MMF Data science project

Jian Wang, Tianyi Long, Tianran Li

October 2020

1 Introduction

In May 2017, the artificial intelligence " Alphago" defeated Kejie, and became the first computer program that defeat a professional Go world champ. Till now, machine learning techinques have been deeply rooted in our every day's life, such as recommendation when we are reading news and handwriting recognition when we are using our cell-phones . It has been nearly 40 years since machine lenaring was called an independent direction in 1980, and after generations of efforts, a large number of classic methods have been born.

However, even though machine learning are successful, these successful applications of machine learning are very far from automated. The reason behind is that human still need to do feature engineering, model selection, and algorithm selection carefully by themselves given that no algorithm or models could perform equally well on all learning problems. Another important reason is that some machine learning methods such as non-linear SVMS crucially rely on hyper parameter optimization.

Thus, automated machine learning(Auto ML), has been developed to take the human out of these machine learning applications. An informal and intuitive description of Auto ML can be expressed as:

$$\max_{configurations} \text{performance of learning tools}$$
$$\text{s.t.} \begin{cases} \text{limited(or no) human assistance} \\ \text{limited computational budget} \end{cases} \tag{1}$$

In general, we want the machine to do one of the following task:
- feature engineering, model selection, and algorithm selection.
- automate the optimizer and searching methods.
- have a evaluation process for each method.

In this project, we are going to deep dive into Auto-sklearn, and we will compare the performance of the Auto Scikit-learn with traditional deep learning methods on a stock price prediction task, and summary on their underlying techniques.

2 Scikit-learn

2.1 introduction to sklearn

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. Also, it is distributed under many Linux distributions, encouraging academic and commercial use.

The library is built upon the SciPy (Scientific Python) and this stack that includes:

- **NumPy**: Base n-dimensional array package
- **SciPy**: Fundamental library for scientific computing
- **Matplotlib**: Comprehensive 2D/3D plotting

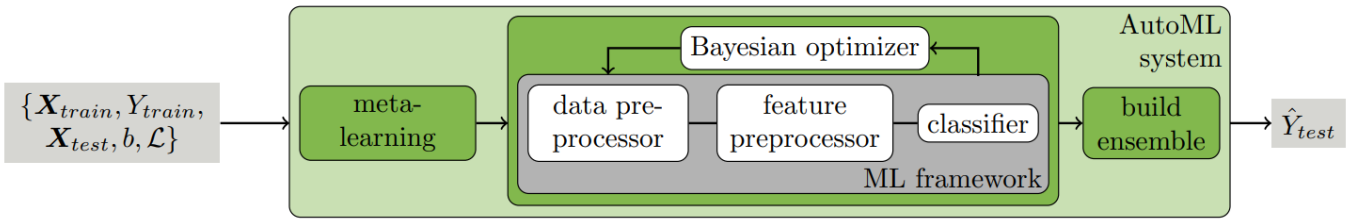
• **IPython**: Enhanced interactive console
• **Sympy**: Symbolic mathematics
• **Pandas**: Data structures and analysis

Extensions or modules for SciPy are conventionally named SciKits. Similarly, scikit-learn provides learning algorithms.

2.2 Fundation of Auto-sklearn

2.2.1 The basic algorithm for Auto-sklearn

Besides the tradtional model-selection techniques, the Auto-Sklearn add two more features: meta-learning and ensemble analysis, and we will discuss these two steps in detail. the general system Auto-Sklearn used can be generalized as below:



L is the loss function for each model, X_{train} is the training set, X_{test} is the test set for input, and Y_{train} is our training set for output.

The traditional ML frameworks needs to choose the best hyperparameter for each model, and then choose the best model to fit the data, where hyperparameter is a parameter whose value is used to control the learning process. Traditional methods to auto-select these hyperparameter includes: grid search, random search, and Bayesian optimization. Bayesian optimization is nowadays the core for any AutoML system since it's the most efficient way to select the hyperparamater.

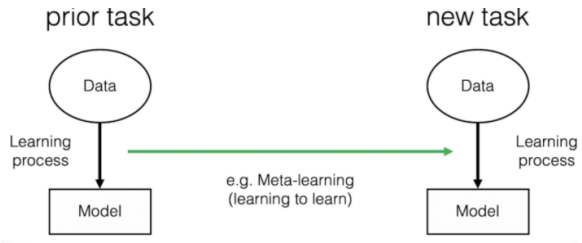
2.2.2 Parameter selection - Bayesian optimization

for any ML algorithm, different hyperparameter will give us different result on the final model.In a nutshell, Bayesian optimization fits a probabilistic model to capture the relationship between hyperparameter settings and their measured performance, and it then use this model to select the most promising hyperparameter setting(trading off exploration of new parts of the space vs exploitation in known good regions), and then evaluates that hyperparameter setting, updates the model with the result, and iterates again.

While Bayesian optimization based on Gaussian process models, it performs best in low-dimensional problems with numerical hyperparameters. As a result, the Auto-sklearn choose **SMAC**: A tree-based Bayesian optimization methods, to select the best hyperparameters.

2.2.3 Model selection - Meta-learning and automated ensemble construction

Meta-learning



The intuition for Meta-learning is : Learn about the performance of machine learning algorithms. The meta-laerning mimics this strategy by selecting the best machine learning framework that are likely to perform well on a new data sets. More precisely, the meta-learning approach for Autosklearn works as follows:

For each machine-learning dataset in a offline-dataset repository(datasets from the OpenML repository, the system evaluated a set of meta-features and used Bayesian optimization to determien and store an instantiation of the given ML framework with strong empirical performance for that dataset. When given a new dataset D, the Auto-sklearn system compute its meta-features, and rank all datasets by their distance to D in meta-feature space and select the stored ML framework for 25 nearest datasets for evlautation.

It's obvious that the meta-learning approach is a complementary to Bayesian optimization we discussed above. While the meta-learning select the best models to evaluate, the Bayesian optimization can fine-tune the performance of each model over time. There are in total 38 meta-features to evaluate a dataset.

ensemble-construction

Ensemble methods are techniques that create multiple models and then combine them to produce improved results and a better model. Note that Bayesian Hyperparameter optimization method discards all the models it trains during the course of the search, even including some that perform prediction task almost as well as the best model. Ratehr than discarding these models, the Auto-sklearn framework store them and use an efficient post-processing method to construct an ensemble out of them. As a result, this automatic ensemble consturcion avoid to commit itself to a single hyperparameter setting and is more robust.

In general, ensembles perform extremely well if the models they are based on are individually strong and make uncorrelated errors, and this is exactly sk-learn want. Auto-sklearn use ensemble selection method to construct their ensemble, which is basically a greedy procedure that start from an empty ensemble and then iteratively adds the model that maximize the ensemble validation performance.

2.2.4 conclusion on Auto-sklearn

The Auto-sklearn are constructed base on the sklearn framework, and it parametrized all the algorithms, preprocessing methods, and data preprocessing methods, resulting in 110 hyperparameters. And then, it use **SMAC** to feature the hyperparameters. works could cut the training time for each models and each data-point to controll the time, and auto-select the best-fit models.

3 Keras

3.1 introduction to Keras

Auto-Keras is the result of research by Jin et al. (2018) [1] from DATA LAB at Texas AM. As the name suggests it made to automate keras. It is a very promising toolkit to ease and speed-up finding optimal deep neural networks. It was released on June 27,2018 under MIT license and is still in a beta/pre-release condition. Let us have a closer look at Auto-Keras. The Auto-kears is a AutoML framework for the deep learning problems.

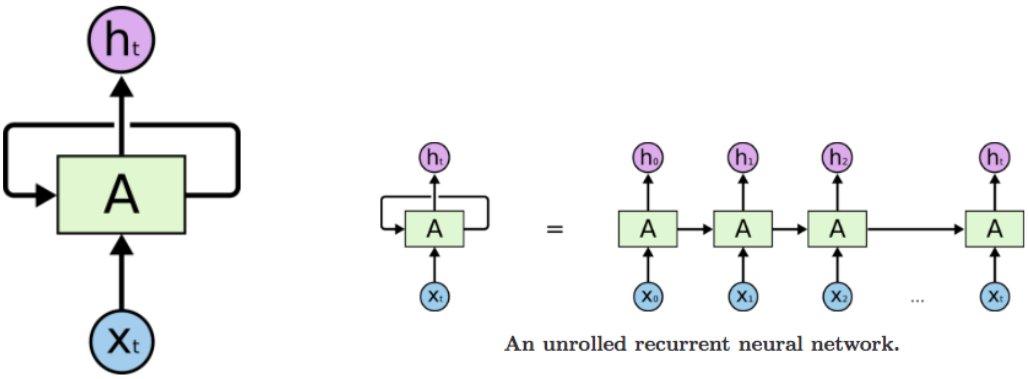
3.2 features

Other AutoML toolkits such as Google's AutoML use neural architecture search to find a (the most) suitable neural network architecture. The main disadvantage is that it is very slow because it requires full retraining from scratch of each model. Auto-Keras uses network morphism to reduce training time in neural architecture search. Further, it uses a Gaussian process (GP) for Bayesian optimization of guiding network morphism.
The Auto Keras aimed to find an optimal neural netowrk $f \in$ F,which could achieve the lowest cost on dataset D.

3.3 Fundation of LSTM

3.3.1 recurrent neural network

A recurrent neural network (RNN) is a class of artificial neural networks where connections between nodes form a directed graph along a temporal sequence. This allows the Neural network to "memory" the past infromation. the structure of RNN could generalize as below:



An unrolled recurrent neural network.

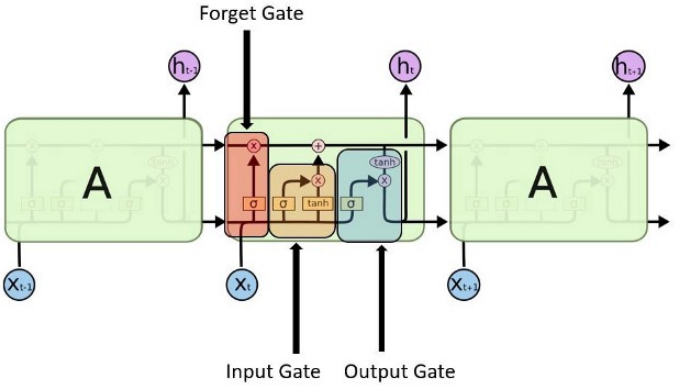
This chain property illustrate that RNN is the best Neural networks framework fo the ts data, because RNN can model sequence of data so that each sample can be assumed to be dependent on previous ones.So it's straightforward to use RNN to deal with the stock-price problem.
However, the disadvantages of RNN is obvious.

Gradient vanishing : Given that solving neural networks always involve Back propagation, we need solve the gradient for each layer. but RNN will have gradient vanishing and exploding problems, which make the historical data that is far from the present time-point does not contribute to the present value. As a result, RNN is not good framework for solving the information heavy ts data.

3.3.2 Long Short term memory

To avoid the problem of RNN, Long Short-Term Memory (LSTM) networks are a modified version of RNN , which makes it easier to remember past data in memory. The vanishing gradient problem of RNN is resolved here. LSTM is well-suited to classify, process and predict time series given time lags of unknown duration. It trains the model by using back-propagation.
The basic intuition behind is that: instead of the neural network layer of RNN, which process the data only by 1 state : the new input information and past information, the LSTM add 3 more gates to the chain and a "cell state' to memory the important information and "forget" the useless information. In other words, the LSTM create a filter to process the information from state to state.

the **cell state** is basically a state that specifically memorize the important information from the past information, and updates layer by layer.
In an LSTM network, three gates are present:



The input gate - discover which value from input should be used to modify the memory. The sigmoid and tanh function/neural network layer are basically functions/layers that transform the data through 0,1 or -1 to 1. Note that the sigmoid neural network could do dimension reduction. The past information h_{t-1} and new information x_t are passing through the sigmod neural network to the cell state. More intuitively, The input gate, determine which

information from last state and new information to be kept, and deliver the modified infromation to the cell state. (Memory gate).

The forget gate - discover what details to be discarded from the block. It looks at the past cell state and try to " forget" or filter the information from paste state. Similar to the input gate, the past information h_{t-1} and the new information $x_t arepassingthrough$It basically receive the information from last state and new information, and then pass through a sigmod neural network layer(as a filter layer), to the cell-state.(forget gate)

The input and forget gate process the data by forget some non-important information and memorize some new information.

The Output gate - output the data determined by: the last state information, the new input information, and the cell state(important information). the x_t and h_{t-1} are passing through a sigmod neural network layer, and C_t are passing through a tanh function.

The LSTM is a better framework to memory the long-term ts data given to its special data structure.

4 Application of ML on stock prediction

4.1 Problem settings

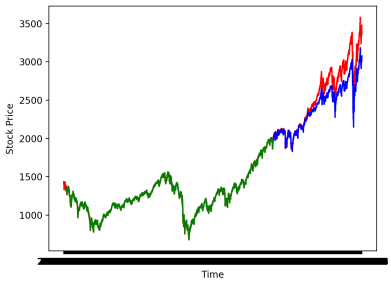
We choose the stock price of SP500 as our data set from 2000-2020. The tech company is developing super fast for the past 20 years. The non-stationary volatility makes it hard for the traditioanl time-series statistical methods to make prediction and fit the model. As a result, it's a good data-set to compare the performance of AutoML framework and the deep-learning methods.

4.2 Application of Auto-sklearn on stock prediction

we apply the Auto-sklearn to use the historical adjusted prices to make predictions one day out into the future, and We impor the TA-lib library to take the five-day, twenty-day, fifty-day moving average into account.

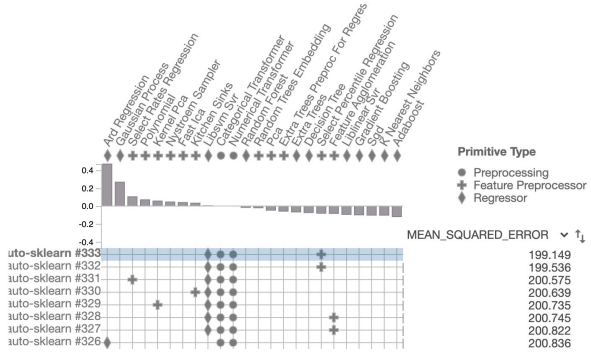
Moving-average is a widely used technical indicator that smooths out price trend by filtering out the " noise" from random-short-team price fluctuations (in other words, removing the trend factor). And we split the dataset into 70% training and 30% testing.

Then we define an AutoSklearnRegressor with a defualt of one hour to find appropriate models(Note that base on the Auto-sklearn algorithm, we could set up our training time to find the best model), and model fitting will be terminated if the machine learning algorithm runs over 30 seconds. Further more, we use Mean Squared Error(MSE) as our loss function to evaluate the performance. The prediction result is below, where the blue line is the actual result, and the red line is the prediction:

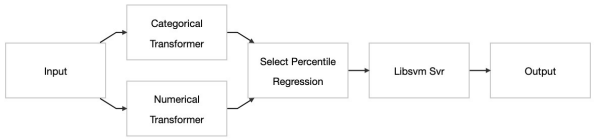


To research on the Pipeline and the exact mixing ML algorithm the auto-sklearn outputs, we use the package Pipeline-Profiler to explore the pipelines. Base on the analysis, we could find that the the primitive contribution model is the Ard Regression, Gaussian process with high MSE, and Adaboost with low MSE. Also, Figure(a) demonstrated that the top two pipelines(332 and 333) use the same prepossessing, feature preprocessor, and regressor following Figure(b) but with different hyperparameters. In addition, Figure(c) sorts each pipelines with ensemble weights, which show

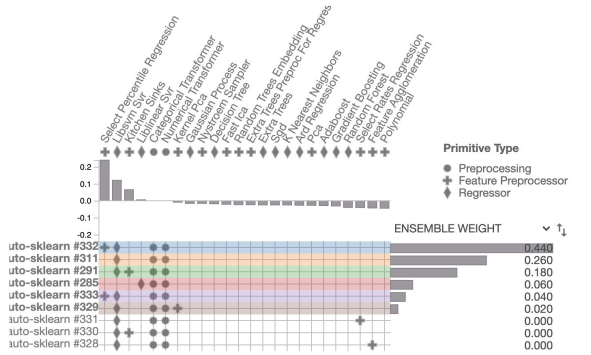
that there are 6 pipelines that ensemble the whole model with weight greater than zero, and we visualize the pipeline comparison through Figure(d).



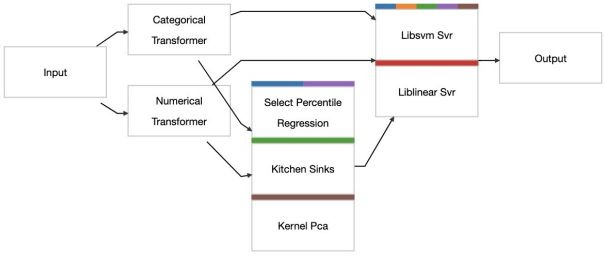
(a) Sorted with Mean Squared Error



(b) Pipeline #333 and #332



(c) Sorted with Ensemble Weight



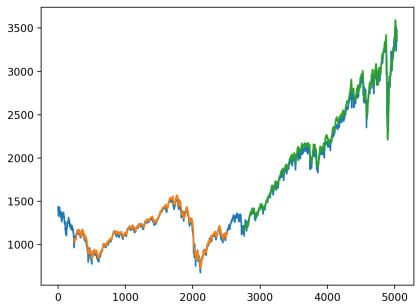
(d) Pipeline Comparison: Pipelines with Ensemble Weight

4.3 Application of LSTM on stock prediction

SP500 stock prices are predicted using LSTM network (Keras-Tensorflow).
Note that we normalized our values in range(0,1) to fit in the neural networking model. In LSTM, our dataset is 50% test data, and 50% trainning data.

The time-step is the time-step we use to input into our LSTM model each time; the batchsize is then clearly the same as the time-step. The epochs is the total number of training(the larger, the fitter, but more likely to cost over-fitting problem). By manual parameter optimization,We set up the timestep, batchsize, epochs to be 240, 240, 95, and the final LSTM network consists of 25 hidden neurons, and 1 output lyaer(1 dense layer).

We use Adam as our optimizer, and the dropout rate to be 0.1, Finally, we use root Mean Squared Error(RMSE) as our loss function to evaluate the performance. The prediction result is below, where the blue line is the actual result, and the green line is the prediction:



4.4 Pros and cons of two framework

The prediction plot shows that the Auto-sklearn cannot predict the black-swan draw-down of the market(the sudden drop or jump of the stock price), which cost a large differences in the prediction value and the actual stock price. However, the LSTM did a pretty well job on this prediction teak. It is clear to see that the LSTM outperform the Auto-sklaern model for this prediction teak on the real data. The reason why the LSTM have such a good prediction is that:

• LSTM is currently the best ML model for time-series data. The traditional regression method for prediction task is just worth than LSTM on this dataset. Given that the Auto-Sklearn are doing all the model-selection and parameter selection stuff based on sklearn instead of deep-learning, it's reasonable to see the differences in performance.

• the LSTM use the total data-set to train the data, and then it uses the future value to predict the past data. This is generally an ideal-setting..

5 Conclusion

Although the deep-learning framework LSTM seems to outperform the Auto-sklearn, That is mainly cost by the model differences. The Auto-sklearn could still provide a great consulting on how people should choose their model and their hyper parameter. We believe that as time evolves, the performance and prediction accuracy of AutoML will become better. At that time, There will be a new evolution of Machine learning, and the threshold of ML will become even lower.

6 Reference

Matthias Feurer, Jost Tobias Springenberg, Aaron klein, Manuel Blum, Katharina Eggensperger, Frank Hutter: Efficient and Robust Automated Machine Learning.
http://papers.nips.cc/paper/5872-efficient-and-robust-automated-machine-learning.pdf

SMAC introduction
https://zhuanlan.zhihu.com/p/139620752

Rich Caruana, Alexandru Niculescu-Mizil, Geoff Crew, Alex Ksikes: Ensemble Selection from Libraries of Models.
https://www.cs.cornell.edu/ alexn/papers/shotgun.icml04.revised.rev2.pdf

Exploring Auto-Sklearn Models with PipelineProfiler.
https://towardsdatascience.com/exploring-auto-sklearn-models-with-pipelineprofiler-5b2c54136044

LSTM introduction and impelmentation
https://zhuanlan.zhihu.com/p/104475016

Understanding RNN and LSTM
https://towardsdatascience.com/understanding-rnn-and-lstm-f7cdf6dfc14e

Meta-learning
https://en.wikipedia.org/wiki/Meta_learning_(computer_science)
Yahoo Finance
https://finance.yahoo.com/quote/^GSPC?p=^GSPC