4-22-2022

Machine Learning Applications using SciKit-Learn and TensorFlow

Trevor Palmatier
Hope College

Kenneth Munyuza
Hope College

Machine Learning Applications *using SciKit-Learn and TensorFlow*

For more information, contact:
Dr. O. Olagbemi
141 East 12th Street, Holland, MI
olagbemi@hope.edu

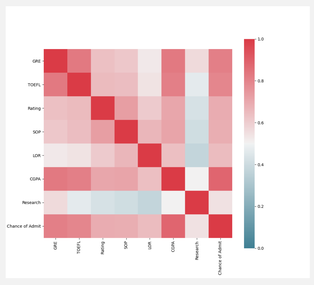
Trevor Palmatier and Kenneth Munyuza (with Dr. Omofolakunmi Olagbemi - Advisor)
Hope College, Holland, Michigan

Introduction

Our goal was to explore different applications of machine learning (ML) and to develop a good understanding of the processes required for the creation of effective ML models. Using SciKit-Learn for traditional ML and TensorFlow for neural networks, we worked on two major ML tasks: Regression and Classification modeling. We then applied that knowledge in constructing a neural network to predict ground reaction forces for biomechanics-related research, utilizing motion capture data from sensors and force plates.

Regression

Regression problems attempt to predict a value based on the input values. We attempted to predict the probability of acceptance into graduate school. To start experimenting, we looked at the Pearson correlation efficient to find the most important features and to find any collinearity in the data. This showed that CGPA, GRE and TOEFL were the most significant predictors of the chance for admission and that there was no notable collinearity in the data. After this verification process, we trained five different models. Each model's performance was measured by three metrics: the R^2 score, mean squared error (MSE), and root mean squared error (RMSE).

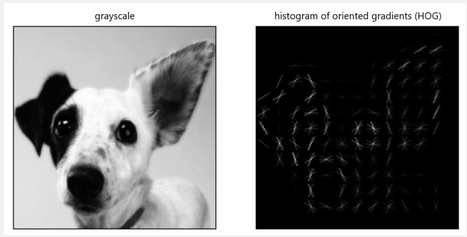


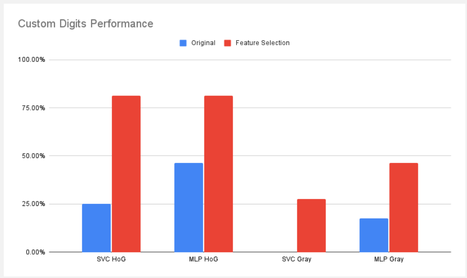
To improve performance, we tuned the input parameters (hyperparameters) for the SVR and Random Forest Regressor models. This involved determining a reasonable range of values for what the hyperparameters should be. This is was a trial and error process to see what combinations gave the best scores. Although there is still room for slight improvements, we found that all of these models performed the regression well, except Decision Tree, which lagged behind the others even after tuning.

Original Performance					Performance with hyperparameter tuned				
Model Name	**R2**	**MSE**	**RMSE**	**Seconds Elapsed**		**R2**	**MSE**	**RMSE**	**Seconds Elapsed**
Linear Regression	0.8108	0.003974	0.05946	1.172		0.8108	0.003971	0.05946	1.093
SGD Regression	0.8081	0.003974	0.05972	0.484		0.8060	0.003997	0.06002	0.469
Random Forest Regressor	0.7819	0.004497	0.06389	0.281		0.7983	0.004255	0.06136	0.922
Decision Tree Regressor	0.5839	0.007789	0.08736	0.047		0.5772	0.007928	0.08818	0.031
SVR	0.6852	0.006248	0.07666	>0.001		0.8154	0.003947	0.05885	0.485

Classification

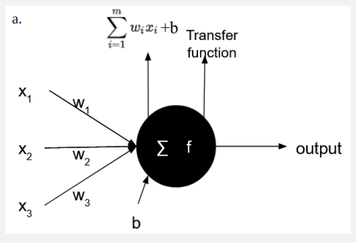
For classification, in addition to working with several other datasets, we made our own dataset of handwritten digits, mirroring the classic MNIST dataset. After we scanned in the digits and formatted the data as 80x80 grayscale images to run through a model, we got terrible results, with our best accuracy being 22.50%. We therefore implemented additional preprocessing. By converting the images to histograms of oriented gradients (HOG), which only retain information where surrounding pixels vary by a certain amount, accuracy improved significantly. Then, using feature selection to remove the pixels with low variance across samples, that primarily being white space in the corners and edges, accuracy reached up to 81.05%, only 10% lower than our accuracy on the much larger and cleanly formatted MNIST dataset.

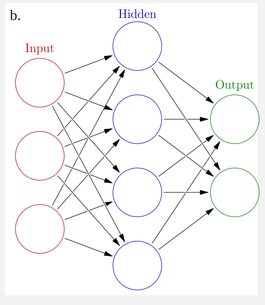




Neural Networks

Neural Network are complex learning models that can be appropriately scaled to the complexity of the data. They are made up of many perceptrons, which have inputs that are passed into a threshold function to determine whether any should be turned on or off. When layering many of these perceptrons together, the resulting network can learn complex patterns in data with ease.





Perceptron (a) and basic artificial neural network (b)

The connection between each pair of perceptrons, also called neurons or nodes, has a weight that determines its contribution to the output. The network is trained through backpropagation by altering each weight to minimize error in the output. After many iteration of this backpropagation step, the network will attain optimal weight values, giving corresponding output values for the model. Our experiments with neural networks led us to image recognition of Arabic characters (with 28 classes). We tested this same dataset with traditional machine learning models which performed poorly. Applying an artificial neural network only improved the performance marginally - from 61.18% to 63.35%. To improve this, we tried a more complex model – a convolutional neural network – which performed significantly better, with a prediction accuracy of 96.22%.

Accuracy values of ML models on Arabic Characters dataset

Model Name	**Accuracy (%)**
Random Forest Classifier	61.18%
Artificial Neural Network	63.35%
Convolutional Neural Network	96.22%

Examples of six different characters in the dataset

Neural Networks with Biomechanics

We also collaborated with a research group led by Dr. Brooke Odle to develop a deep neural network (DNN) to predict ground reaction forces (GRFs) when performing a variety of tasks. In a lab setting, these forces are measured using force plates. With our DNN, they can be predicted outside of the lab environment when relevant data from inertial measuring units (IMUs) are provided as inputs. Training data for our network was collected from two participants. Ten motions, comprised of squatting, lifting, and leaning tasks, were performed in ten trials, with ten repetitions per trial. The data was run through a low pass filter to remove noise and was split into the individual repetitions. 50,000 samples were extracted from each participant's data (100,000 total). Each sample had 49 IMU readings (DNN inputs) and 10 GRFs (DNN outputs). All the different data files were then combined into one dataset which was used to train a DNN.

DNN Architecture	**Model score (0 to 1)**	**Loss (error)**
2 layer ANN - *Huber*	0.8633	5.065
4 layer ANN - *MSE*	0.9143	67.335
4 layer ANN – *Huber*	0.9263	2.912

We developed a two-layer DNN with 98 and 49 neurons respectively in the layers. It achieved a model score of 0.8633. We then tried a DNN with four hidden layers with 196, 147, 98 and 49 neurons respectively, and an output layer of 10 neurons. With this architecture, we achieve a model score of 0.9263.

Future Work

As promising results were achieved, we plan to obtain data from more participants for the DNN to make it more robust and better able to generalize. We also plan to experiment with combinations of subsets of existing features in the DNN and evaluate the performances. Lastly, we plan to develop task-specific DNNs and compare results with the general DNN.

Acknowledgements

- Dean of Natural and Applied Sciences and Computer Science department for funding this research.
- Dr. Brooke Odle (Engineering) for the collaboration opportunity and data collection.
- Dr. Ryan McFall (Computer Science) for helpful suggestions at various points in the study.

References

Graduate Admissions Dataset:

- Mohan S Acharya, Asfia Armaan, Aneeta S Antony : A Comparison of Regression Models for Prediction of Graduate Admissions, IEEE International Conference on Computational Intelligence in Data Science 2019

Arabic Character Dataset:

- A. El-Sawy, M. Loey, and H. EL-Bakry, "Arabic handwritten characters recognition using convolutional neural network," WSEAS Transactions on Computer Research, vol. 5, pp. 11–19, 2017. https://doi.org/10.1007/978-3-319-48308-5_54 https://link.springer.com/chapter/10.1007/978-3-319-48308-5_54
- A. El-Sawy, H. EL-Bakry, and M. Loey, "CNN for handwritten arabic digits recognition based on lenet-5," in Proceedings of the International Conference on Advanced Intelligent Systems and Informatics 2016, vol. 533, pp. 566–575, Springer International Publishing, 2016. https://www.wseas.org/multimedia/journals/computerresearch/2017/a045818-075.php
- Loey, Mohamed, Ahmed El-Sawy, and Hazem El-Bakry. "Deep learning autoencoder approach for handwritten arabic digits recognition." arXiv preprint arXiv:1706.06720 (2017). https://arxiv.org/abs/1706.06720