**Supervised Learning for Time Series Forecasting**

**Abstract**

This technical report will be on the usage of supervised learning models in the scenario of time series forecasting. The data was transformed in a way that included all the required lag values, including seasonal lags, in the same row. Then, the models were trained and tested. These supervised learning models were tuned based off root-mean-squared-error. The supervised learning model’s individual hyperparameters were tuned, if it had one, on top of the number of lags. The models tested were XGBoost, KNN, SVR, and Linear Regression. Certain models performed better than others, and some models just completely fail to capture the variation of the forecast. However, in the end, the classic ARIMA model still outperformed the supervised learning models. Also, what I have found is that the supervised learning models are unable to capture the residuals from the training and sometimes may be unable to capture the seasonality of the time series.

**Introduction**

Time Series forecasting is the process of analysing time series data using statistics and modelling to make predictions and inform strategic decision-making. Supervised Learning is a subcategory of machine learning; it is identified by it’s use of labelled datasets to train algorithms that to classify data or predict outcomes accurately. In essence, Time Series forecasting is Supervised Learning but with the inclusion of the time dimension, the past and present. With this thought, it seemed possible that a Supervised Learning model would be able to perform the tasks of a Time Series forecasting model. All that needs to be done is to solve the inclusion of past and present data.

**Related Works**

One related of the related works was done by Unai Lopez Ansoleaga. He has written about the machine learning approach of solving time series problems. He did his testing on Kaggle’s Bikes Sharing Demand competition dataset. He used Microsoft’s Light Gradient Boosting Machine Model, which was developed by Microsoft and “it beats the standard XGBoost in training speed and sometimes accuracy”.

By training without lags, he got this result:

Chart, line chart

Description automatically generated

Chart, funnel chart

Description automatically generated

After checking the importance of the variables, he created lags of the count based on the variable with highest importance.

Chart, line chart

Description automatically generated

This has significantly improved his model.

**Experiment**

Exploratory Data Analysis

The dataset I will be using is a Kaggle dataset on Digital Currency. There are 1000 records from 2018-05-07 to 2021-01-30. The variables are:

Text

Description automatically generated with low confidence

From this dataset, I want to forecast the trading volume of digital currency with respect to the different types of stock prices. Using the Dickey-Fuller test, it shows that the time series of the trading volume is stationary with a p-value of 0.036416, which is less than 0.05. The seasonal decomposition of the time series also shows that there is seasonality of 7 days.

Histogram

Description automatically generated with medium confidence

Also checking the correlation between all the variables:

Chart, histogram

Description automatically generated

This heat map shows that there is a direct correlation between all the features except for volume. Therefore, there is reason to drop all but one of the features when building the models.

Feature Engineering

In order to create the lags, I used the pd.shift() function to move the values n spaces up or down and created a new variable of the lags. E.g. with lag = 2

|  |  |  |
| --- | --- | --- |
| **var (t)** | **var(t-1)** | **var (t-2)** |

The same thing was done with seasonal lags but instead of shifting them n spaces, they were shift n x period spaces. E.g. with period = 2 and lag = 2

|  |  |  |
| --- | --- | --- |
| **var (t)** | **var(t-2)** | **var (t-4)** |

Model Building

I created 4 supervised time series models using the supervised learning models, XGBoost, KNN, SVM, and Linear Regression. They are all built in the same way it is just that the models used to train the predict is the respective models. Before the data is sent to the models, they will always undergo the feature engineering mentioned previously.

When performing the forecast, the model will assume that the dataset entered is a direct continuation of the dataset used to fit it. So, it will take the lags from the fitting dataset to predict the first few values. However, for the subsequent values that fall outside of the lags of the fitting dataset, the values used will be that of the predicted values. E.g. the model will predict the first value to be ***n*** then for the next value the ***var (t-1)*** will be the ***n*** value previously.

Hyperparameter tuning

I performed a walk-forward validation test for hyperparameter tuning. For example, I performed 4 splits of test size 100, what this means is that the dataset is split into 4 chunks, the last 3 chunks are size 100. At the start the first chunk will train and forecast against the second chunk. Then, the first chunk plus second chunk will train and forecast against the third chunk. Lastly, the first plus second plus third chunk will train and forecast against the last chunk.

Results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model** | **Other Hyperparameters** | **N \_lags** | **Seasonal\_n\_lags** | **RMSE** |
| XGBoost | - | 3 | 2 | 51327.08772845146 |
| - | 6 | - | 56851.1820771072 |
| KNN | k = 13 | 1 | - | 45001.89115279937 |
| k = 5 | 1 | 2 | 60406.69353189595 |
| k = 5 | 9 | - | 52201.52201.216485396195 |
| SVM | - | 1 | 9 | 61131.72429583171 |
| - | 9 | - | 62289.679460216226 |
| Linear Regression | - | 9 | 5 | 48584.054721672845 |
| - | 3 | - | 38095.67789188195 |

**Discussion**

Some models such as the XGBoost with n\_lags = 3 and seasonal\_n\_lags = 2 will show some form of seasonality in the forecast.

Chart, histogram

Description automatically generated

However, there are other models that are unable to show any form of seasonality or randomness in the forecast. Such as SVM.

Chart, line chart, histogram

Description automatically generated

Example from SVM with n\_lags = 1, seasonal\_n\_lags = 9

Comparing against the classic ARIMA model.

Chart, histogram

Description automatically generated

The ARIMA model is better at capturing the seasonality and error of the forecast as compared to the supervised learning models. However, on the right of the forecast, there is a large deviation from the actual data. This caused the RMSE for this model to be very high.

**Conclusion**

In conclusion, the standard ARIMA model is still generally better than the supervised learning models at forecasting as it can capture a more accurate seasonality and error. However, the supervised learning models are not a completely unviable option. There are some that do not work, but those that do work are able to capture some of the seasonality of the forecast. However, none of them are able to capture the error of the forecast.

**References**

Ahmed. (2021, January 30). *Digital currency - time series*. Kaggle. Retrieved August 12, 2022, from https://www.kaggle.com/datasets/ahmedadam415/digital-currency-time-series

Ansoleaga, U. L. (2022, January 6). *Time Series forecasting with supervised machine learning*. Medium. Retrieved August 12, 2022, from https://towardsdatascience.com/time-series-forecasting-with-machine-learning-b3072a5b44ba