University of Zurich

University of Zurich[UZH]

Machine Learning in Finance

Benjamin Zimmermann

Group Project: Constructing Investment Recommendations

Authors:

Nadine Brönnimann, 18-702-167

Jorge Fernando da Silva Gonçalves, 18-707-422

Damjan Kostovic, 18-717-371

Sari Issa, 18-745-273

Date of Submission

April 18, 2021

Contents

1 Introduction

The intrinsic value of a specific stock is unclear and there are several methods on how to find out if a company is overpriced or underpriced in the stock market. The seminar "Machine Learning in Finance" has introduced some important machine learning (ML) algorithms and techniques which enable us to produce precise investment recommendations. With the dataset on stocks of the US stock market obtained from kaggle ("200+ Financial Indicators of US stocks (2014-2018)"), over the sample period from 2014 to 2018, we derived an investment strategy. The investment recommendation uses the yearly price change of a stock relative to S&P500 returns.[1] This summary aims to outline the final setup on our recommendation model.

First, in Section 2 we will present how we prepared the data. Then, in Section 3 we will show how we derived our investment recommendations and describe the machine learning algorithms and techniques used in this project. Section 4 presents the results, and Section 5 concludes.

2 Data Cleaning

2.1 First steps

To have a better idea of the data, we took a closer look at the initial state of the datasets. We conducted an exploratory data analysis (EDA) and analyzed the structure and features of the data. This step was done carefully in order to avoid erroneous data or data with extreme outliers for our ML approaches which could lead to wrong conclusions. The data consists of five datasets with yearly stock data from 2014 to 2018. There are different numbers of stocks and also different variables which operate on different scales. For that reason, one of the first steps before fitting the algorithms was to standardize the variables to have a mean of zero and a variance equal to 1 for each feature.

Another issue is that the data includes many missing values (NaN's) and zero values. Contrary to the missing values, we will assume that zero values are correct and leave them in our dataset. In order to check whether the NaN's occur systematically, we constructed heatmaps for each year. The heatmaps indicate that the NaN's occur systematically to some extent, so simply using a median imputer or dropping the data could be problematic. Our analysis would be less accurate

[1] https://www.macrotrends.net/2324/sp-500-historical-chart-data

since the way in which systematic NaN's are missing is a piece of information itself which should not be disregarded by dropping the data or using a rudimentary mean imputer. The analysis also suggests that there is no clear relation between the missing values and the price variation of the corresponding stock. Therefore, we dropped some observations that do not contain any relevant information about the price variation. Observations with more than 50% NaN-values are removed, such as features with more than 20% missing values. This leads to more compact datasets and was helpful for computational efficiency.

2.2 Missing values

The large number of missing values was the major challenge in the data preparation process. We applied different methods to handle missing values and to examine which imputation method improves our model performance the most. The quality of the NaN imputation should, in theory, have a large effect on classification performance.

Our first approach was the imputation with sector-wise median values. We applied three modifications to the median-imputed dataset to obtain three median-imputed datasets. The first modification did not remove outliers at all. The second modification removed equity premium outliers. Finally, the third modification removed equity premium outliers and added additional macroeconomic variables.

Secondly, we used a Random Forest iterative imputer. Interestingly, we did not observe a clear improvement in classification quality when using the more complex iterative imputer but rather a deterioration. Due to this fact, there were no advantages that could be retrieved by this method.

2.3 Feature Selection and Economic Variables

For feature selection, we first removed all the features which consisted of more than 20% of NaN's. The remaining features were then used in the testing and training of the models. For one dataset, we also added yearly macroeconomic indicators for the US, namely the real interest rate, GDP growth, Inflation and unemployment rate. These variables take into account the economic environment for every year.

After standardizing the variables, we performed a Principal Component Analysis (PCA) which is a helpful tool for the exploratory data analysis and simplifies the dimensional representation of our data (James et al., 2013). For our EDA, the scree plot of the PCA showed that several of the principal components are needed to constitute a large amount of variability in the dataset. However, the first 100 components then are able to explain almost all variability which means that the

remaining components are redundant. This leads to multicollinearity which is an issue for some of the algorithms. To tackle this issue, we used regularization-based and tree-based algorithms.

2.4 Response Variable

The next step was to define our response variable and bring it to the desired form. Given that the investment recommendation is based on the stock's performance compared to the S&P500, we classified the stocks into different investment strategies. To quantify the relative performance to the S&P500, we deducted the yearly S&P500 returns from the price variation of each share which resulted in the equity premium. Using the equity premium, we classified the shares as follows: "Buy" is recommended if the share outperformed the S&P500 by more than 2.5 percentage points, whereas "Hold" is the right strategy if the share performed similar to the index. "Sell" should be applied if the share performed below the S&P500 by more than 2.5 percentage points for the corresponding year. With the aim of deriving an investment recommendation that predicts the share's performance in the right direction, we aimed to classify shares into these investment recommendation groups.

One finding was, that there was a class imbalance in the data. For example, there were always relatively few shares which belonged to the "Hold" group. The class imbalance was once tackled within SVMs. This resulted however in a significantly worse performance and was, therefore, not used for the other algorithms.

3 Deriving the Investment Strategy

To predict equity movements and train our model we used several machine learning algorithms. More precisely, we ran the following algorithms: Multiple Linear Regression, Ridge Regression, Lasso Logistic Regression, Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA), Random Forest, Gradient Boosting and Support Vector Machine (SVM). In this summary we will not explain the implementation process any further because the jupyter notebooks cover the methods comprehensively.

4 Results

This section presents a summary of the results from our predictive models. Table 1 reports the accuracy scores on the test sets of the Machine Learning algorithms applied in this case study. We see that the performance does differ to some extent

between the different algorithms used. Strikingly, QDA performed by far the worst and achieved very low test scores. We discuss in Section 5 possible reasons for this. SVM seems to be the algorithm which consistently produces the best test scores and seemed to work fairly well. The dataset which includes the economic variables (and without outliers) consistently produces the best test scores across most of the algorithms used. However, the best accuracy was achieved with SVM on the median imputed data with removed outliers using a polynomial kernel. Therefore, we rely our investment recommendations on this combination. The other datasets perform well in some settings, but some instability is viewed across different algorithms. Additionally, one can see that the different algorithms produce differently stable results when using the four datasets. Some algorithms like SVM or $GradBoost$ have stable test scores among all datasets, while MLR or QDA have a much higher variability. Overall, the difference in test scores between the median imputer with outliers and without outliers is very small except for the MLR. There seems to be some evidence in the results that removing the outliers and adding the economic variables does slightly improve performance.

Table 1: Machine Learning Algorithm Score

This table presents the test scores for the corresponding algorithm. MLR refers to the Multiple Linear Regression, $RidgeReg$ to Ridge Regression, $LassoLogReg$ to the Lasso Logistic Regression, $RandFor$ to the Random Forest algorithm, $GradBoost$ to Gradient Boosting and SVM to Support Vector Machines.

	Test Score			
Algorithm	Median imputer	Outlier Removal	Economic variables	Iterative Imputer
MLR	0.4743	0.3981	0.5911	0.4477
$RidgeReg$	0.5446	0.5523	0.6023	0.5098
$LassoLogReg$	0.5513	0.5584	0.6060	0.5475
LDA	0.5429	0.5501	0.5999	0.5093
QDA	0.1135	0.1215	0.3384	0.1811
$RandFor$	0.5673	0.5714	0.5631	0.5646
$GradBoost$	0.5781	0.5879	0.5843	0.5776
SVM	0.6020	0.6072	0.6067	0.5972

5 Conclusion

Overall, it is obvious that the issues with the dataset discussed above had a detrimental impact on the precision of our recommendation algorithms. In most of the years our test score was only slightly better than always classifying a stock as the class with the highest occurrences. The reasons for this relatively poor performance could not be pinpointed to specific issues either. There was no algorithm that performed exceptionally well, and variations within the algorithms, like using a GridSearchCV Pipeline or using more complex models often did not cause large improvements either. If computational efficiency would not be a problem, we may have been able to improve performance by imputing a larger share of the NaNs. Another possible issue is that we used too many variables in most cases. The feature selection was not strict enough. This is visible in the poor performance of the QDA classification. QDA delivered a much worse prediction than LDA. The high amount of features and the resulting multicollinearity led to high variances for QDA and subsequently low test scores. However, we were able to produce a stable prediction across eight different algorithms integrating many different important factors. The validity and the results from the ML algorithms used in this study are convincing and enables a reliable prediction although it has to be treated with care. Consistent with conclusion from Ou and Wang (2009), we recommend applying different algorithms to predict stock price movements. This enables investors to forecast future returns more accurately because each classification delivers information that helps improve predictions for investment recommendation.

References

Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. *In Proceedings of the fifth annual workshop on Computational learning theory.*

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32.

Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., & Shah, R. (1993). Signature verification using a" siamese" time delay neural network. *Advances in neural information processing systems*, *6*, 737–744.

Drucker, H., Burges, C. J., Kaufmann, L., Smola, A., & Vapnik, V. (1997). Support vector regression machines. *Advances in neural information processing systems*, *9*, 155–161.

Friedman, J., Hastie, T., & Tibshirani, R. (2001). *The Elements of Statistical Learning.* New York: Springer.

Geurts, P., Ernst, D., & Wehenkel, L. (2006). Extremely randomized trees. *Machine Learning*, *63*(1), 3–42.

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An introduction to statistical learning.* New York: springer.

Ou, P., & Wang, H. (2009). Prediction of Stock Market Index Movement by Ten Data Mining Techniques. *Modern Applied Science*, *3*(12), 28–42.

Raschka, S., & Mirjalili, V. (2017). *Python machine learning: machine learning and deep learning with Python, scikit-learn, and TensorFlow.* Packt publishing ltd.