Prediction of Corporate Bond Prices Based on Machine Learning Algorithms

Chiba Naoto

Nihon University, Shizuoka-Ken, Mishima City Omiya-Tyou 1-10-28, 206room, Japan

Keywords: Machine learning, Deep learning, Bonds

Abstract: This article uses whether several state-of-the-art machine learning or deep learning methods (GBDT models, NN, etc.) can be well applied to financial applications. We conduct comprehensive experiments on corporate bond price forecasting using several different computational models and provide an in-depth analysis and comparison of their performance. Furthermore, our results suggest that deep learning methods may not always be omnipotent when the data space is rather low/sparse, which is consistent with our general intuition and previous literature.

1. Introduction

In this project, we have tested that whether several state-of-the-art Machine Learning or Deep Learning Methods (GBDT Models, NNs, etc.) could be well adopted to Financial Applications. Moreover, unlike most of the previous literatures which mainly focus on the predictions of the equity market or indexes, we have worked on the bond market, which is usually considered as being short of sufficient information, but also being more structured meanwhile. Luckily, a bond price prediction challenge was once held on Kaggle sponsored by Benchmark Solutions, which aims to predict the next trade price of corporate bonds. The training data includes 762,678 data samples and 61 features in total, thus providing us sufficient training data to use. Based on such dataset, we shall then be able to conduct a comprehensive experiment to analyze and compare the performances of different computational models[1].

2. Methodology

2.1 Preprocessing

For Machine & Deep Learning, data preparation and pre-processing truly plays a critical role since it has laid the foundation for all the following processes. Without the data being well-prepared, it may easily come up with the “garbage in, garbage out” situation. Therefore, before we actually feed the raw data into our prediction models, we shall first conduct the Exploratory Data Analysis (EDA) process, where we have filled the missing sample points using -999, which is significant enough for our models to identify them as abnormal values. After that, we have adopted the StandardScaler() method from scikit learn to get rid of the scale effects.

Then, we shall concentrate on the dimensionality of the datasets. More specifically, if the degree of freedom has gone too high, it would be hard for us to find sufficient sample points to ascertain each aspect. Moreover, a lot of Machine Learning/Deep Learning models would inevitably involve the measuring and sequencing of “distances” (norms), whereas the concept of “near” and “far” would become quite ambiguous in high-dimensional spaces, thus incurring obstacles for such comparison/ranking. These issues caused by sparse datasets and difficult distance ordering are commonly encountered by almost all the ML/DL models, which is called the “Curse of dimensionality”[2].

Therefore, to resolve such problem, intuitively we shall try to reduce the number of dimensions. In other words, that is to try to find a low-dimensional subspace of the initial dataset, thus in which the density of the samples and the distance ranking difficulties will be largely ameliorated, as shown in figure 1.

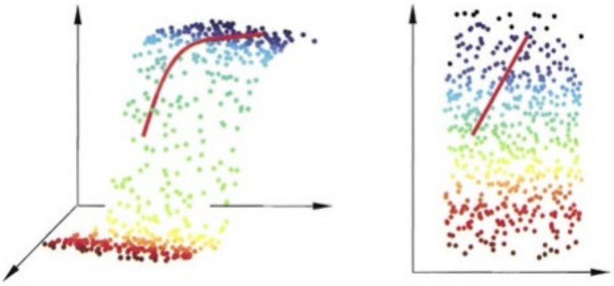


Fig.1: Diagram of Dimension Reduction

Specifically, we have adopted the Principal Component Analysis method in this project, which aims to try to find several linear combinations of the initial features as the new Principal Components (PCs, i.e., projections of the initial space to a hyperplane)[3]. Nevertheless, though such thought seems to be quite simple and plain, we have to also realize that such subspace or projection would inevitably mean loss of information. More specifically, we have actually sacrificed some features that are considered to be less important to achieve the target of dimensionality reduction. Therefore, it would result in an intractable trade-off between how much information we'd like to remain and how many dimensions we may want to cut-off. Hence, to resolve such dilemma, we have plotted the curve with Number of Components versus Cumulative Explained Variance to help us achieve higher efficiency of such dimension reduction process. According to the figure, one may easily notice that as #PCs has reached 42, the marginal information gain for further enlarging the #PCs would no longer be significant. Therefore, we have set the threshold as 42 components/0.999 cumulative explained variance, as shown in fig. 2.

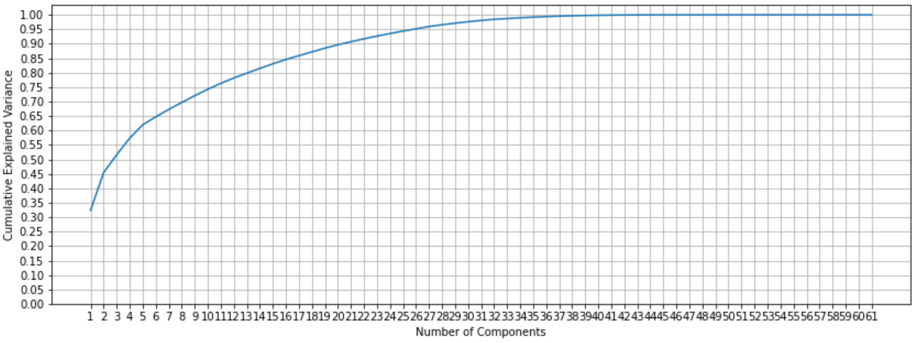


Fig.2: Cumulative Explained Variance Vs #Pcs

For hold-out cross validation, we have set 2/3 of the initial dataset for training and remaining 1/3 for testing, where the evaluation metric adopted here is the L1-Loss (MAE).

2.2 Models & Results

For this project, we have tested three kinds of models, respectively, Generalized Linear Models for traditional statistical methods as benchmark, LightGBM for Machine Learning and Recurrent Neural Networks (RNN) for Deep Learning.

2.2.1 Generalized Linear Models (Benchmark)

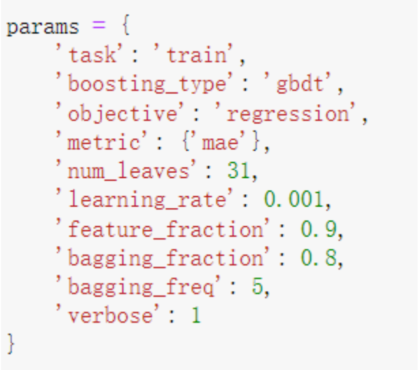
For these traditional statistical methods, we will not adopt the PCA techniques, as shown in table 1. which is usually deemed as techniques from Statistical/Machine Learning. Instead, we shall directly feed in the initial data without conducting dimension reduction (other pre-processing procedures remained). Moreover, we have also brought in the previous champion solution of the Kaggle challenge as another benchmark.

Table 1 Initial Data(Pca)

Methods	Loss-Train	Loss-Test	Duration
Contest Winner	-	0.68031	-
Ordinary Least Squares	0.628032	0.628364	2.75s
Weighted Least Squares	0.690423	0.691042	2.82s

2.2.2 Lightgbm

For the LightGBM Model, we have set up two different groups, with & w/o PCA respectively. The settings of hyperparameters are shown in the figure 3.



```
params = {
    ’task’: ’train’,
    ’boosting_type’: ’gbdt’,
    ’objective’: ’regression’,
    ’metric’: {’mae’},
    ’num_leaves’: 31,
    ’learning_rate’: 0.001,
    ’feature_fraction’: 0.9,
    ’bagging_fraction’: 0.8,
    ’bagging_freq’: 5,
    ’verbose’: 1
}
```

Fig.3: Hyperparameter Configuration for Lightgbm

Apart from the configurations above, we have also set a maximum boosting round of “num_boost_round=100,000” and an early stopping condition of “early_stopping_rounds=10”, which means that the training will be terminated early if the performance of our model cannot be further ameliorated after 10 consecutive rounds, even if it hasn’t reached the limit of maximum boosting rounds. If such condition is triggered, one may consider that the training performance has achieved a relatively stable convergence and could therefore be terminated early to avoid over-training and time waste. The training results are shown in fig.4 and fig.5. where we could notice that both groups have been early stopped. Table 2 shows the initial data (using PCA).

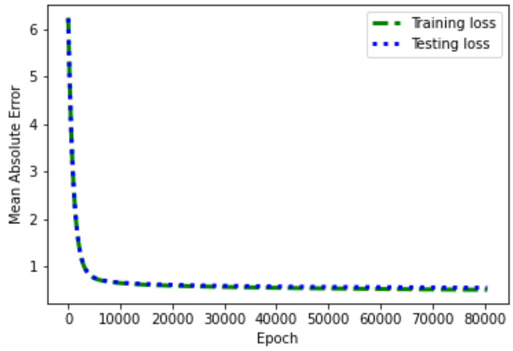


Fig.4: Learning Curve - Lightgbm (with Pca)

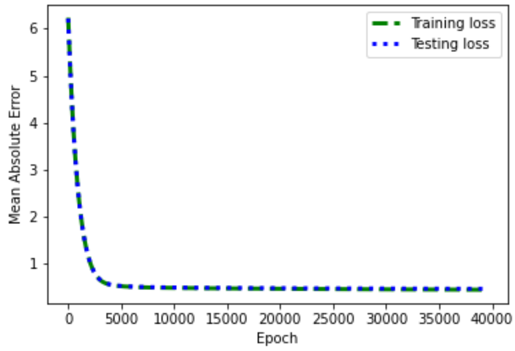


Fig.5: Learning Curve - Lightgbm (W/o Pca)

Table 2 Initial Data (with Pca)

Methods	Loss-Train	Loss-Test	Duration
LightGBM-with PCA	0.50648	0.531248	~25 min (80532 rounds)
LightGBM-w/o PCA	0.451926	0.471946	~14 min (39467 rounds)

2.2.3 Rnn

For RNN, having conducted PCA, we have constructed a network with four hidden layers. The detailed network structure is illustrated in the figure 6.

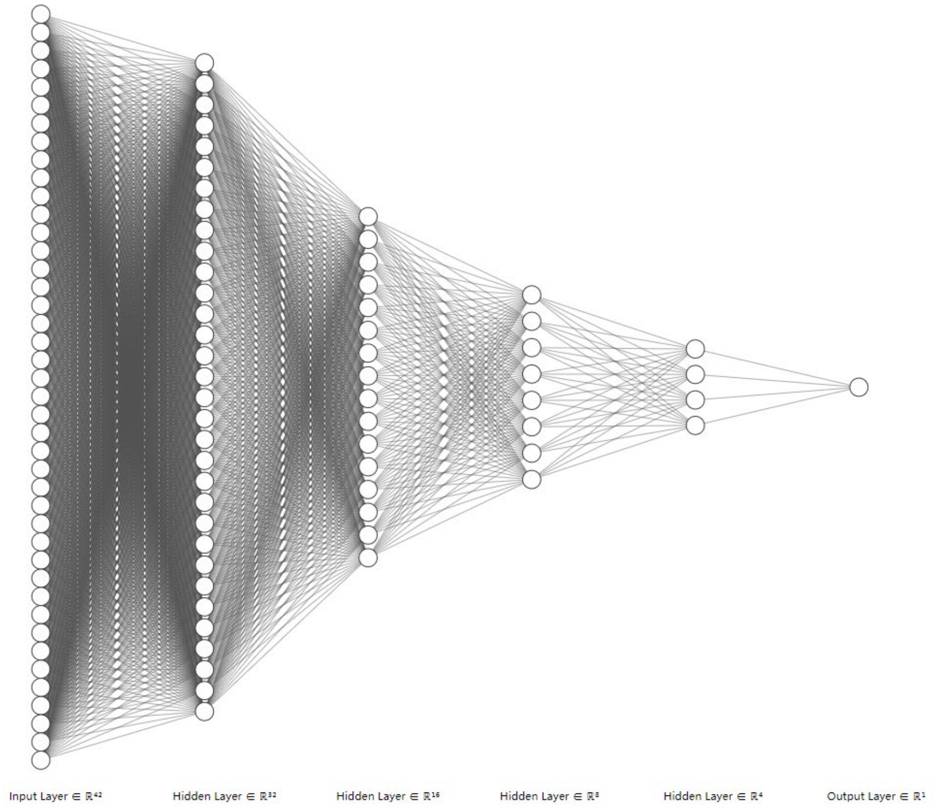

Input Layer ∈ $\mathbb{R}^{42}$
Hidden Layer ∈ $\mathbb{R}^{32}$
Hidden Layer ∈ $\mathbb{R}^{16}$
Hidden Layer ∈ $\mathbb{R}^{8}$
Hidden Layer ∈ $\mathbb{R}^{4}$
Output Layer ∈ $\mathbb{R}^{1}$

Fig.6: Structure of the Neural Network

For all hidden layers, we have adopted ‘relu’ as activation function, where the optimizer has been set to ‘adam’, with a learning rate of 0.001, batch size of 64 and epochs of 100.

The learning curve and training results are shown in Figure 7.

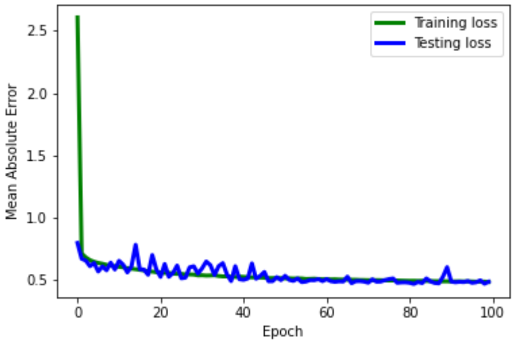


Fig.7: Learning Curve – Rnn

Table 3 Initial Data (Curve – Rnn)

Methods	Loss-Train	Loss-Test	Duration
RNN	0.475694	0.48327	~47 min

As shown in Table 3, It could be seen from the figure that at the early stage, the performances of our model do fluctuate a lot during training. However, as training progresses, it has gradually tended to converge to a relatively consistent level.

3. Conclusions

We summarize the performance of all models adopted in our project as shown in Figure 4.

Table 4 Initial Data

Methods	Loss-Train	Loss-Test	Duration
Contest Winner	-	0.68031	-
Ordinary Least Squares	0.628032	0.628364	2.75s
Weighted Least Squares	0.690423	0.691042	2.82s
LightGBM-with PCA	0.50648	0.531248	~25 min (80532 rounds)
LightGBM-w/o PCA	0.451926	0.471946	~14 min (39467 rounds)
RNN	0.475694	0.48327	~47 min

One may conclude from such results that apart from the traditional statistical methods with an extremely low time cost, LightGBM (w/o PCA) has dominated all other methods in terms of training time and prediction accuracy, which comes as no surprise-

Prior, as mentioned above, the dataset chosen for our project is not that large, with only 61 features in total. Therefore, probably directly feeding in the initial data won't incur too much dimensionality issues, but could actually provide full information for our models to better learn the implied correlations underneath. Hence, LightGBM (w/o PCA) may outperform LightGBM (with PCA), just as indicated by our results.

Moreover, for a highly structured financial product like bonds, the implied relationships between the features and our prediction target may not be that sophisticated. Hence, the inherent advantage of RNN on dealing with long-term/large-scale data or complicated relationships may not be fully utilized. On the contrary, the large training set and computing power required may conversely impede its performances. Therefore, probably for such prediction task with a relatively sparse dataset, our project has just somewhat proved the philosophical saying - "less is more".

In a nutshell, we have conducted a comprehensive experiment on corporate bond price prediction with several different computational models, with their performances being thoroughly analyzed and compared. Moreover, our results have indicated that the Deep Learning methods may not always be omnipotent under the condition of a rather low-dimensional/sparse data space, which is in accord with our general intuition and previous literatures.

References

[1] Khor E T. A data mining approach using machine learning algorithms for early detection of low-performing students[J]. The international journal of information and learning technology, 2022(2):39.

[2] Garcia Rodriguez M J , Rodriguez Montequin V , Aranguren Ubierna A , et al. Award Price Estimator for Public Procurement Auctions Using Machine Learning Algorithms: Case Study with Tenders from Spain[J]. Studies in informatics and control, 2021(4):30.

[3] Chen, J., X. Shen , and Q. Chen . "Prediction of Maximum Surface Settlements of BaiHua Tunnel Section based on Machine Learning." Journal of Physics: Conference Series 2185.1(2022):012042-.