IEEE Second International Conference on Data Stream Mining & Processing
August 21-25, 2018, Lviv, Ukraine

Using Stacking Approaches for Machine Learning Models

Bohdan Pavlyshenko
SoftServe, Inc., Ivan Franko National University of Lviv
Lviv, Ukraine
bpavl@softserveinc.com, b.pavlyshenko@gmail.com

***Abstract*—In this paper, we study the usage of stacking approach for building ensembles of machine learning models. The cases for time series forecasting and logistic regression have been considered. The results show that using stacking technics we can improve performance of predictive models in considered cases.**

***Index Terms*—machine learning, stacking, forecasting, classification, regression**

I. Introduction

One of effective approaches in machine learning classification and regression problems is stacking. The main idea of stacking is using predictions of machine learning models from the previous level as input variables for models on the next level. Using multilevel models with stacking approach is very popular among the participants of Kaggle [1] community. On Kaggle platform, different business companies propose their problems with datasets for data scientists competitions to develop predictive models with the best accuracy. Time series can be analysed by different approaches such as ARIMA, linear models, machine learning models [2].

In this study, we consider the applying of stacking approach to predictive models for time series and for logistic regressions.

II. Using Linear Regression for Models Stacking

We are going to consider several simple cases of approaches in the sales times series forecasting. For our study, we used the data set from Kaggle competition 'Rossmann Store Sales' [3]. Combining different predictive models with different sets of features into one ensemble, one can improve the result accuracy.. There are two main approaches for model ensembling - bagging and stacking. Bagging is a simple approach when we use weighted blending of different model predictions. Such models use different types of classifiers with different sets of features and meta parameters. If forecasting errors of these models have weak correlation, then these errors will be compensated by each other under the weighted blending. The less is the error correlation of model results, the more precise forecasting result we will receive. Let us consider the stacking technic [4] for building ensemble of predictive models. In such an approach, the results of predictions on the validation set are treated as input regressors for the next level models. As the next level model, we can consider a linear model or another type of a classifier, e.g. Random Forest classifier or Neural Network. In our study, linear regression and machine learning models were from scikit-learn python package, neural network was from Keras python package. It is important to mention that in case of time series prediction, we cannot use a conventional cross validation approach, we have to split a historical data set on the training set and validation set by using time splitting, so the training data will lie in the first time period and the validation set - in the next one. Fig. 1 shows the time series forecasting on the validation sets obtained using different models. Predictions on the validation sets are treated as regressors for the linear model with Lasso regularization. Fig. 2 shows the results obtained on the second-level with linear regularized model. Only two models from the first level (gradientBoosting and ExtraTree) have non zero coefficients for their results. For other cases of sales datasets, the results can be different and the other models from the first level can play more essential role in the forecasting.

III. Sales time series forecasting

The company Grupo Bimbo organized Kaggle competition 'Grupo Bimbo Inventory Demand' [5]. In this competition, Grupo Bimbo invited Kagglers to develop a model to forecast accurately the inventory demand based on historical sales data. I had a pleasure to be a teammate of a great team 'The Slippery Appraisals' which won this competition among nearly two thousand teams. We proposed the best scored solution for sales prediction in more than 800,000 stores for more than 1000 products. Our first place solution can be found at [6]. To built our final multilevel model, we exploited AWS server with 128 cores and 2Tb RAM. For our solution, we used a multilevel model, which consists of three levels (Fig. 3). We built a lot of models on the 1st level. The training method of most 1st level models was XGBoost. On the second level, we used a stacking approach when the results from the first level classifiers were treated as the features for the classifiers on the second level. For the second level, we used ExtraTrees classifier, the linear model from Python scikit-learn and Neural Networks. On the third level, we applied a weighted average to the second level results. The most important features are based on the lags of the target variable grouped by factors and their combinations, aggregated features (min, max, mean, sum) of target variable grouped by factors and their combinations,

978-1-5386-2874-4/18/$31.00 2018 IEEE

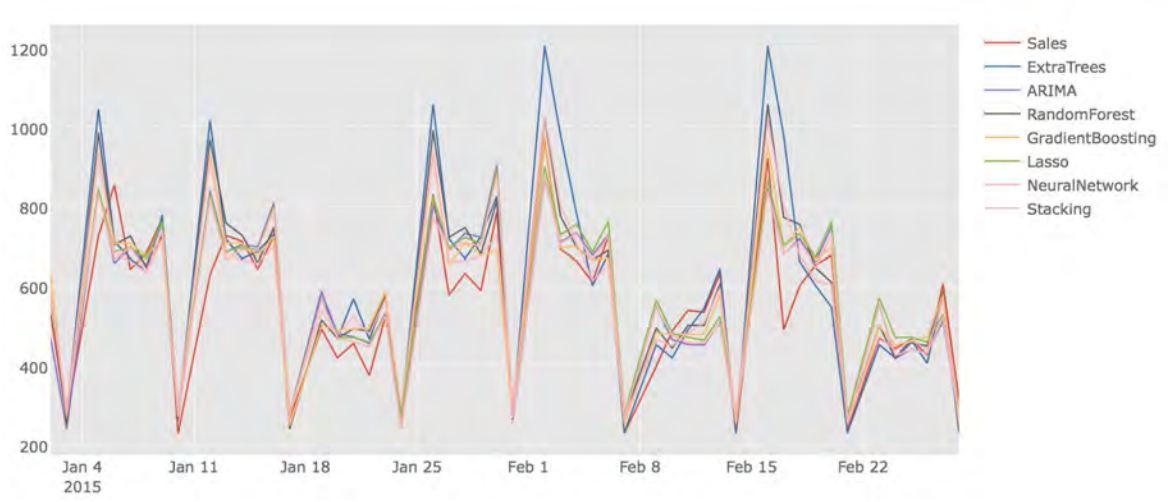

Fig. 1. Different methods for time series forecasting

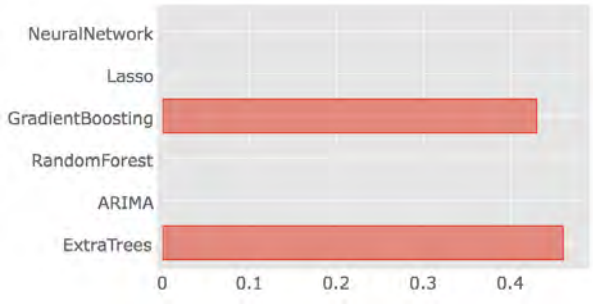

Fig. 2. Coefficients for stacking linear regression

frequency features of factors variables. One of the main ideas in our approach is that it is important to know what were the previous week sales. If during the previous week too many products were supplied and they were not sold, next week this product amount, supplied to the same store, will be decreased. So, it is very important to include lagged values of target variable as a feature to predict next sales. More details about our team's winer solution are at [6]. The simplified version of the R script is at [8]. Our winner solution may seem to be too complicated, but our goal was to win the competition and even a small improvement in forecasting score required essential numbers of machine learning models in the final ensemble. Real business cases with a sufficient accuracy can be simpler.

IV. Stacking approach for logistic regression

Let us consider using stacking approach for logistic regressions problems. In the Kaggle competition 'Bosch Production Line Performance' [11], the problem of internal failures on assembly lines was considered. The data set consists of measurements for components on assembly lines. This case is a type of logistic regression problem with highly imbalanced classes. The problem lies in predicting which parts will fail a quality control. In the work [12], the logistic regression approach in manufacturing failure detection was considered. As a data set for the analysis, we used the data from Kaggle competition 'Bosch Production Line Performance' [11]. The data set has a lot of anonymized features. For modeling we used linear, machine learning and Bayesian approaches. To find influence of different factors we exploited the generalized linear model. Using Bayesian approach for logistic regression, we can get the probability distribution function for model parameters. Having statistical distribution we can make risk assessments. To build machine learning models, we used XGBoost classifier from R package 'xgboost' [7], [9], [10]. The data in this set have highly imbalanced classes. To reduce this problem we used undersampling approach. The samples with positive value 1 for target variable were retained without changes. The samples with value 0 for target variable were randomly sampled, so the total number of data items was reduced. For categorical features we used one-hot encoding. The results of the classification were the probability for positive responses. Combining machine learning models and linear or Bayesian models on different levels can give us improved results for logistic regression. Fig. 4 shows a diagram of such possible stacking model. On the first level, there are different XGBoost classifiers with different sets of features and subsets of samples. On the second level, probabilities predicted on the first level can be blended with appropriate weights using linear or Bayesian regression. To evaluate classification performance we used Matthews correlation coefficient (MCC):

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$$

where TP represents the number of true positives, TN represents the number of true negatives, FP represents the number of false positives, and FN represents the number of false negatives. Let us consider the use of generalized linear model for stacking logistic regression with independent variables which are the probabilities predicted by XGBoost models on the first levels.

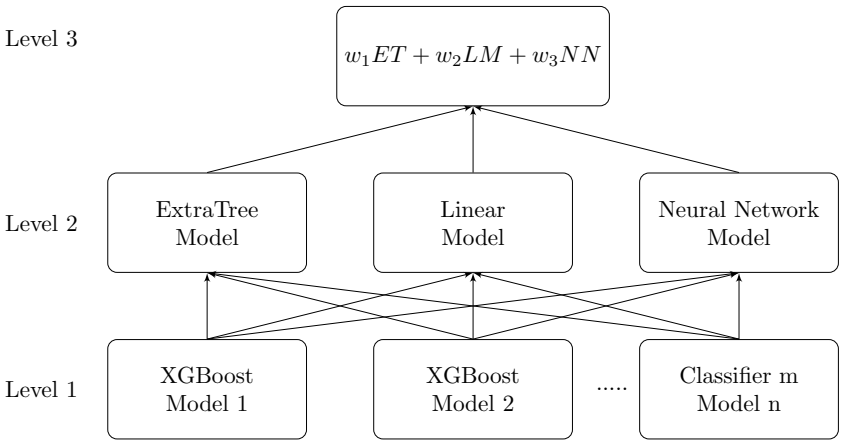

Fig. 3. Mulitlievel machine learning model for sales time series forecasting

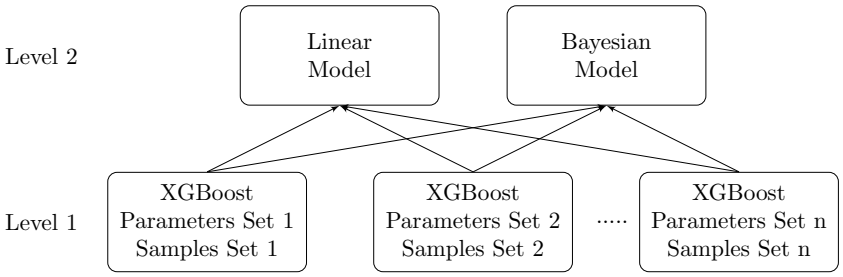

Fig. 4. Stacking model for logistic regression

We used different sets of parameters for 3 XGBoost models, they are - set 1: max.depth = 15, colsample_bytree = 0.7; set 2: max.depth = 5, colsample_bytree = 0.7; set 3: max.depth = 15, colsample_bytree = 0.3. For these 3 models, we used the same subset of samples. Fig. 5, 6 show the dependence of Matthews correlation coefficient from a probability threshold for different subsets of features, where features set 2 is features set 1 with 4 added magic features. So-called magic features which are based on the ID of samples were considered by the participants of the competition at [13]–[15]. For Bayesian models, we used the same 3 subsets of parameters with different subsets of samples. As it was shown above, for different samples subsets, we received slightly different results for Matthews correlation coefficient. These differences can be taken into account using Bayesian model. For Bayesian inference we used JAGS sampling software [16], [17]. We used Bayesian model for logistic regression. As covariates we used the probabilities predicted by three XGBoost models. Fig. 7 shows the boxplots for coefficients of probabilities predicted by different XGBoost models.

V. Conclusion

In our study, we considered stacking approaches for time series forecasting and logistic regression with highly imbalanced data. Using multilevel stacking models, one can receive more precise results in comparison with single models. For stacking machine learning models the linear regression with Lasso regularization, other machine learning model, Bayesian model can be used. Using stacking model on the second level with the covariates that are predicted by machine learning models on the first level, makes it possible to take into account the differences in results for machine learning models received for different sets of parameters and subsets of samples. As obtained results show, using stacking approach for machine learning models we can improve performance of predictive models.

References

[1] Kaggle: Your Home for Data Science. URL: http://kaggle.com
[2] B. M. Pavlyshenko. “Linear, machine learning and probabilistic approaches for time series analysis,” in IEEE First International Conference on Data Stream Mining & Processing (DSMP), Lviv, Ukraine, pp. 377-381, August 23-27, 2016.
[3] “Rossmann Store Sales”, Kaggle.Com, URL: http://www.kaggle.com/c/rossmann-store-sales .
[4] D. H. Wolpert. “Stacked generalization.” Neural networks, 5(2), pp. 241-259,1992.
[5] Kaggle competition “Grupo Bimbo Inventory Demand ” URL: https://www.kaggle.com/c/grupo-bimbo-inventory-demand
[6] Kaggle competition “Grupo Bimbo Inventory Demand” #1 Place Solution of The Slippery Appraisals team. URL:https://www.kaggle.com/c/grupo-bimbo-inventory-demand/discussion/23863

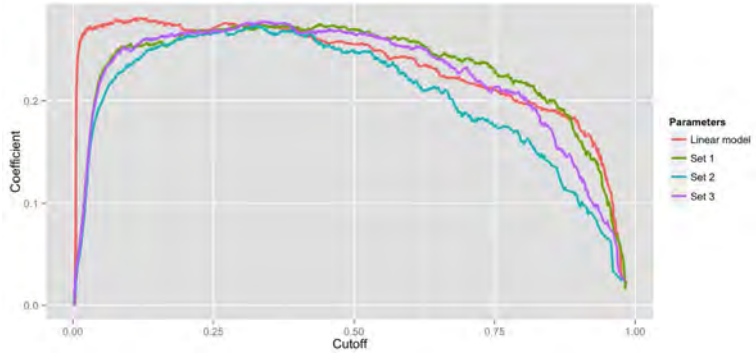

Fig. 5. Matthews coefficient for different XGBoost parameter sets (feature set 1)

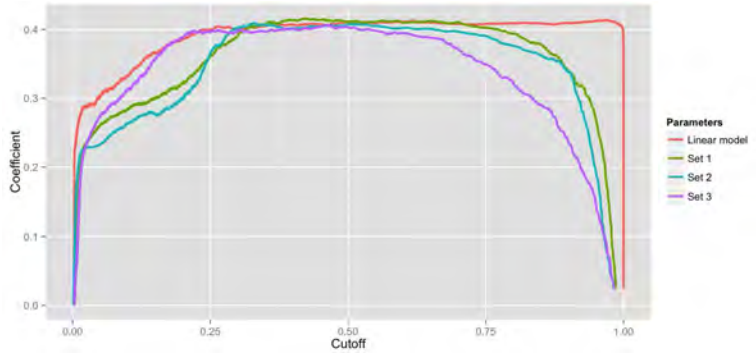

Fig. 6. Matthews coefficient for different XGBoost parameter sets (feature set 2)

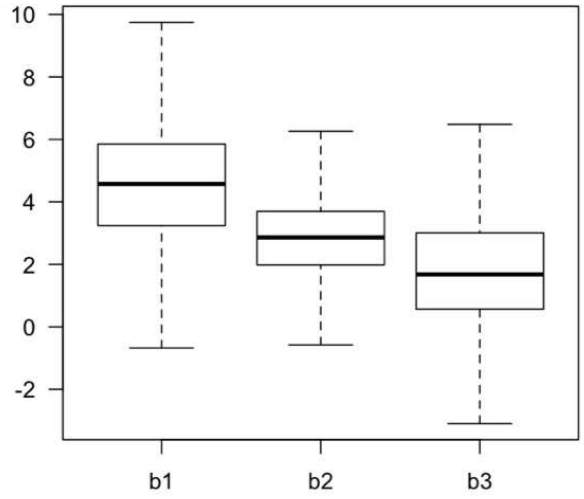

Fig. 7. Boxplots for coefficients of probabilities predicted by different XGBoost models

[7] T. Chen and C. Guestrin. "Xgboost: A scalable tree boosting system." In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. ACM. 2016, pp. 785-794.

[8] Kaggle competition "Grupo Bimbo Inventory Demand" Bimbo XGBoost R script LB:0.457. URL: https://www.kaggle.com/bpavlyshenko/bimbo-xgboost-r-script-lb-0-457

[9] J. Friedman. "Greedy function approximation: a gradient boosting machine.", Annals of Statistics, 29(5):1189-1232, 2001.

[10] J. Friedman. "Stochastic gradient boosting.", Computational Statistics & Data Analysis, 38(4):367-378, 2002.

[11] Kaggle competition "Bosch Production Line Performance". URL: https://www.kaggle.com/c/bosch-production-line-performance

[12] B. Pavlyshenko. "Machine learning, linear and bayesian models for logistic regression in failure detection problems.,' in IEEE International Conference on Big Data (Big Data), Washington D.C., USA, pp. 2046-2050, December 5-8, 2016.

[13] Kaggle competition "Bosch Production Line Performance". The Magical Feature : from LB 0.3- to 0.4+. URL:https://www.kaggle.com/c/bosch-production-line-performance/forums/t/24065/the-magical-feature-from-lb-0-3-to-0-4

[14] Kaggle competition "Bosch Production Line Performance". Road-2-0.4+. URL:https://www.kaggle.com/mmueller/bosch-production-line-performance/road-2-0-4

[15] Kaggle competition "Bosch Production Line Performance". Road-2-0.4+ –>FeatureSet++. URL: https://www.kaggle.com/alexxanderlarko/bosch-production-line-performance/road-2-0-4-featureset

[16] John Kruschke. Doing Bayesian data analysis: A tutorial with R, JAGS, and Stan. Academic Press, 2014.

[17] Martyn Plummer. JAGS Version 3.4.0 user manual. URL:http://sourceforge.net/projects/mcmcjags/files/Manuals/3.x/jags_user_manual.pdf