Assignment 1 - Supervised Learning

Prithi Bhaskar
pbhaskar8@gatech.edu

1 Classification Problems

Two classification problems have been identified for the purpose of this assignment. They were chosen based on their ability to enable comparisons and analysis of various supervised learning algorithms. The details of the problems and their respective data sets have been discussed in detail in the following sections.

1.1 Problem 1

Problem1 is a binary classification problem that aims to classify if the income of a person is greater than 50K based on several census parameters, such as age, education, marital status, hours of work per week, etc. The dataset used for solving the same is the Adult dataset from the UCI Machine learning repository. This dataset consists of 48842 instances and 13 variables, of which 32,561 instances are presented as training set and the remaining constitute the test set. The Adult dataset is a slightly imbalanced one with the majority class representing 75% of the dataset. Oversampling was done to account for the imbalance and relevant metrics chosen to analyse the performance of various algorithms.

1.2 Problem 2

Problem 2 is a multi label classification problem involving the Avila dataset from the UCI Machine learning repository. The Avila data set has been extracted from 800 images of the 'Avila Bible', an XII century giant Latin copy of the Bible. The prediction task is to associate each pattern of the script to a copyist. The features include numerical data such as intercolumnar distance, margins, interlinear spacing, etc. The dataset consists of a training set of 10430 samples, and a test set of 10437 samples. There are 12 classes to which the data belong to and the distribution is imbalanced with the majority class representing 4286 instances and the minority class representing 5 instances. Oversampling was done to account for the imbalance and the model's performance on the test set was evaluated with the original imbalanced testset.

1.3 What makes the datasets interesting?

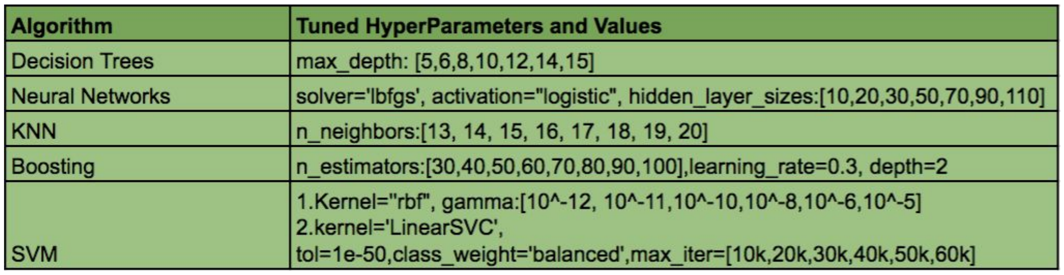
One of the main factors that makes the datasets interesting is that the data sets carry a non-trivial amount of data as well as features, thereby requiring algorithms that can make complex decision boundaries and at the same time are able to generalise well. This attribute of the problems render them flexible for doing a lot of experiments and thus help to gain insights into the limitations and strengths of various supervised learning algorithms.

2 Problem 1 - Analysis

2.1 HyperParameters

An initial round of experiments was performed by passing different values of hyper parameters to the respective classifiers. The parameters that yielded significant changes to the models' performance were selected for further tuning. The below table summarizes the hyperparameters chosen for further tuning. Different ranges of values were passed to the hyperparameters and the values that yielded the best performance are summarized below. Rest of the hyperparameters assume default values.

Table 1 — Tuned Hyperparameters(Problem1)



Algorithm	Tuned HyperParameters and Values
Decision Trees	max_depth: [5,6,8,10,12,14,15]
Neural Networks	solver='lbfgs', activation="logistic", hidden_layer_sizes:[10,20,30,50,70,90,110]
KNN	n_neighbors:[13, 14, 15, 16, 17, 18, 19, 20]
Boosting	n_estimators:[30,40,50,60,70,80,90,100],learning_rate=0.3, depth=2
SVM	1.Kernel="rbf", gamma:[10^-12, 10^-11,10^-10,10^-8,10^-6,10^-5] 2.kernel='LinearSVC', tol=1e-50,class_weight='balanced',max_iter=[10k,20k,30k,40k,50k,60k]

2.2 Metrics

Since the dataset is imbalanced, the metrics that take into account, false negative rates and false positive rates were considered. Among such metrics, ROC AUC and F1 Score were chosen for evaluating the Adult dataset since they both are widely accepted as good metrics when evaluating imbalanced data.

2.3 Analysis

Initially, for all the algorithms, the chosen hyperparameters were assigned a range of values and the performance metric(F1 Score) was recorded as a function of the hyperparameter values. This helped in identifying the values for the hyperparameters that yielded the best performance, which were then used to plot learning curves to further evaluate the models.

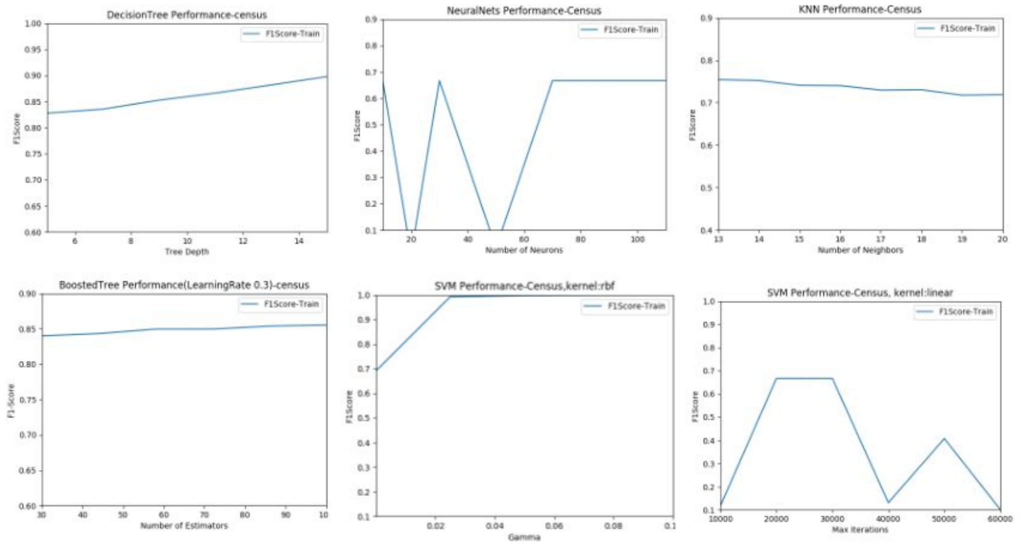


Figure 1 Performance of the SL Algorithms as a function of hyperparameters

2.3.1 Validation Curves - Analysis

From the above figures, it can be observed that some of the algorithms perform better on the data when compared to others. For example, decision trees fit the training data with increasing F1-Score as the tree is allowed to grow deeper. Here, an F1-score of 0.90 on the training set can be perceived as a fairly good score since the metric takes both false positives and false negatives into account and by definition, it emphasises the lowest value. Boosting the decision trees is found to increase the performance even with aggressive pruning (max_depth=2)as the number of estimators(weak learners) is increased.

SVM model with rbf kernel achieves a score of 1 for a gamma value of 0.03 and then stabilizes. The rbf kernel is able to perfectly define boundaries around the data and hence is the best kernel model among others. Among the other models, the linear SVM achieves a better score compared to the other models.

The performance of neural networks was low and as shown in the figure, it keeps oscillating between 0.7 and 0.1 and then stabilizes at 0.7. Similar trends can be observed for knn as well. The value remains roughly the same as the values for number of neighbors are changed. These behaviors can be explained by studying the data set and its features. The values of the features belong to a wide range and hence normalising them might lead to better performance. Another way to increase the performance of these three models is to leverage domain knowledge and implement feature reduction techniques to overcome the "Curse of dimensionality." The second solution was implemented by empirically reducing the dimension of features and by tuning the hyperparameters to the new data. The results are summarized below.

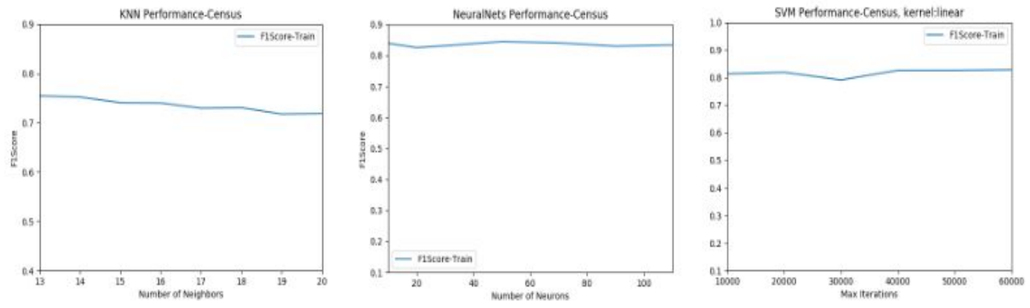


Figure 2 Performance of the SLAlgorithms after applying Feature Reduction

As was expected, all the three models performed better after applying feature reduction, some even matching the performance of decision trees. However, feature reduction had no impact on the performance of decision trees and boosted trees. This is because decision trees inherently choose only the features that provide the maximum information gain and ignore the ones that have high impurity. For this instance, the decision tree algorithm calculates gini impurity of a feature to measure the quality of splits in the tree.

As a next step, the features that yielded the best performance in all models were chosen and passed to the models and the respective learning curves were plotted to perform model complexity analysis. The scoring metric used for plotting learning curves is ROC AUC(a measure of sensitivity). This was done to verify that the models do not lag on metrics other than precision and recall.

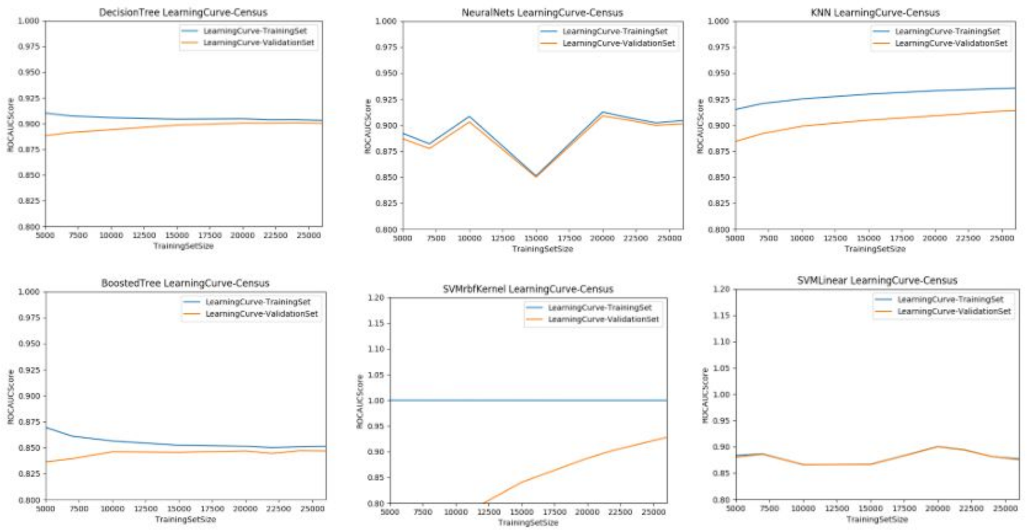


Figure 3 Learning Curves of the SL Algorithms

2.3.2 Learning Curves - Analysis

It can be observed that all of the models have a low variance since the training and validation scores are converging but also not a very high bias since the training score is around 0.9. At this point, the models need more features which will increase the complexity of the models thus making them capable of understanding the data better.

2.3.3 Performance of models on Test set

Apart from the chosen performance metric, the average time taken by the models to fit the data during a 5-fold cross validation was also recorded in order to evaluate the speed of the models. The same has been summarised below.

Table 2 — A summary of fit times of the models

Algorithm	Decision Trees	NeuralNets	KNN	Boosting	SVM(RBF Kernel)
Average Fit time(In Seconds)	1.581945	11.161161	0.998794	76.93318	37.8588

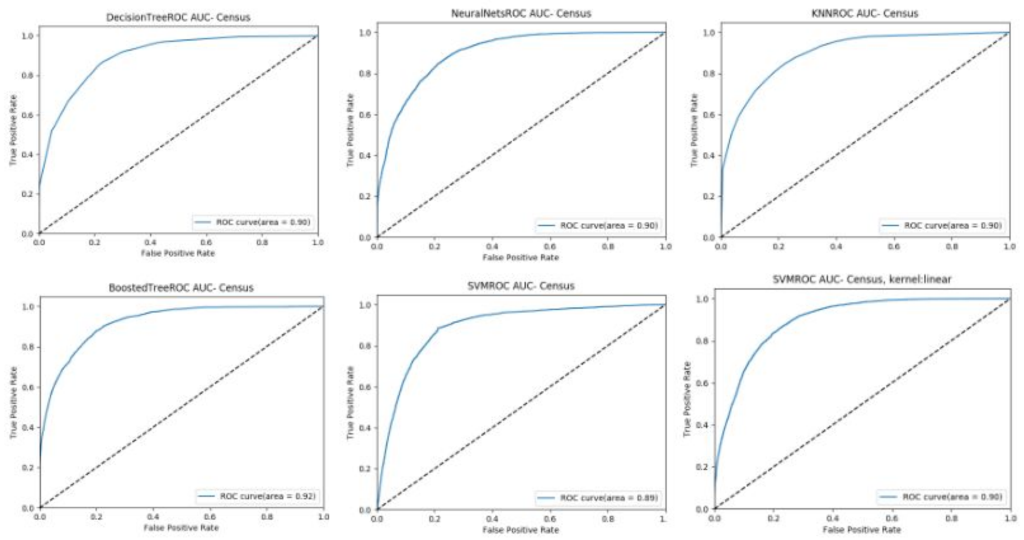


Figure 4 Performance of the SL models on Testset(Adult)

2.3.4 Best Performing Model

In this problem(the Adult dataset), boosted decision trees perform the best in terms of precision and recall but perform the worst in terms of speed. This is because of the number of weak learners used to learn the data and generalise. All other models perform equally well but differ in fit times. SVM lags behind other models by a huge factor in speed. KNN’s fit time is the lowest as it is a lazy learner. However, KNN’s prediction time is high making it undesirable for scaling. The best algorithm is thus identified based on the tradeoff between several factors like speed, accuracy and complexity of the models.

3 Problem 2 - Analysis

3.1 HyperParameters

As in problem1, an initial round of experiments was performed by passing different values of hyper parameters to the respective classifiers. The parameters that yielded significant changes to the models’ performance on training data were selected for further tuning. The values that yielded the best performance are summarized below. Rest of the hyperparameters assume default values.

Table 3— Tuned Hyperparameters(Problem2)

Algorithm	Tuned HyperParameters and Values
Decision Trees	max_depth:[8,9,10,11,12,13,14,15]
Nerual Networks	solver='sgd',activation='tanh',hidden_layer1_sizes:[20,30,40,50,60,70,80,90], hidden_layer2_sizes=20
KNN	n_neighbors:[1,2,3,4,5,6,7,8,9]
Boosting	n_estimators:[30,40,50,60,70,80], learning_rate=0.2, depth=7
SVM	1.kernel='rbf', gamma:[0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8] 2.kernel='poly', gamma:[0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1.0]

3.2 Metrics

Weighted F1 score - F1score calculated for each label and averaged, weighted by the number of true instances for each label and confusion matrix were used to evaluate the performance of the models. These two metrics were chosen because they take into account, the model's performance in each of the labels and hence are a good measure of the overall performance.

3.3 Analysis

As in Problem1, the chosen hyperparameters were assigned a range of values and the performance metric(weighted F1 Score) was recorded as a function of the hyperparameter values. This helped in identifying the values for the hyperparameters that yielded the best performance, which were then used to plot learning curves to further evaluate the models.

3.3.1 Validation Curves - Analysis

After a couple of rounds of experimenting and tuning the hyper parameters, all the models were able to learn the training data equally well. However, as can be observed in Figure 4, few models achieve their highest performance results only when the parameters take extremely high or low values, which might bring down the models' ability to generalise. For example, decision trees achieve a score above 0.95 only when the tree is allowed to grow till depth 15. Neural Networks achieve good results at various combinations of hidden layer sizes as shown in the figure but the results are not consistently improving or declining.

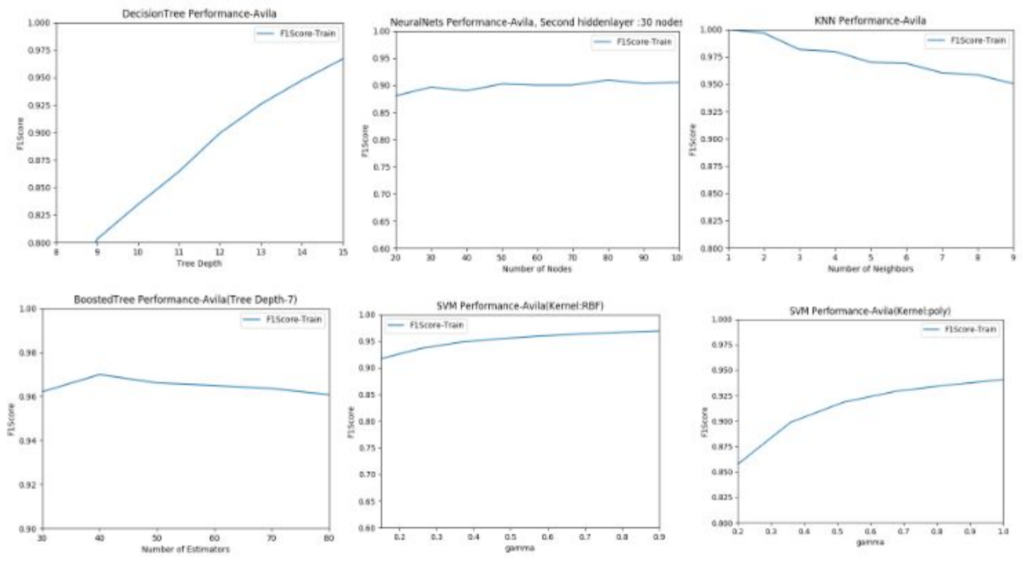


Figure 5 Performance of the SL models as a function of hyperparameters

KNN achieves a score of 1 when the number of neighbors is 1 and keeps decreasing as the number is increased. One characteristic of the training data that best explains this behavior is that the nearest neighbor of a data point is always of the same class as the data point itself. But this might not be representative of the unseen test set. Boosting the decision trees yields better performance results when the tree is allowed to grow till depth 7. The SVM kernels that yield the best results are ‘rbf’ and ‘poly’ and both models yield good results as gamma is increased and after a point, the score stabilizes.

Learning curves were plotted to study the models' behavior with unseen data and to perform model complexity analysis. The learning curves have been shown below. Scores shown in the figures are the average of scores obtained from a 5-fold cross validation on the training data. The hyperparameters were passed the values that yielded the best performance from the validation curves to analyse the models for bias-variance trade-offs.

3.3.2 Learning Curves - Analysis

It can be observed from Figure 6 that all the models have a low bias but also a low variance except neural networks. Neural networks model has a high bias with respect to the reduced dataset and is underfitting the data. But the performance of neural networks was as good as the other models with the full training data. This illustrates one of the key weaknesses of using neural networks - the need for more data than other machine learning algorithms. This can be fixed by further tuning the hyperparameters such that the model performs equally well with lesser data or by adding additional data to the training set.

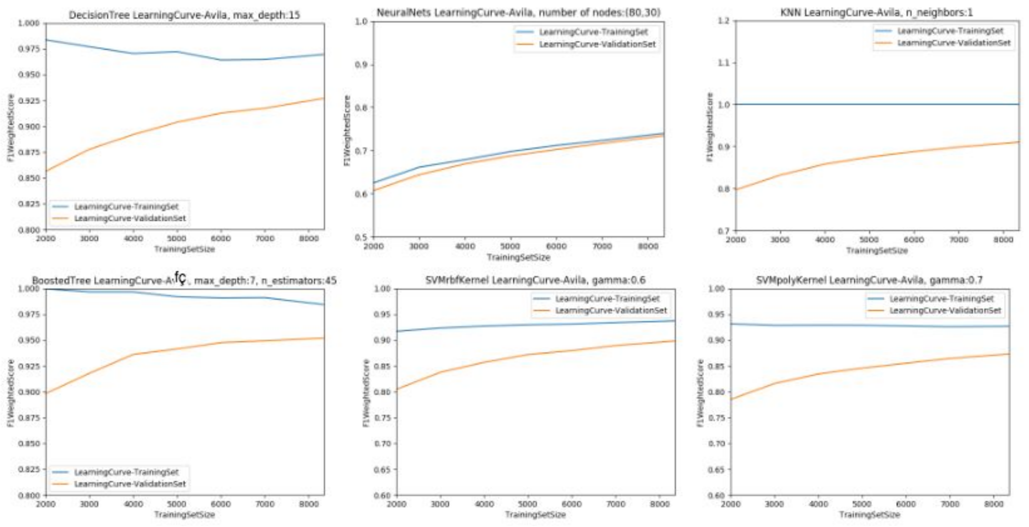


Figure 6 Learning Curves of the SL Algorithms-Avila

On the other hand, KNN with number of neighbors as 1 has a low bias as well as low variance. The behaviour reinforces the structure of data inferred from the validation curves. The model is able to perform well with the validation set since the whole of training data follows the same pattern - the nearest neighbor is of the same class as the data point being looked at. But at this point there is no way to predict if the model will be able to generalise well when tested on new data. One way to further evaluate the model is to look for more data (if available) that defy the current structure and add it to the dataset. But that would certainly lead to a decline in performance with n_neighbors set to 1 and hence it has to be tuned to new values.

Decision trees as well as boosted trees perform well on both training and validation data without any overfitting. So the models have low bias as well low variance and can be evaluated with the test set. Both the SVM models have low bias and low variance and hence do not call for any changes to the model's design. However, the SVM rbf kernel outperforms the SVM poly kernel and hence might perform better with the test set.

3.3.3 Performance of models on Test set

The performance of the models on the test set were evaluated and plotted in order to compare the performances of the models. The same has been shown below.

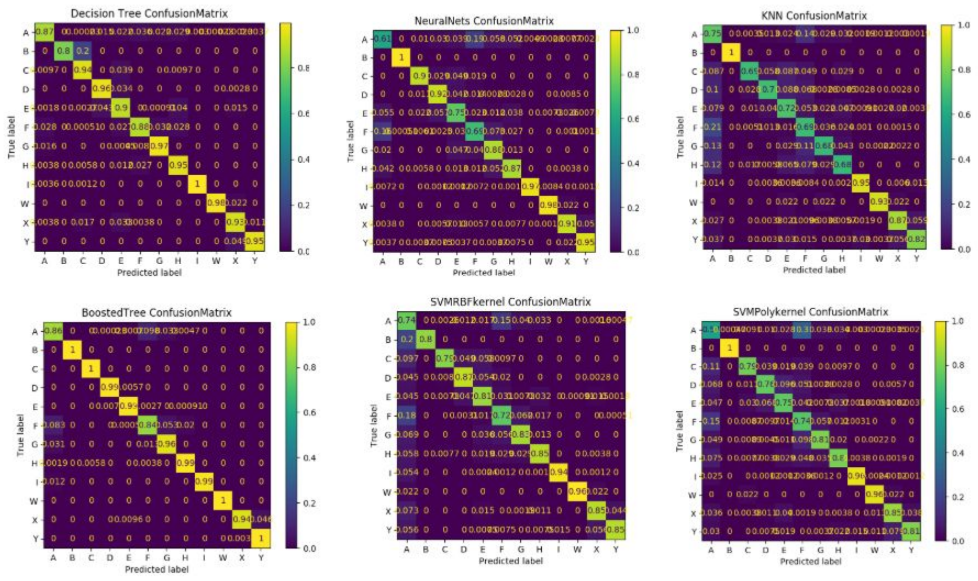


Figure 7 Confusion Matrices of the SL Algorithms-Avila

As expected, the performance of KNN has declined due to the differing structure and placements of data points from the training set. This can be avoided by increasing the number of neighbors and making the model generalise better. Also, neural networks do not exhibit a performance so low as observed from the training curves. They do perform well when presented with enough data. However, they perform extremely well(a score of 1) only with some of the classes and moderately well in identifying other classes. This can be fixed by adding more data points that belong to the classes missed by neural networks. Decision trees and boosted trees perform equally well with all classes as was observed from the learning curves. Boosted trees are able to generalise even better than the decision trees using an ensemble of learners, each of them only half as deep as the decision trees. SVM with 'rbf' kernel performs better than the one with 'poly' kernel. This is inline with the observations from the learning curves. Like neural networks, SVM also performs well with some classes and moderately well with other classes. As more and more data is added, SVM will be able to define sharp boundaries around each of the classes and generalise better with unseen data.

3.3.4 Best Performing Model

The best performing model, as always, can be defined based on the tradeoffs between several factors such as speed, accuracy and complexity. This being a multilabel classification problem, even more tradeoffs are to be considered such as the need to identify few classes with high precision than the others. If the overall performance of the model is desired to be good(identify all classes with high precision), boosted trees are a good choice for the current set of data, although with a compromise on optimal space complexity. If optimal space complexity is desired, decision trees perform the best with the current set of data. SVM, Neural networks and KNNs would serve as good models if data is scaled to include more instances of each of the classes.

4 References

1. Scikit-learn: Machine Learning in Python, Pedregosa *et al.*, JMLR 12, pp. 2825-2830, 2011.
2. Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake Vanderplas, Arnaud Joly, Brian Holt, Gaël Varoquaux, "API design for machine learning software: experiences from the scikit-learn project", *European Conference on Machine Learning and Principles and Practices of Knowledge Discovery in Databases*, 2013.
3. G. Lemaître, F. Nogueira, and C. K. Aridas, “Imbalanced Learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning,” Journal of Machine Learning Research, vol. 18, no. 17, pp. 1–5, 2017.
4. J. D. Hunter, "Matplotlib: A 2D Graphics Environment", Computing in Science & Engineering, vol. 9, no. 3, pp. 90-95, 2007.
5. Wikipedia contributors. (2020, February 4). Bias–variance tradeoff. In *Wikipedia, The Free Encyclopedia*. Retrieved 07:35, February 7, 2020, from https://en.wikipedia.org/w/index.php?title=Bias%E2%80%93variance_tradeoff&oldid=939182665
6. Oltranu, A. (2018, Jan 03). Tutorial: Learning Curves for Machine Learning in Python.Dataquest.https://www.dataquest.io/blog/learning-curves-machine-learning/
7. Donges, N.(2019, Nov 06). 4 REASONS WHY DEEP LEARNING AND NEURAL NETWORKS AREN'T ALWAYS THE RIGHT CHOICE.builtin.https://builtin.com/data-science/disadvantages-neural-networks