Ray M. Liu
September 5, 2019
CS 7641: Machine Learning
Dr. Isbell

Supervised Learning

Overview: This paper will be a thorough analysis of five different learning algorithms: (Decision Trees with pruning, Neural Networks, Boosting, Support Vector Machines, and K-Nearest Neighbors) on two datasets of my choice. I will be analyzing the different hyperparameters and tuning them to fit my dataset. The goal of this assignment is to develop a better understanding of the trade-offs between the different types of supervised Machine Learning algorithms, a better intuition of what hyperparameters to use for different datasets, and a better understanding of each supervised learning algorithm. I will not be implementing Cross-Validation for this problem because we have a very large dataset. To keep things simple, a Train/Test split should suffice for the assignment.

Dataset:

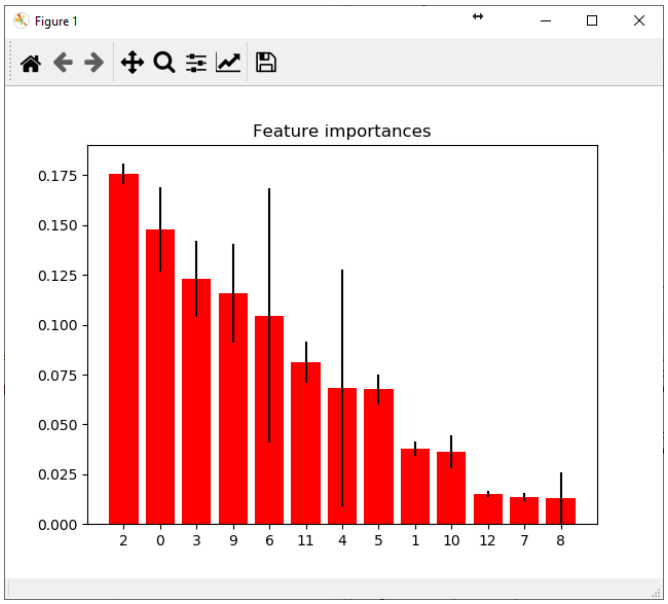
- The first dataset is the "Adult Data Set" from the UCI Machine Learning Repository. The dataset was obtained from the 1994 Census database with 48,842 data points, 14 features, and 2 target categories: More than 50k income or Less than 50k income. The data was extracted using the following conditions: AAGE > 16 and AGI > 100 and AFNLWGT > 1 and HRSWK > 0. The features are composed of a mixture of discrete and continuous variables therefore data preprocessing is a necessary step before being fed into the model.
- The second dataset is the "Graduate Admissions Data Set" from Kaggle. The dataset predicts admission rates based on these features: GRE score, TOEFL score, University Rating, SOP, LOR, Undergraduate GPA, and research experience. The prediction values are modified so that the problem goes from being a regression problem to a classification problem. The threshold value used was 0.7, where admission rates greater than or equal to 0.7 meant high chances of admission (value of 1) and less than 0.7 as being a low chance of being admitted (value of 0).

Data-Preprocessing (Dataset 1 and Dataset 2):

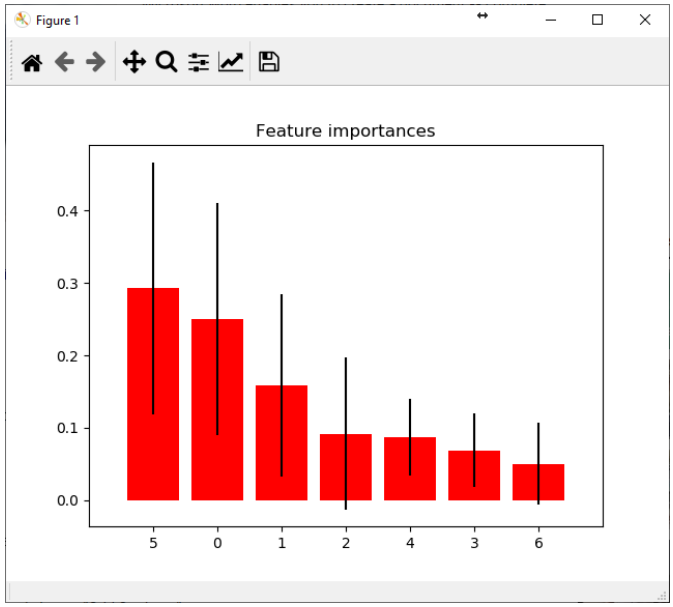
- **Labels**:
  1) The labels are currently '>50k' or '<=50k'. They need to be labeled as 0 or 1.
  2) The labels currently range from 0 to 1.0. They will be converted so that an admission rate of 0.7 means 1 (highly likely) and less than 0.7 means 0 (unlikely)
- **Continuous Variables**:
  1) There are five continuous features: age, fnlwgt, capital-gain, capital-loss, and hours-per-week. They all have different ranges, i.e. age ranges from 0-100, hours-per-week ranges from 0-60, and capital-loss ranges from $-\infty$ to $+\infty$. Therefore, normalization will be required to even out the number ranges.
  2) There are 7 continuous features: serial no., GRE score, TOEFL score, university rating, SOP, LOR, and CGPA. Serial No. serves as an index which provides no information. As for the rest of the data, they will be normalized to mean 0 and std dev 1.
- **Duplicate Feature**:
  1) There exists a duplicate feature, education and education-num, that requires removal.
- **Categorical Variables**:
  1) One-Hot-Encoding is needed for converting certain categorical variables to their number equivalents. However, naively mapping words to numbers will create an implicit relationship that may not be intended. For example, if United-States is mapped to 5 and Puerto-Rico is mapped to 6, the algorithm can mistake United-States as being less than Puerto-Rico ($5 < 6$). If we do not intend for relationships to exist, it is better to One-Hot-Encode to prevent the implicit relationship from forming. However, a well-placed mapping can provide a lot of information. In our case, education has a relationship where Doctorate > Master's > Bachelor's > HS-grad. Carefully placing One-Hot-Encoders and Label Encoders for categorical features is an important task that requires meticulous and careful planning.
  2) Research has already been encoded to have values of 0 or 1. The target variable will be binarized to have values of 0 or 1 with the threshold set to 0.7. Where chance of admission greater than or equal to 0.7 will have value of 1 and chance of admission less than 0.7 will have a value of 0.

Ray M. Liu
September 5, 2019
CS 7641: Machine Learning
Dr. Isbell

Data Exploration:



Income Dataset



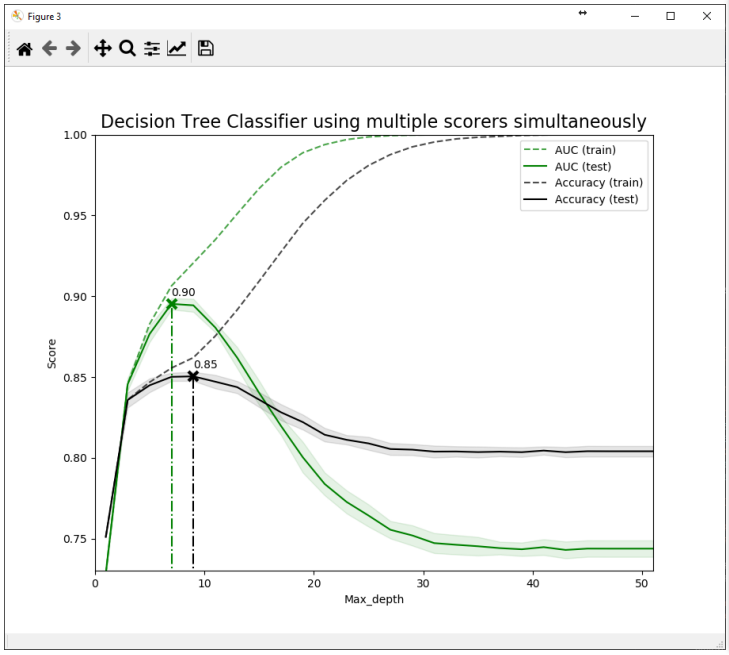
Grad Admissions Dataset

According to the plot for the income dataset, the top five features are 2, 0, 3, 9, and 6. These correspond to: fnlwgt, age, education-num, capital-gain, and relationship. At first the findings were surprising because one would assume hours-per-week has a huge impact on the income but the data says that hours-per-week (feature 11) comes in at rank 6 for importance. After careful analysis of the problem, hours-per-week has very minimal impact on the target variable because it is the hourly rate combined with hours-per-week that will decide one's income and not only the hours-per-week. As suspected, the three weakest features are 12, 7, and 8. These correspond to native-country, race, and sex.

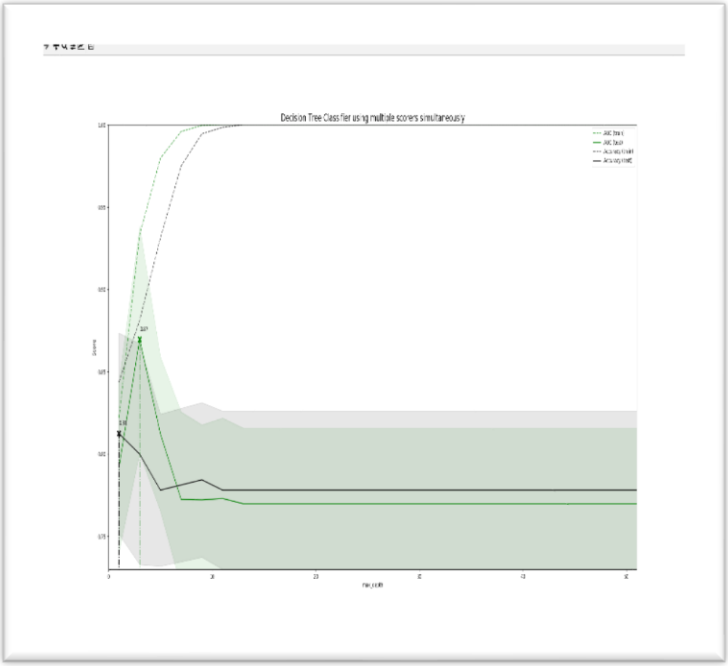
According to the plot for the graduate admissions dataset, the top three features are 5, 0, and 1. These correspond to the CGPA, GRE score, and the TOEFL score. The three weakest features are 4, 3, and 6: these correspond to the LOR, SOP, and research experience. The data obtained for feature importance has validity because graduate schools care more about whether a student will succeed in their program and for that to happen, they need to be academically well equipped.

Decision Trees: Decision Trees are one of the few intuitive supervised learning algorithms for new learners. They take an approach similar to humans for solving problems, where a user would ask a question to better narrow down their answer(s). On a high level, the user asks an initial question which narrows the solution down, such as "is the solution an object", until the search space has been narrowed down to the solution. Decision Trees are very powerful, simple, and provides a lot of interpreting power, but they also have shortcomings. One of which is overfitting. Overfitting occurs when too many specific questions are asked by the model and the model has a generalization issue. To prevent this, Pre and Post pruning is used to remove tree nodes that are unnecessary or provide little benefit for the model.

Pruning: Pruning was implemented for the Decision Tree Classifier to reduce overfitting of the training data; the pruning method used here was max-depth. Max depth was one of the easier ones to implement while providing a lot of power in preventing overfitting because one of the main reasons for overfitting is attributed to large depth sizes. Therefore, Max Depth pruning is one of the best options for preventing overfitting. For the sake of brevity of this course, the simpler method with good results was implemented. The more complex version, as mentioned in citation [1], combined with max depth pruning may yield better results.



Income Dataset



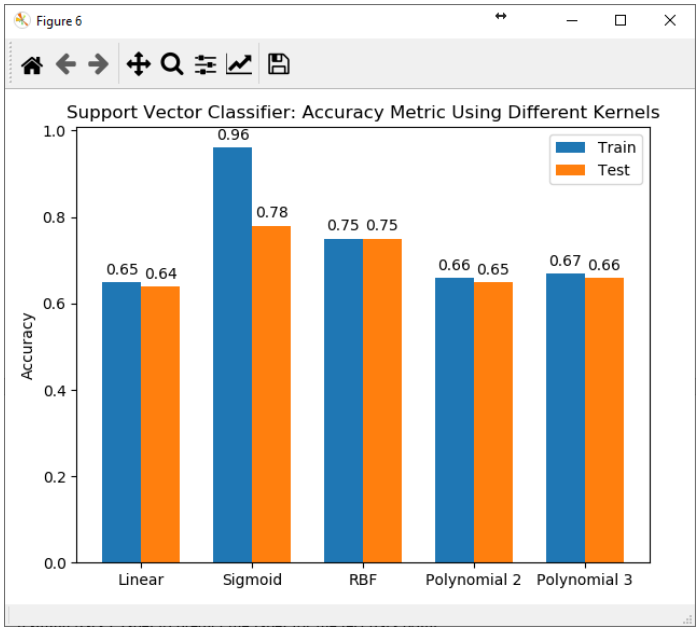
Grad Admissions Dataset

Results: In the plot above, it is evident that pruning the max depth has promising results. Having a depth of 8 provides the best training and test score trade-off for the income dataset whereas the graduate admissions dataset performs better with a max depth of 3. Having a depth more than 8 for the income dataset and more than 3 for the graduate admissions dataset causes the model to overfit as seen in the plot above. For example, having a depth of 20 results in a training set accuracy of 0.98 whereas the test set accuracy is 0.80. Given the model obtained a test set accuracy of 0.85 before, the accuracy went down by 0.05. This is an issue because the goal of the Decision Tree Classifier is to better generalize on future unseen data and not to memorize the training set. Therefore, for this problem the optimal depth will be 8 for the income dataset and 3 for the graduate admissions dataset as obtained from pruning the max depth of the Decision Tree.

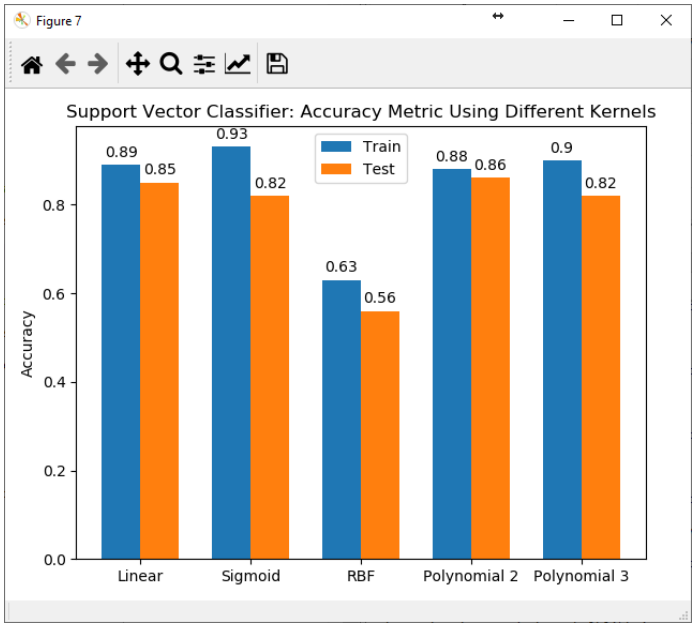
Support Vector Machines: The next supervised algorithm will be the Support Vector Classifier. SVC's use kernels to fit the data and tunes itself to find the right parameters to best fit the training data. The SVC will be tuning its own parameters using the specific kernel function to fit our training data, so the hyperparameters used for this problem will be the kernels. The sklearn library has four specified kernels for SVC's: radial basis function kernel (RBF), sigmoid kernel, linear kernel, and polynomial kernel. For this problem, all four kernels will be used and for the polynomial kernel we will be using degrees 2 and 3. Due to the size of the income dataset (n = 30,000), training time takes hours. However, using the graduate admissions dataset (n = 401), the training time only took a couple of minutes.

Ray M. Liu
September 5, 2019
CS 7641: Machine Learning
Dr. Isbell

Max Iteration = 10,000

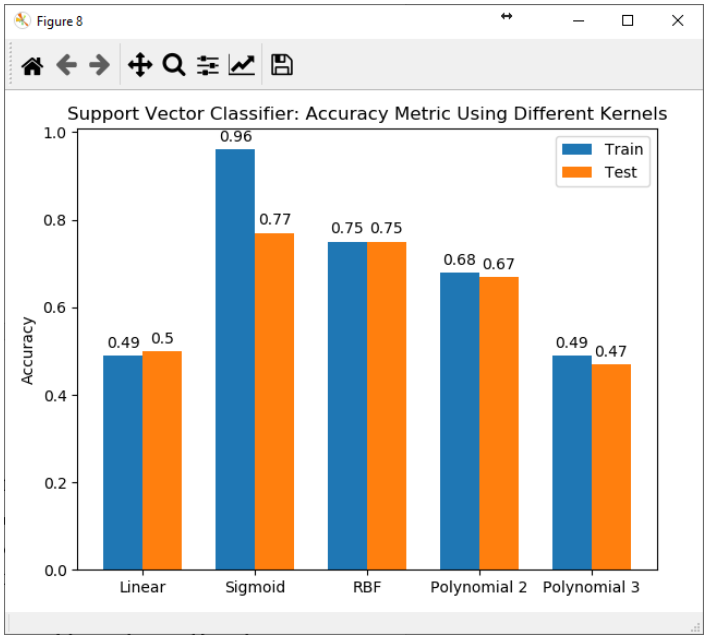


Income Dataset

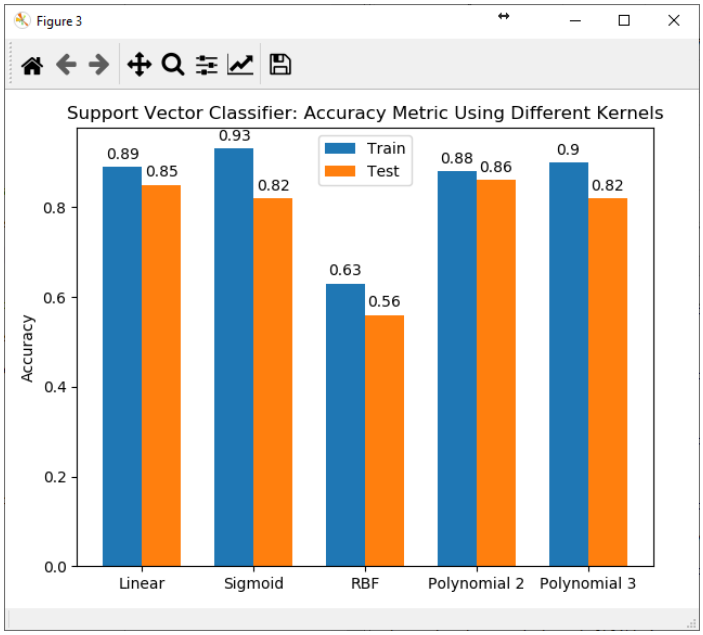


Grad Admissions Dataset

Max Iteration = 100,000



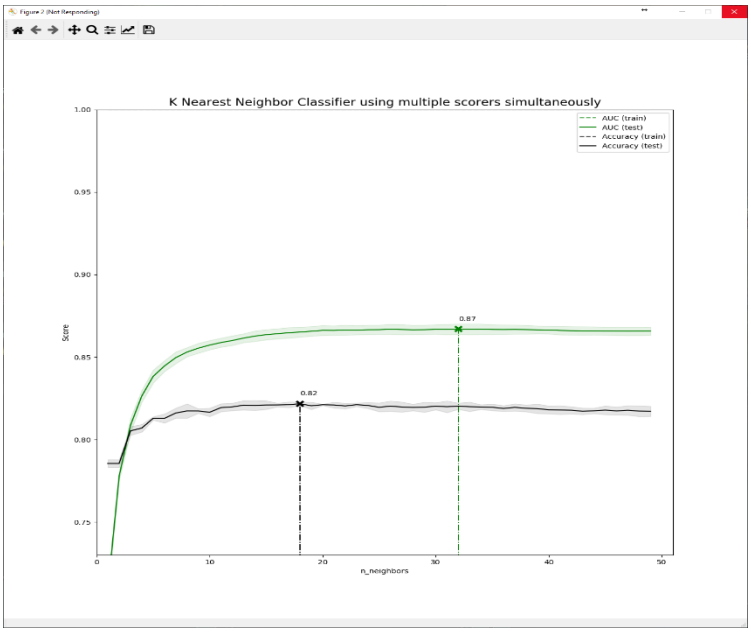
Income Dataset



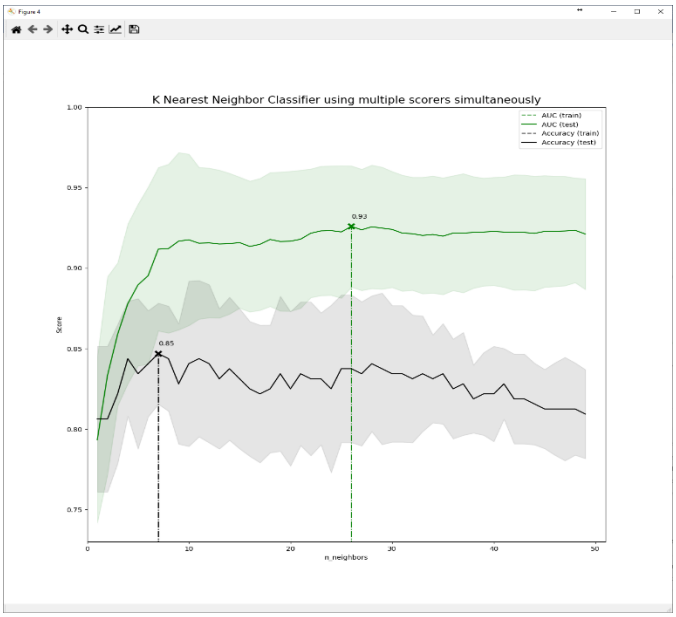
Grad Admissions Dataset

Results: Every problem has a different optimal model and usually, different kernels fit poorly and not as well. But for this problem, we have about similar accuracies with different kernels. However, we can see that sigmoid has a really large accuracy for the training set and a test accuracy that's around the same as the other kernels. This was trained using a max iteration of 10,000 and as the number of maximum iterations goes up, the accuracy will fluctuate and the results will be clearer. SVM's are really good baseline models given how quickly it takes to train them for decent results. With a longer training time, the SVM algorithm will fit the data better and show us which kernels are truly better for the problem. As shown in the plots above, a max iteration of 100,000 shows that a linear and polynomial with degree 3 are not as great for the income dataset in comparison to the sigmoid, RBF, and polynomial with degree 2 kernels. However, we can see that the graduate admissions dataset has similar results for a max iteration of 10,000 and 100,000. This means that the SVC has reached the optimal result at 10,000 iterations and training more would not improve the model.

K-Nearest Neighbors: K-Nearest Neighbors is a powerful non-parametric supervised algorithm that provides great accuracy but can be very memory intensive. The algorithm has many perks, including but not limited to, fast train time (one of the fastest algorithms to train), robust to outliers given a large enough hyperparameter K, and is very simple to interpret. However, with great perks comes great responsibility; the time and space complexity will grow with the size of the dataset because K-NN stores the training dataset in memory or a database; unlike parametric models which store model parameters. The algorithm works as follows: the model stores the training data in memory, hard drive, or a database to lookup or retrieve when needed and when the time comes to classify a new example, K-NN looks at all the training data and retrieves the K training samples that are most similar to the prediction data point using a provided distance metric and finally weighing the obtained training data's label to predict the label for the prediction data's point.



Income Dataset

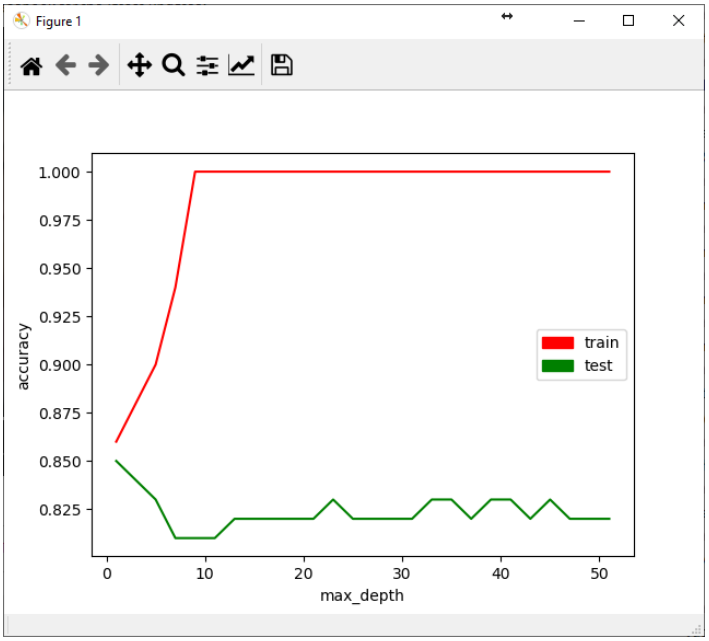


Grad Admissions Dataset

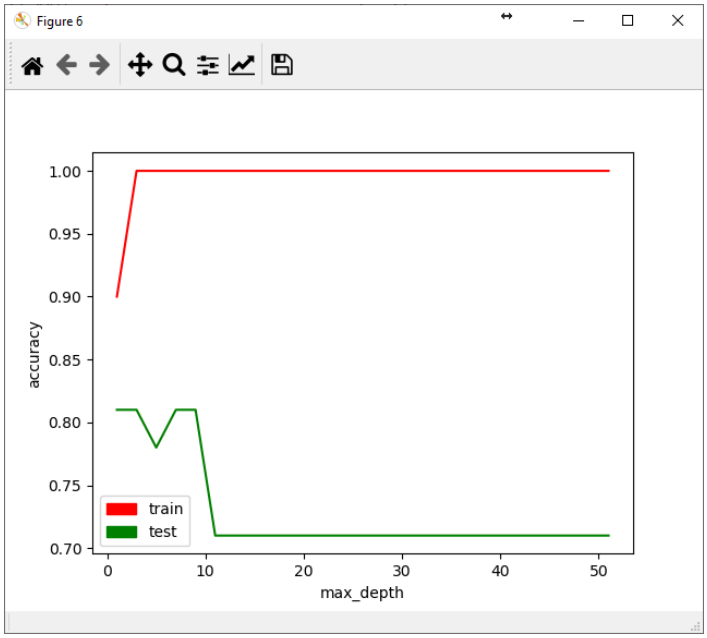
Results: The results from the plot show that the number of neighbors doesn't matter after K = 15 for the income dataset. However, for the graduate admissions dataset, the plot shows that a large value for K would cause the accuracy to go down for the test set. This happens because the model is taking more points than necessary and those points provide harmful information for the model. The training and test accuracy used the same data yet the accuracy never hit 1.0. This means that the training data has outliers causing the voting for K-NN to have errors.

Adaptive Boosting (AdaBoost): A renowned ensemble learning technique, Adaboost, is the next algorithm to implement and model the dataset. Adaboost uses an ensemble of decision tree classifiers to model the data and follow human intuition for learning; if the student knows one topic really well but struggles in another topic, the student will put more effort into learning where they struggle and less effort into what they already know. The hyperparameters for this problem are the number of estimators and the maximum depth, as they are simple yet powerful for solving the problem of overfitting.

Pruning: Like the Decision Tree Classifier from earlier, the pruning method used here will be the maximum depth. As this assignment is focused more on the brevity of algorithms than the complexity, using maximum depth for pruning has simplicity yet has a huge impact.



Income Dataset

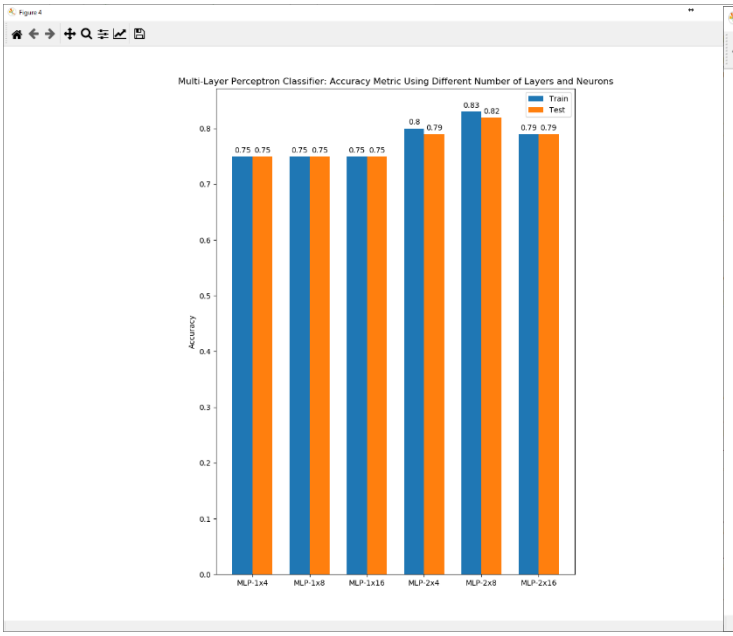


Grad Admissions Dataset

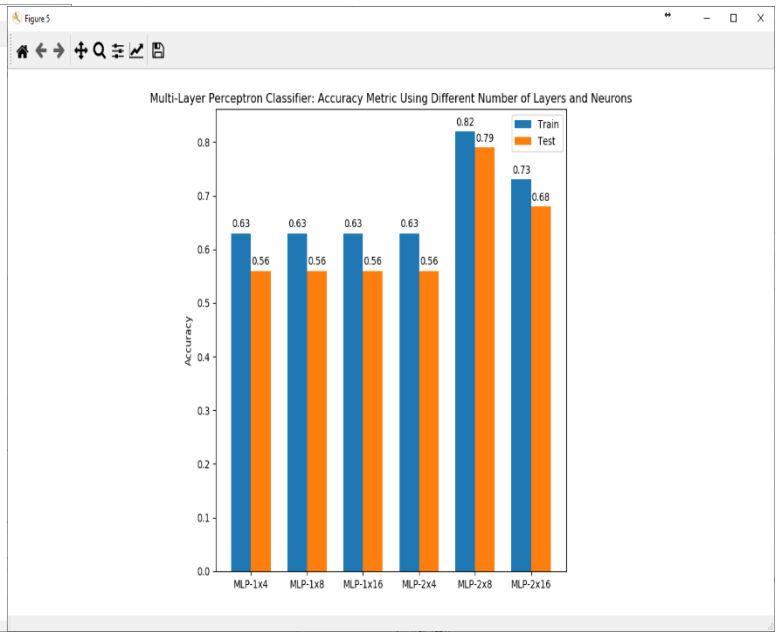
Results: The default number of estimators (50) provided by sklearn serves as a great choice for the problem. Using pruning to determine the best maximum depth for the problem will help prevent overfitting. In the plot above, it is evident that without pruning, the model will overfit and generalize poorly. With pruning, the plot says that a max depth of 1 is the best fit for the income and graduate admissions dataset. And it makes sense intuitively because the definition of boosting algorithms are that they use a depth of 1 to obtain a weak classifier which will be built upon later.

Neural Networks: Neural Networks are one of the most powerful algorithms with infinite possibilities. Many State-of-the-Art models are achieved using deep neural networks. With convolutional neural networks dominating the image recognition field and recurrent neural

networks dominating the natural language processing field, such as attention mechanisms, transformer XL, word embeddings, BERT, word2vec, LSTM's, etc. For this problem, I will be using a simple MLP from the sklearn library as simplicity and brevity is the core of this assignment. Sklearn provides easy and fast to implement code for very simple neural network architectures, but for more complex models Keras and TensorFlow provides users with the simplicity of creating a large network and access to a large toolbox such as gradient clipping, custom layers, pretrained word embeddings, and pretrained models.



Income Dataset



Grad Admissions Dataset

Results: The hyperparameters used for this problem: Stochastic Gradient Descent, Adaptive Learning Rate, batch size of 32, data shuffling, and early stopping. The plot above shows that using a Multi-Layer Perceptron with 2 layers and 8 neurons each yields a 0.83 accuracy for the training set and a 0.82 accuracy for the test set. Using a simple neural network architecture to model the data, I was able to achieve a 0.83 and 0.82 accuracy. For the simplicity of this assignment, I decided to use a small neural network and nothing extravagant. But, using residual networks, deep neural networks, dropout, batch normalization, a fine-tuned learning rate and number of layers and nodes, will require in-depth analysis which is beyond the scope of this assignment. However, using a simple MLP shows that more layers is not necessarily a bad hyper

parameter for this problem as it shows in the plot that the data is rather complex. The parameter early stopping was used to prevent overfitting by stopped the model early when the validation accuracy started to decrease rapidly which performed exceptionally well because the difference between the training and test accuracy does not differ significantly.

Conclusion: The final winning models for both datasets are K-Nearest Neighbors and SVM with SVM being the winner by only a slight margin. The results aren't surprising because SVM's are common for baseline models due to how simple they are to train while providing fabulous results. Near the end of the assignment, I came to learn about the imbalance of datasets. One thing I wish I did was compute the confusion matrix to measure the imbalance of the dataset and whether any of the models randomly guessed a target class to obtain a high accuracy. Therefore, I wanted to dive deeper into using the f1 score as a metric. Another thing I wish to do is spend more time cleaning the data because both datasets contain outliers. Having great data that has been cleaned really well will provide higher accuracy even when using a simple model. Therefore, spending more time to better prepare and clean the data for the model will yield high accuracies. However, preparing the data, preprocessing, and feature engineering are all beyond the scope of this course. In conclusion, the assignment was fun and I learned a lot about the five supervised learning algorithms: Decision Tree Classifiers, Support Vector Machines, Neural Networks, K-Nearest Neighbors, and Tree Boosting Algorithms. While implementing the supervised learning algorithms, the biggest takeaways for me were tree pruning methods, how to plot models and visualizing the results, the benefits of feature importance, and last but not least, the simplicity and importance of pruning for Decision Trees.

Citations and Resources Used:

[0] https://scikit-learn.org/stable/auto_examples/ensemble/plot_forest_importances.html

[1] https://stackoverflow.com/questions/49428469/pruning-decision-trees

[2] https://scikit-learn.org/stable/auto_examples/model_selection/plot_multi_metric_evaluation.html#sphx-glr-auto-examples-model-selection-plot-multi-metric-evaluation-py

[3] https://matplotlib.org/3.1.1/gallery/lines_bars_and_markers/barchart.html#sphx-glr-gallery-lines-bars-and-markers-barchart-py

[4] https://matplotlib.org/tutorials/introductory/pyplot.html

[5] https://matplotlib.org/3.1.1/tutorials/intermediate/legend_guide.html