TensorFlow versus H20: Predicting the SP500

Kenneth E. Davis M.S.
kedavis@umail.iu.edu
Indianapolis, IN 46235

Abstract

Data science tools can help us both better understand underlying phenomenon and better identify future courses of action. However, selecting the right tool for the job requires a further challenge and an art. The present analysis compares the well known TensorFlow package, in Python, to the lesser utilized but more statistically friendly package H2O, in R, to predict the SP500 stock market index as a multi-parameter economic time series. Results, innovations, ease or difficulty of use, and future applications of each data science package are reported. Local machine and virtual machine instances in Amazon AWS AMIs are evaluated. Overall fit of economic parameters are examined. TensorFlow and H2O offer unique and powerful neural network and recurrent neural network/LSTM solution pathways that are well worth consideration.

Keywords: TensorFlow, H2O in R, Neural Network, LSTM, SP500, Python

1. Introduction

At the present moment, machine learning is increasingly being recognized for new, often stunning innovations, nearly on a daily basis. These advances occur in wide variety of fields, and include recommendation engines[1-3], image recognition[4], games of strategy[5], gambling[6,7], crime prediction[8-10], military strategy[11], radiology[12-14], neural fMRI mapping in the human brain[15], and drone recognition of healthy crops[16], among many others.

Although published advancements are very impressive, many new discoveries remain unreported, as they carry large degrees of secrecy and efficacy. In profit driven organizations, protection of proprietary methods and trade secrets relate directly to profitability in a competitive environment. Alternatively, military applications relate immediately to national defense and security which would be quickly exploited or rendered obsolete upon technique sharing. For equity investing, the efficacy of successful methods would erode upon exposure, and nullify the probability of gains in a game of investment survival often characterized by extreme competition. For these reasons, it is understandable that the reproducibility and duplication of many referenced findings is often difficult to achieve.

Despite these challenges, the current study attempts to illustrate the advantages and differences between two popular but disparate machine learning tools, available today at no cost, that can help answer difficult research questions. The topics addressed here include what machine learning platform to use, as well as what data science artificial intelligence techniques best help obtain competitive research solutions. The present study compares TensorFlow/Keras in Python, to H2O in R, for predicting the stock market's SP500 index.

2. Materials and Methods

The present analysis utilizes readily accessible economic data to predict the level of the stock market, using cost-free, GNU General Public License software packages as follows:

2.1 Software and OS

Analysis Software

Python 3.6.8 with Spyder 3.3.2
TensorFlow 1.6.0 (older version to insure cross compatibility with Amazon Cloud installations)

R Studio 1.456
R 3.5.0
H2O 3.18.0.11

Anaconda Navigator 1.9.6

Operating Systems

Windows 10 Pro with Core i7 8-Core CPU 3.4 GHz

Amazon Cloud AWS EC2 Windows 2016 Server
(Deep Learning Amazon Machine Image AMI, + various 4, 8, and 16 CPU's depending on task)

Mac OS Sierra 10.12.6 (with Python TensorFlow and H2O R installations used during travel)

2.2 Data Sources

a.) Economic data: FRED database (St. Louis Fed) [17]
b.) SP500 historical data from Yahoo Finance
c.) Quandl package for PMI data (https://www.quandl.com required a free registration at the time of study)

2.3 Variables Used

Outcome Variable: SP500 Index [1992 thru 2018]

Predictors: Economic Indicators

1. Unemployment Rate
2. Consumer Price Index (CPI)
3. Purchasing Managers Index (PMI)
4. lagged SP500

Monthly values.

2.4 Model Architecture

The primary architectures were:

TensorFlow/Keras in Python.
2- hidden layers with
200 nodes each,
5000 epochs.

H2O in R
5-hidden layers with
200 nodes each,
2000 epochs.

In TensorFlow/Keras in Python, a
Sequential Model was selected with
Relu Activation, and a
1-node output layer with Linear Activation.
Compilation utilized the
Adam Optimizer, with a
Loss Function of MSE.

In H2O in R,
The architecture identified above, with the loss function RMSE, and
all settings set to default (relu activation and adam optimization),
including random seeding.

3. Results

3.1 Best Model achieved (n=25 iterations)

Table 1: Best RMSE and R^2 by platform

	RMSE	R^2 (SSR/SST)
Keras TensorFlow in Python	**9.3**	0.9997
H20 in R	10.1	0.9997

3.2 Consistency of RMSE (Reproducibility, n=25)

Table 2: Average RMSE, standard deviation, min and max of n=25 identical runs

RSME	mean	**Std**	Min	max
Keras TensorFlow in Python	**13.6**	3.6	**9.3**	24.7
H20 in R	14.8	**2.5**	10.1	**19.8**

3.3 n=1 Out Of Sample Prediction (n=24 months^)

Table 3: Maximum Loss, percent residual, versus the buy and hold strategy 2016-2017, with a 1-month rolling test target. Training data was 1992 thru 2015, rolling forward monthly.

	RMSE	R2	**Max Loss**
TensorFlow/Keras in Python	**78.6**	**.8406**	-4.3%
H20 in R (2-layer gave better max loss)	99.4	.7451	-3.9%
Buy and Hold (worst month loss)			**-3.6%**

^sliding prediction window was 24 individual months, Jan 2016 thru Dec 2017

3.4 Cross Validation

Cross validation was performed ceremonially. In the case of out of sample prediction (as the forward SP500 level does not necessarily remain within in-sample value ranges) better results were acquired with higher epoch counts. The two scoring histories exampled below are from H2O in R (for cleaner looking default outputs):

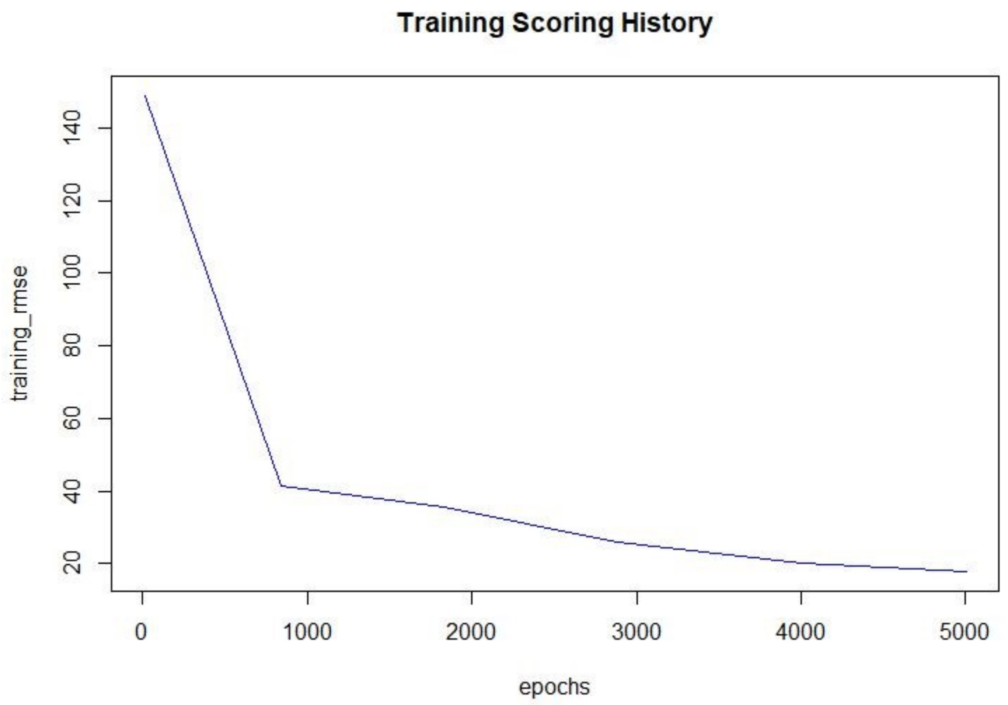

Figure 1: Scoring history by epoch number with H2O in R

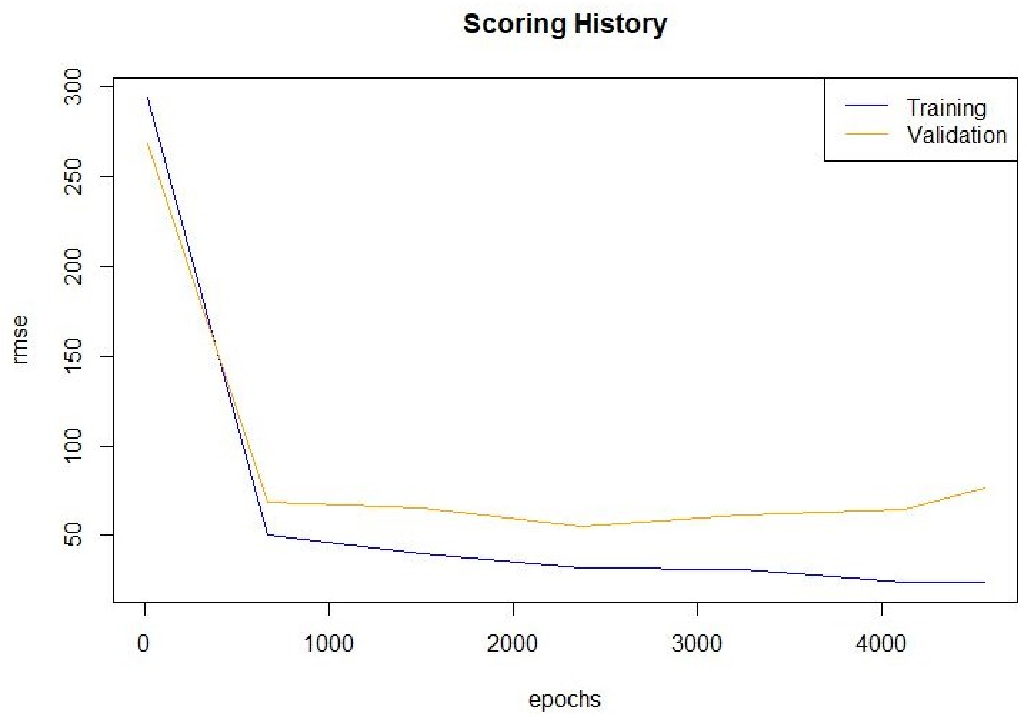

Figure 2: 5-fold cross validation with 90% training data to 10% validation data

3.5 Single Best Model, by package

Table 4: Traditional Methods, TensorFlow/Keras and H2O methods, compared. Best models. Empty cells were compared, but not optimized in favor of the better performing Neural Network.

		RMSE	R^2
individual methods[18]	Linear Regression/GLM	84	.9246
	AR(1)	51	.9911
	ARIMA	50	.9829
	NNTAR	47	.9907
TensorFlow/Keras in Python	Bayesian^ (2-Layer)	49.8	
	LSTM (2-period unit)	40.0	
	NN (Feed Forward)	9.3	.9997
H20 in R	GLM	>	
	Auto ML	-	
	Boltzmann machines	>	
	Deeply Randomized Tree	>	
	Deep Learning/NN	10.1	0.9997
	Stacked Ensemble	-	

^for Bayesian NN: n_samples =150 samples and n_iter=3500

3.6 2-Layer versus 5-Layer (n=25 iterations)

Table 5: 2-layer and 5-layer architecture (hidden layer count) compared including computational times

	Ave RMSE (std)[range]	epochs	time
Keras/TensorFlow Python (2-layer)	**13.6** (+/-3.6) [9.3, 24.7]	5000	28 min
Keras/TensorFlow Python (5-layer)	17.8 (+/-8.8) [10.0, 39.0]	2000	50 min

	Ave RMSE(std)[range]	epochs	time
H20 in R (2- layer)	19.2 (+/-1.8) [15.3 , 22.8]	5000	11 min
H20 in R (5-layer)	**14.8** (+/-2.5) [10.1 ,19.8]	2000	22 min

4. Discussion

4.1 Machine Learning versus Traditional Methods

Both TensorFlow/Keras in Python and H20 in R performed admirably in researching the outcome under study, the SP500 Index. Each technique successfully validated the other, and quite notably, neural net *substantially* outperformed traditional statistical methods, linear regression, ANOVA, AR(1), and ARIMA, by reporting RMSE's with minimum values of 9 and 10 (table 4, section 3.5), compared to values of between 50 and 90 (table 4, section 3.5) for the traditional methods[18].

4.2 Predicting the SP500

As for predicting the SP500, although each model performed well in general, neither method was yet able to shield users from the maximum losses compared to the buy and hold strategy (table 3, section 3.3). TensorFlow/Keras in Python reported a maximum monthly loss (largest losing residual) of -4.3% and H2O in R -3.9%, where buy and hold reported a maximum loss of -3.6%

4.3 Best Model: TensorFlow/Keras in Python

TensorFlow/Keras in Python produced the best model as measured by RMSE (table1, section 3.1). To the authors' best knowledge, this may be because the internal parameterization and methodology allows a wider range of possible outcomes compared to H2O in R. The trade off is with the range and standard deviation of multiple iterations, as discussed below.

4.4 The Most Consistent Model: H2O in R

H2O in R produced the most consistent results when running multiple passes of the identical model, as evidenced by a 30% to 50% or more reduction in the standard deviation and range for RMSE as seen in table 2, section 3.2, and table 5, section 3.6.

This did help when ruling out models along the research process, where extreme models occurring at the extended range may misrepresent a particular architecture (to be excluded mistakenly). This property however likely ultimately causes the best model to flow from TensorFlow/Keras in Python.

4.5 Ease of Learning: H2O in R

H2O in R was characterized by a very rapid rate of acquiring strong models quickly in the human learning phase. The time from example to actionable model occurred within minutes or hours, as opposed to hours or days in TensorFlow/Keras in Python.

This is primarily due to considerably less time spent on data manipulation and formatting in H2O in R. Python involves the use of a wider variety of libraries and packages, and the peculiarities involved with each. For H2O in R, once the data is the H2O environment, the analysis proceeds quickly.

A good example of this is scaling and normalization. In TensorFlow/Keras in Python, the user must customarily scale and rename the data between numpy arrays and panda dataframes in order to get a sensible deep learning result, where in H2O, scaling is automatically performed by default.

4.6 Speed: H2O in R

Processing times using 8 to 16 current cores were often 2 to 3 times faster with H2O in R (table 5, section 3.6).

However, if you have vast server and/or cloud level resources, this speed difference would be nullified.

4.7 Community Support: TensorFlow/Keras in Python

Answers were easier to find for TensorFlow/Keras in Python, and also more numerous. This was good and bad, more answers had to be considered, but there were more opportunities for solutions.

Stack Overflow and other online resources favored TensorFlow and Python, however all required solutions under study were readily available for both platforms. In both platforms, current authors were never 'stuck' for more than a matter of minutes, or on rare very occasion, hours.

4.8 Customization, Upscalability, Flexibility: TensorFlow/Keras in Python

Creation of custom loss functions and a multitude of specific tools greatly favor TensorFlow in Python. The engineering focus and coding compatibility (C++ etc) is apparent in TensorFlow/Keras and Python. The cost of this is only difficulty and complexity, but the advantages are numerous.

4.9 Coding Ease: H2O in R

Along with auto-scaling, issues related to sub-setting and referencing values favors H2O in R.
For the Python coder, using .iloc /.loc, indexing from 0 – 99 (for example) instead of 1-100, and inclusive versus exclusive ranges of data in loops add a layer of coding complexity avoided in R. This is particularly apparent with LSTM methods, where indexing reaches a crescendo.

4.10 Available Data Science Methods: Tied, favoring TensorFlow/Keras

The newest and most awe inspiring methods generally come from TensorFlow/Keras in Python. Convolutional networks and multiple methodologies involving neural networks (e.g. the policy network and monte carlo tree search of AlphaGo[5]) are much more numerous coming from DeepMind/Google's phenomenal work in this field.

However, the investigational research focus of H2O in R has many salient advantages as well. AutoML became available several months (roughly a year or so) prior in H2O rather than TensorFlow/Keras, and stacked ensembles and various other informative methods are much more readily available within the H2O framework with no additional package or data configuration.

Both platforms however provide interesting methods in any regard. Large, computationally intense and immediately implementable projects greatly favor TensorFlow/Keras in Python, while deep, methodical research and the understanding of variable relationships appears to favor H2O in R.

4.11 Study Weaknesses

The weaknesses of the present study include the following:

GAM models were not included, mostly due to time constraint. Further, new platform innovations of 2019 were not updated along the duration of this study (for consistency's sake).

PyTorch/Facebook and Azure/Microsoft were not yet reviewed, only due to learning time constraints.

A further in-depth, rigorous optimization of Bayesian and LSTM techniques were not pursued in favor of feed forward neural network, which performed very clearly and quickly at a higher level for the present study question. Lastly, 'solving' the SP500 was not strictly emphasized here in favor of the evaluation of analysis software.

4.12 Final Summary

The choice of machine learning tools largely depends on what is most important to the user, time, resource availability, flexibility and downstream application of the process.

The current authors continue to utilize both platforms and software packages presently: H2O in R for a speedy research and exploration phase, and then TensorFlow/Keras in Python for the customizable, powerful feature set going forward.

Both tools are key in performing, and enjoying, a truly great era in research, scientific understanding, and technological advancement.

Kenneth E. Davis, M.S.
kedavis@umail.iu.edu
September 2019

5. References

1. Y. Zhou et al., "Large-Scale Parallel Collaborative Filtering for the Netflix Prize," Proc. 4th Int'l Conf. Algorithmic Aspects in Information and Management, LNCS 5034, Springer, 2008, pp. 337-348.

2. Carlos A. Gomez-Uribe and Neil Hunt. 2015. The Netflix recommender system: Algorithms, business value, and innovation. ACM Trans. Manage. Inf. Syst. 6, 4, Article 13 (December 2015), 19 pages. DOI: http://dx.doi.org/10.1145/2843948

3. G.D. Linden, J.A. Jacobi, and E.A. Benson, Collaborative Recommendations Using Item-to-Item Similarity Mappings, US Patent 6,266,649 (to Amazon.com), Patent and Trademark Office, Washington, D.C., 2001.

4. K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In Proceedings of CVPR, pages 770–778, 2016.

5. Silver D, Huang A, Maddison C J, et al: Mastering the game of Go with deep neural networks and tree search. Nature 529(7587), 484–489 (2016)

6. Noam Brown and Tuomas Sandholm. Baby Tartanian8: Winning agent from the 2016 annual computer poker competition. In Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-16), pages 4238–4239, 2016.

7. Michael Bowling, Neil Burch, Michael Johanson, and Oskari Tammelin. Heads-up limit hold'em poker is solved. Science, 347(6218):145–149, January 2015.

8. Ferguson, A.G. "Big Data and Predictive Reasonable Suspicion." University of Pennsylvania Law Review 163(2): 339 – 410. 2015.

9. V. Grover, R. Adderley, and M. Bramer, "Review of current crime prediction techniques," in Applications and Innovations in Intelligent Systems XIV, ed: Springer, 2007, pp. 233-237.

10. Wang X., Gerber M.S., Brown D.E. Automatic Crime Prediction Using Events Extracted from Twitter Posts. In: Yang S.J., Greenberg A.M., Endsley M. (eds) Social Computing, Behavioral - Cultural Modeling and Prediction. SBP 2012. Lecture Notes in Computer Science, vol 7227. Springer, Berlin, Heidelberg. 2012.

11. The Economist. Artificial intelligence is changing every aspect of war. Battle algorithm. A new type of arms race could be on the cards. The Economist, September 7, 2019.

12. Ursula Schmidt-Erfurth, Amir Sadeghipour, Bianca S Gerendas, Sebastian M Waldstein, and Hrvoje Bogunovic. Artificial intelligence in retina. Prog Retin Eye Res. 2018 Nov; 67:1-29.

13. Bluemke DA. Radiology in 2018: are you working with AI or being replaced by AI? Radiology 2018 ; 287(2):365–366.

14. Yi PH, Hui FK, Ting DSW. Artificial intelligence and radiology: collaboration is key. J Am Coll Radiol. 2018. https://doi.org/10.1016/j.jacr.2017.12.037

15. Khazaee, A., Ebrahimzadeh, A., Babajani-Feremi, A. Application of advanced machine learning methods on resting-state fMRI network for identification of mild cognitive impairment and Alzheimer's disease. Brain Imaging Behav. 10 (3), 799–817. 2016.

16. E. Puig et al., "Assessment of crop insect damage using unmanned aerial systems: a machine learning approach," in Proc. of 21th Int. Congress on Modelling and Simulation, pp. 1420– 1426 (2015).

17. FRED Economic Data. Federal Reserve Bank of St. Louis, One Federal Reserve Bank Plaza, St. Louis, MO 63102. https://fred.stlouisfed.org/

18. Davis KE. Advantageous Statistical Tools for Stock Market Investing. 2018. In JSM Proceedings, Statistical Computing Section. Alexandria, VA: American Statistical Association.