Incremental Data Sets For Decision Trees in Python

Jeremy Swerdlow
Earlham College
801 National Rd. West
Richmond, Indiana 47374
jjswerd14@earlham.edu

December 12, 2017

1 Background and Motivation

Machine learning is one of the fastest growing areas of computer science currently. Massive leaps forward have let companies create smart software, such as Google Assistant or Apple's Siri, to help its users. Other companies have used it to help predict specific results.

The success of these technologies derives from a change of focus in the mid 1990s to data-driven machine learning, instead of rule driven.[2] By focusing on past cases of the data, and identifying patterns from within it, the models are able to predict outcomes based on new scenarios with which they are faced.

However, to have a successful model, a large amount of data is needed. Google processes petabytes of data each day, one of which is equal to a million gigabytes. In total, their warehouses store around 10-15 exabytes of data, each of which are equal to a million petabytes.[5] In total, Google is believed to have a million, million gigabytes of data stored.

While it is easy to increase the amount of space data can inhabit by adding more storage, when handling the massive amounts of data often required to create as sophisticated of a model as needed for a specific problem, the time to use all of the data can be much more expensive. Even with the parallelization provided by softwares such as Hadoop, which use MapReduce ideas to handle the massive amounts of data, calculations and the creation of a ML model such as an Artificial Neural Network (ANN) or a decision tree can take hours or days. This is exemplified because of the current construction of many models. Most implementations of machine learning algorithms can only be trained once; after their initial training, to train the model again requires the same data to be fed again.

Certain models have been further developed though, to increase time efficiency in ways other than just by parallelization. A feature recently added to modern Deep Neural Networks, incremental application of data allows for a single model to be reused when new data is added. Instead of requiring a full re-training of the model, and in many cases a new model in of itself, the deep neural networks allow data to be added to the existing model, greatly reducing the time it takes to update the model to contain the most recent, and often most relevant, data.

1.1 Aim Of The Project

With this idea in mind, the aim of this project is to adapt this behavior from neural networks to decision trees within Python. As one of the fastest growing languages in terms of use by companies, Python has become a staple in data science, especially thanks to the Jupyter Notebook.[1] These interactive work frames for Python allow a user to code and test in the same location, and display many graphs in-line with their cells, and are free to use.[7]

Decision trees and neural networks both support supervised learning methods, and with the belief their data structures are similar enough, the goal of this project is to use the new methods for incremental data addition available in neural networks to introduce the same capabilities to decision trees.

2 Related Work

As mentioned in the background, machine learning is a rapidly growing field, and so there is plenty of research using ANN's and decision trees. ANN's have been applied to a variety of problems, such as image classification and data mining, with varied success.[4][6]

Similarly, decision trees have been applied to a myriad of supervised learning problems, such as data mining for genetic algorithms.[3] They have become so popular, research has gone into optimizing the methods by changing hyperparameters to the models; one of the most popular methods currently in production is presented in XGBoost, and has had research done on its success.[12]

However, none of this research has been into increasing the time efficiency of the decision trees; instead, the focus has been on the accuracy of their results. This is where the research being conducted will extend the current capabilities of decision trees in Python.

3 Designing a Python Decision Tree with Incremental Data

To fully make use of the research and implementations of ANN's which allow models to be updated with data instead of needing to recreate them, this project will begin by reviewing source code for basic implementations of both ANN's and decision trees, before continuing with more commonly used versions. The sci-kit learn package of Python provides basic implementations of these two models.[9][8] For production versions, XGBoost as described earlier will be used for the decision trees, similarly TensorFlow, a python package, will be used for the neural networks.[11][10]

4 Timeline

To make the information on the timeline easily accessible, a Gantt chart of the dates on which each part of the process will be worked on has been included in Figure 1 of Appendix A. As can be seen, the work is distributed across the semester and across the tasks. I make use of the whole semester, attempting to overlap related concepts when important, but otherwise spread them out.

References

[1] L. Kim, "10 Most Popular Programming Languages Today," Inc.com, 01-Jun-2015. [Online]. Available: https://www.inc.com/larry-kim/10-most-popular-programming-languages-today.html. [Accessed: 12-Dec-2017].

[2] B. Marr, "A Short History of Machine Learning – Every Manager Should Read," Forbes. [Online]. Available: https://www.forbes.com/sites/bernardmarr/2016/02/19/a-short-history-of-machine-learning-every-manager-should-read/. [Accessed: 09-Dec-2017].

[3] Kenneth Sörensen and Gerrit K. Janssens, "Data mining with genetic algorithms on binary trees," European Journal of Operational Research, no. 151, pp. 253–264, 2003.

[4] Giorgio Giacinto and Fabio Roli, "Design of Effective Neural Network Ensembles for Image Classification Purposes." .

[5] C. Carson, "How Much Data Does Google Store?," Cirrus Insight, 10-Dec-2017. [Online]. Available: /blog/much-data-google-store. [Accessed: 11-Dec-2017].

[6] Dr. Yashal Singh and Alok Signh Chauhan, "Neural Networks in Data Mining," Journal of Theoretical and Applied Information Technology.

[7] "Project Jupyter." [Online]. Available: http://www.jupyter.org. [Accessed: 12-Dec-2017].

[8] scikit-learn, scikit-learn decision tree. .

[9] scikit-learn, scikit-learn neural network. .

[10] TensorFlow, TensorFlow. .

[11] DMLC, XGBoost. .

[12] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," arXiv:1603.02754 [cs], pp. 785–794, 2016.

Appendix A

Figure 1: Estimated Deadlines for Project