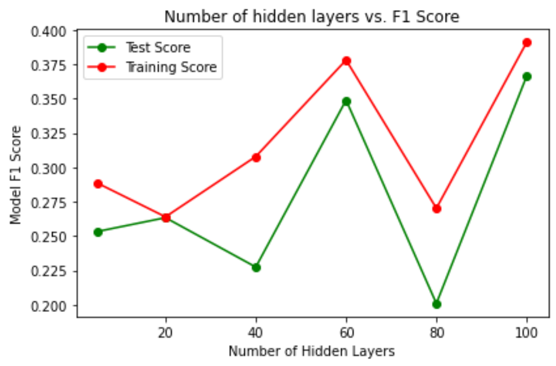
3.2 Neural Network Validation Curves

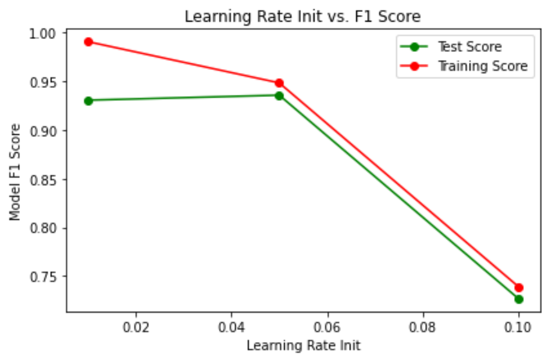
The spam dataset has a moderate gap between test and train performance indicating the neural network may be underfitting. To combat this, we can add more training data or try implementing cross-validation.

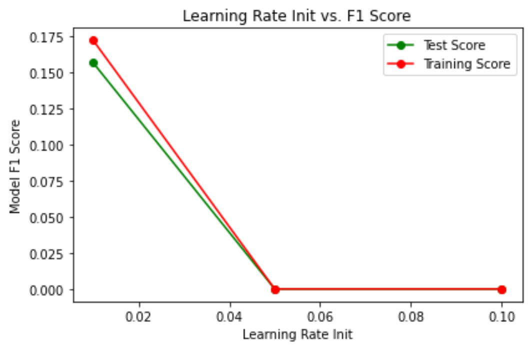
Spam Dataset:

Churn Dataset:





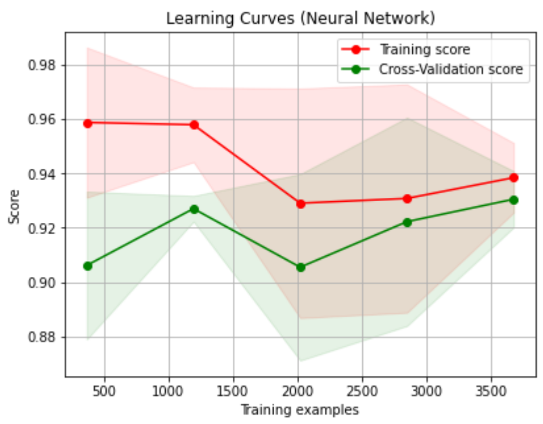




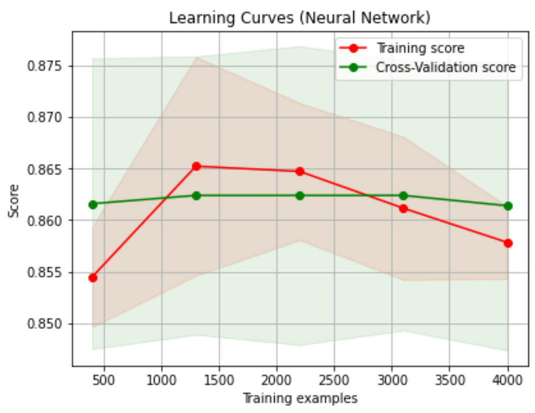
3.3 Neural Network Learning Curves

After determining the optimal combination of hyperparameters, I again implemented a 5-fold cross-validation on the 80% of data used for training. In the spam dataset, model performance initially decreases but later increases around 2000 training samples. This may be caused by the gradient descent optimizer finding a local minimum and getting stuck. The churn dataset, on the other hand, has a roughly similar score across all training size iterations. The results are as follows:

Spam Dataset:



Churn Dataset:



4 BOOSTING (ADABOOST)

I selected the AdaBoost classifier for the boosting section of this assignment, because it weighs "difficult" cases more heavily in successive estimators with the goal of improving performance.

4.1 Decision Tree Hyperparameter Tuning and Performance

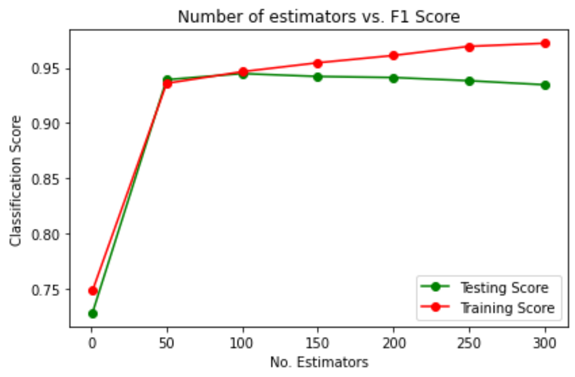
I used Sci-Kit Learn's GridSearchCV method to search over a finite list of number of estimators and learning rate values that yield the highest score. The best hyperparameters and scores for each dataset are as follows. Interestingly, test scores were slightly higher than the train scores for the Spam dataset. This may be due to a smaller proportion of "difficult" cases that the learner will miss in the test set as opposed to the train set.

Dataset	Num. Estimators	Learning Rate	Train Score	Test Score
Spam	100	0.1	93%	94%
Churn	100	0.1	88%	87%

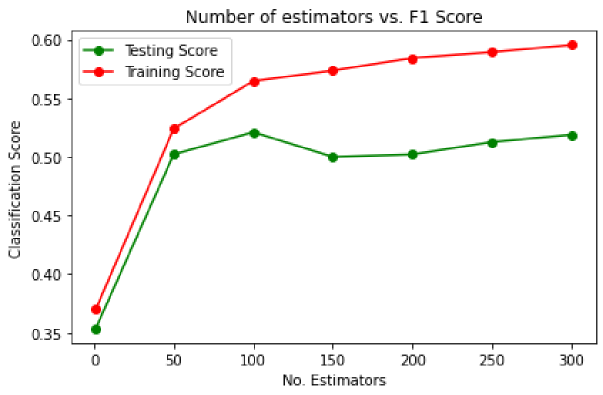
4.2 AdaBoost Validation Curves

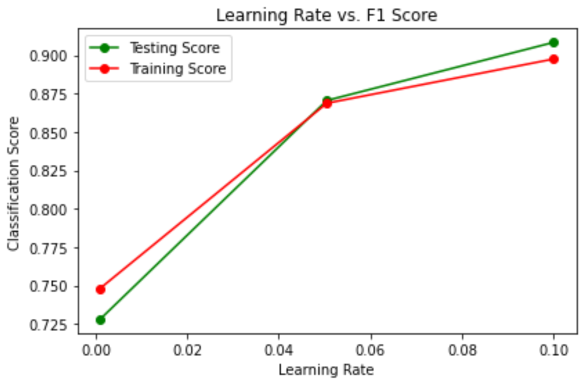
In both datasets, score performance initially increases across both the training and testing sets as we increase the number of estimators to 100. After this point, training scores continue to increase while testing scores decrease. This is a hallmark indication of model overfit, which means that we should not use more than 100 estimators in our final model.

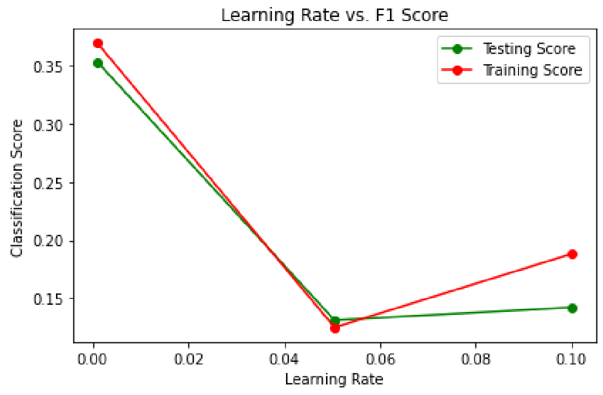
Spam Dataset:



Churn Dataset:



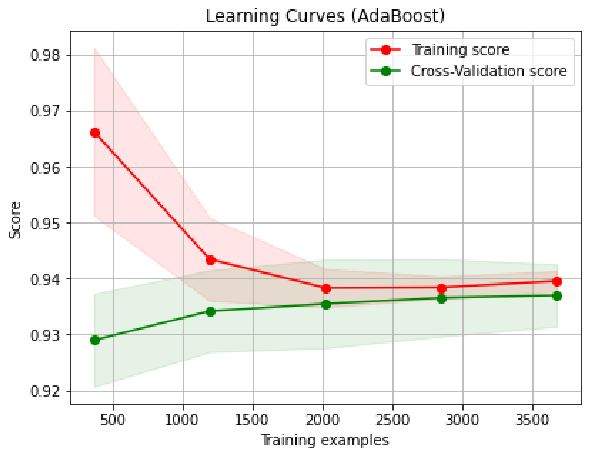




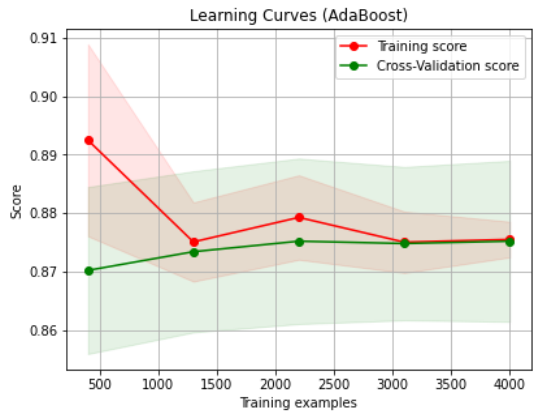
4.3 AdaBoost Learning Curves

In both datasets, training scores initially decrease before rising slightly, and the cross-validation scores increase constantly. This shows how increasing the sample size reduces overfitting.

Spam Dataset:



Churn Dataset:



5 SUPPORT VECTOR MACHINE (SVM)

5.1 SVM Hyperparameter Tuning and Performance

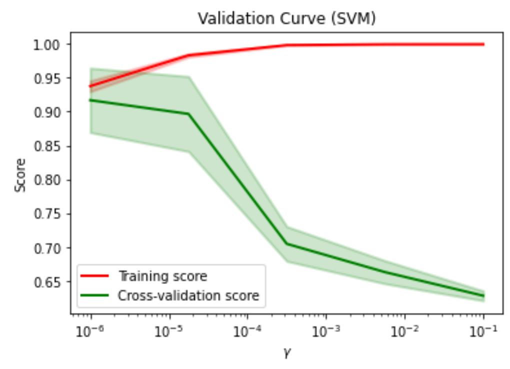
I used Sci-Kit Learn's GridSearchCV method to search over a finite list of regularization parameters, C, and Kernel Coefficients (gamma) to find the combination that yields the highest score. The best hyperparameters and scores for each dataset are as follows. For the spam dataset, even after implementing hyperparameter tuning and 5-fold cross-validation, the model yields a wide gap in performance between training and testing scores.

Dataset	C	Gamma	Train Score	Test Score
Spam	10	0.0001	99.8%	81.5%
Churn	0.1	1	87%	85%

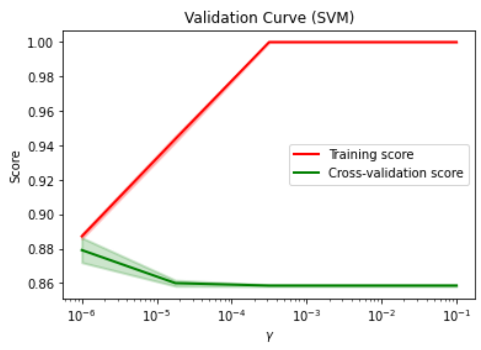
5.2 SVM Validation Curves

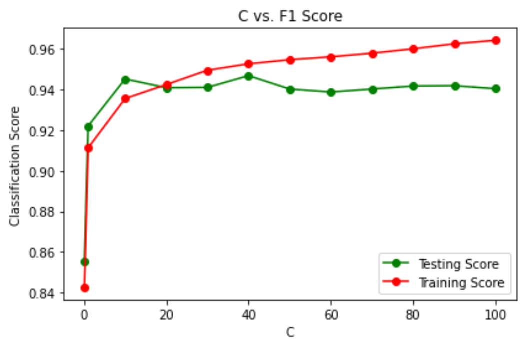
In both datasets, as we increase gamma, training scores increase as testing scores decrease, which indicates that we should choose a lower gamma value to avoid overfitting.

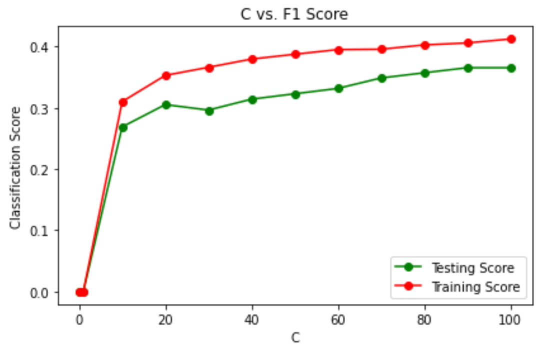
Spam Dataset:



Churn Dataset:

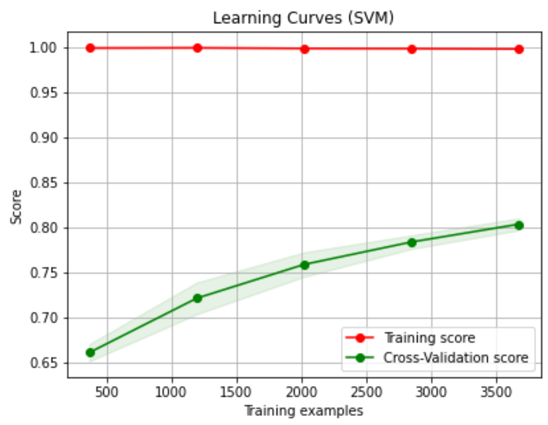




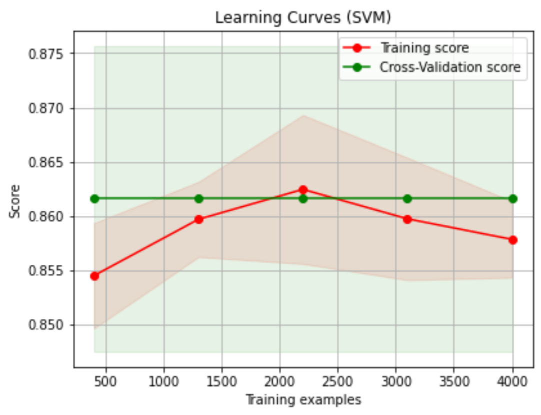


5.3 SVM Learning Curves

Spam Dataset:



Churn Dataset:



6 K-NEAREST NEIGHBORS (KNN)

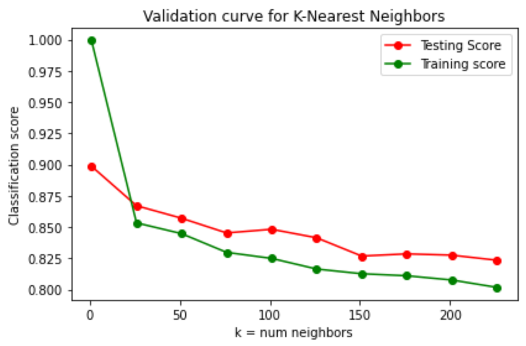
6.1 KNN Hyperparameter Tuning and Performance

I used Sci-Kit Learn's GridSearchCV method to search over a finite list of various numbers of neighbors to find the one that yields the highest score. The best hyperparameters and scores for each dataset are as follows. For the spam dataset, even after implementing hyperparameter tuning and 5-fold cross-validation, the model yields a gap in performance between training and testing scores.

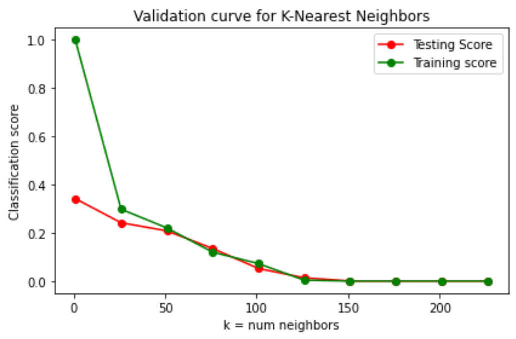
Dataset	Num. Neighbors	Train Score	Test Score
Spam	10	99.8%	81.5%
Churn	20	88%	86%

6.2 KNN Validation Curves

Spam Dataset:

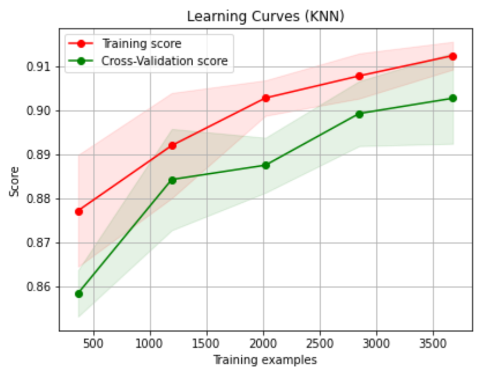


Churn Dataset:

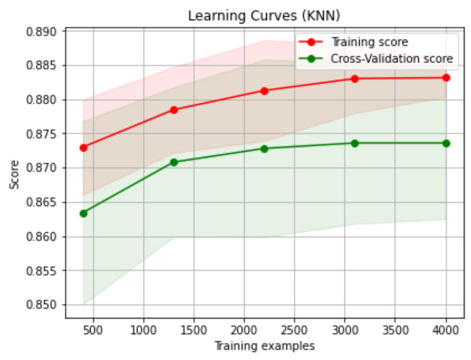


6.3 KNN Learning Curves

<u>Spam Dataset:</u>



<u>Churn Dataset:</u>

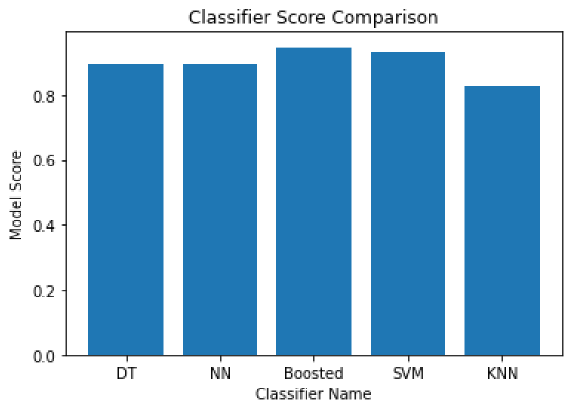


7 CLASSIFIER COMPARISON

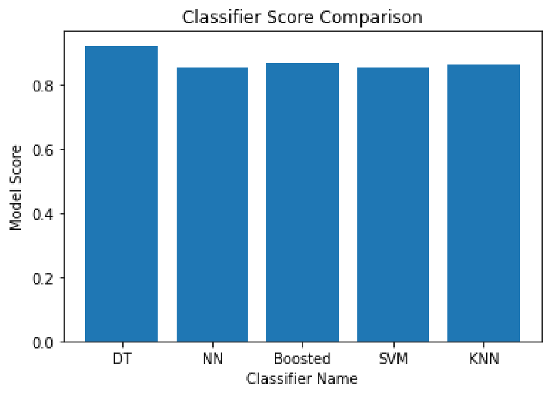
7.1 Test Set Accuracy Comparison

I will first compare model scores across the 5 tuned models on the 20% of data held out for testing.

<u>Spam Dataset:</u>



<u>Churn Dataset:</u>

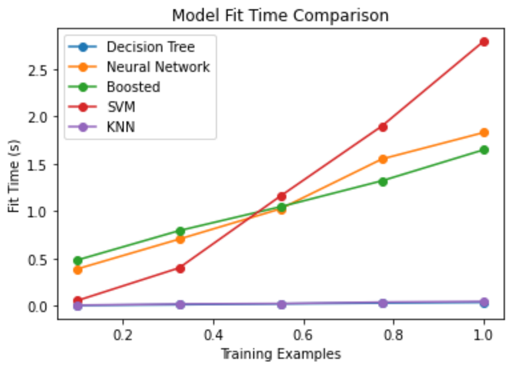


<u>Spam Dataset:</u> While all datasets attain an 80%+ score, the Adaboost has the clear advantage in terms of model score. Somewhat counterintuitively, KNN attains the lowest score of the 5 models compared. I have read multiple papers about fraud research that outline specific approaches of KNN to detect fraud anomalies. While there is definitely still room to tune parameters further and introduce additional training data, this example shows how data source and quality can have a large impact on model performance and should factor heavily into a model decision choice.

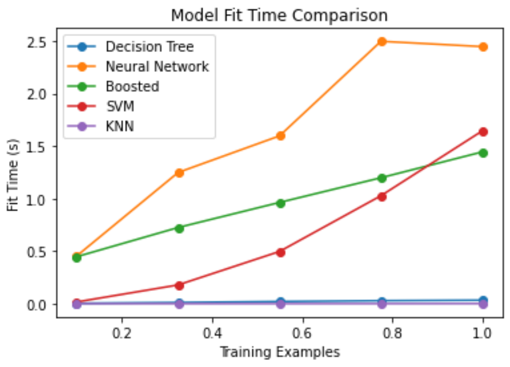
Churn Dataset: In this dataset, the decision tree clearly yields the highest predictive power. The other 4 models produce similar model scores (~86%). One possible explanation for this difference is that these models may be underfitting, and there may be room to further tune the hyperparameters of these 4 models . For example, the neural network reached the maximum number of iterations multiple times during training, which indicates the model has not yet converged and is likely underfitted.

7.2 Model Fit Time Comparison

Spam Dataset:



Churn Dataset:



In both datasets, Decision Trees and KNN had the shortest training time by a large margin. On the other hand, the Neural Network, AdaBoost, and SVM learners had the longest training time in both datasets. This is due to the more complex methods of learning employed by these models, which also allowed them to be among the highest scoring models analyzed in this paper.

8 CONCLUSION

Every machine learning model is only as good as its data. While I spent a great deal of time optimizing hyperparameters and researching the best methods to employ these machine learning techniques, I discovered that the single most important predictor of a “good” model is how good its data is. This definition of “good” is somewhat arbitrary, but, in this case, I am strictly referring to maximizing model scores. I tried implementing these techniques on a variety of datasets before landing on the spam and churn datasets. Some of the common challenges I ran into were imbalanced datasets, in which one target variable occurred significantly more than the other, small datasets, in which models had difficulty converging, and datasets with low correlation between features and target variables. In most real-world situations, we do not have the luxury of choosing another dataset if we run into any of the aforementioned challenges. Instead, we must employ additional advanced techniques to remedy the problems, such as resampling, better feature selection, etc.

This analysis both confirmed and directly contradicted some of my initial hypotheses. For example, I initially thought that the decision tree would take the least amount of time to train, and the neural network would take the most time. This held true for the churn dataset, but my implicit assumption was that model score would correlate to training time. Thus, I expected the neural network would have the highest model score in all scenarios. In the case of the churn dataset, the neural network took the longest time to train and also yielded among the lowest model scores. While there is still room to improve the neural network, I did not expect this behavior. I also learned the power of Sci-Kit Learn's grid search functionality for hyperparameter selection and how cross-validation can increase model performance. In future iterations of this analysis, I would like to explore additional hyperparameters and explore various numbers of folds of cross-validation (I used 5-fold cross-validation in this project).

9 REFERENCES

1. "Sci-Kit Learn Documentation." *Scikit*, https://scikit-learn.org/stable/.
2. "SVM Validation Curves." *Scikit*, https://scikit-learn.org/stable/auto_examples/model_selection/plot_validation_curve.html.
3. "SVM Hyperparameter Tuning Using GRIDSEARCHCV: ML." *GeeksforGeeks*, 4 Oct. 2021, https://www.geeksforgeeks.org/svm-hyperparameter-tuning-using-gridsearchcv-ml/.
4. "Plotting Learning Curves." *Scikit*, https://scikit-learn.org/stable/auto_examples/model_selection/plot_learning_curve.html.