S&P500 trading volume data prediction

Katerina Vankova

June 25, 2023

1 Introduction

The aim of the task is to explore S&P stock index volume data, suggest and analyze various models to estimate $\mathbb{E}[v_{d+1}|\mathcal{F}_d]$, where $d = 1, \ldots, n$ is a trading day, v_{d+1} is volume of the trades on a day $d+1$ and $\mathcal{F}_d$ is information available till the day d (inclusive). Our reference model is defined as $\hat{\mathbb{E}}[v_{d+1}] = v_d$. Finally, I compare the models' out-of-sample performance in regards to R^2 and SSE metrics.

Available data come from the time period: January, 1st 2000 till December, 31st 2018. In particular, I fit models recursively over 17-year period (starting with period from January, 1st 2000 till December, 31st 2016), and then I make one-day ahead predictions for a time period starting on January, 1st 2017 and extending through December, 31st 2018.

2 Data Exploration

The whole data set consists of 4778 observations, where each observation is a tuple of a trading date and corresponding volume. The training data set consists of 4522 observations (from January, 1st 2000 till December, 31st 2016 and then moving one day ahead).

The basic statistics describing the whole dataset are present in Table **??**. The average volume is 3.07 billion and the median volume is close to the average, totalling 3.17 billion. The volume range spans from 0.36 billion to 11.46 billion. The standard deviation is 1.50 billion. The skewness is 0.67, with kurtosis 0.79. Financial data are usually skewed and do not follow a normal distribution, which is the case here. Financial data tend to have heavy tails. It is caused by extreme events (e.g. financial crisis or other sudden and extreme events that cause large price movements).

min	**max**	**mean**	**median**	**std**	**skew**	**kurt**
0.36	11.46	3.07	3.17	1.50	0.67	0.79

Table 1: Basic statistics of the whole data set (in billions, 10^9).

To provide more insight into the data, I plot the time series of the whole data set and histogram of volume in Figure **??**. We can see that the data are not stationary, since the mean and variance change over time. Mean/variance changes is caused by increasing or decreasing magnitude of volume (that can be a response to changing stock prices, positive or negative financial reports of the companies whose stocks are part of the S&P index, or unexpected events such as financial crisis - the most impactful one was during $2007-2009$. We can see the significant increase in trading volume for this period in Figure **??**). Also, mean and variance of trading volume has changed since 2007 because of the increased use of electronic trading platforms (availability of high-frequency trading).

Furthermore, I will focus on the first train set (January, 1st 2000 - December, 31st 2016) in the data analysis, if not stated otherwise. We can also see that the data are skewed to the right, which is in accordance with the skewness value.

I also tested stationarity of the training data set using Augmented Dickey-Fuller test. It is a unit root test (it tests the existence of a unit root in a univariate process). Its null hypothesis is that there is a unit root, with the alternative that there is no unit root. I obtained its test statistic equal to -2.38 with $p-$value 0.14, therefore I cannot reject the null hypothesis on the level of $\alpha = 0.05$, i.e. I cannot reject that the time series is non-stationary. This is in accordance with the visual inspection of the time series.

I have also provided the autocorrelation function (ACF) and partial autocorrelation function (PACF) of the training data set in Table **??** for lags of 1, 20 and 250 trading days. Figure **??** shows

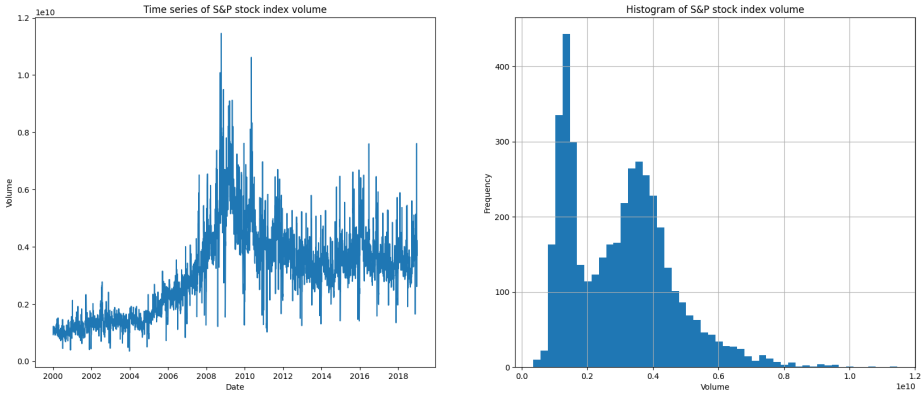

Figure 1: Time series and histogram of the whole data set.

ACF and PACF functions. Both functions assume that the series is stationary (which unfortunately may not be the case in our data, which can be seen also in the Figure **??**). The ACF is a measure of the correlation between the time series and a lagged version of itself. The PACF is similar to an ACF except that each partial correlation controls for any correlation between observations of a shorter lag length. I calculated ACF and PACF for lags of 1, 20 and 250 trading days. We can see that ACF decreases slowly, which indicates that the series is not stationary.

ACF(1)	**ACF(20)**	**ACF(250)**	**PACF(1)**	**PACF(20)**	**PACF(250)**
0.92	0.82	0.66	0.92	-0.02	0.08

Table 2: Basic statistics of the training data set (in billions, 10^9).

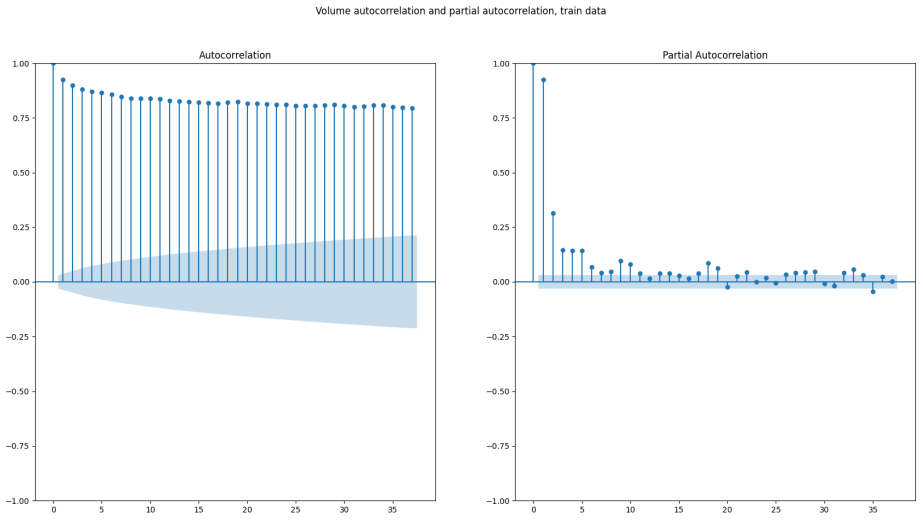

Figure 2: ACF and PACF of the training data set.

3 Modeling

Let $\hat{\mathbb{E}}[v_{d+1}] = v_d$ be our reference model, i.e. the prediction for the next day is the volume of the current day. I will compare the performance of the following models with respect to the reference model:

- Linear regression,
- Exponential smoothing (Holt, Winters),
- ARIMA,
- SARIMA,
- Temporal Fusion Transformer (TFT).

I will start with a simple linear regression. In this method, old data impact the fitted curve with the same weight as the current data.

I will therefore also explore an exponential smoothing method capturing seasonality that was introduced by Holt and Winters in 1960s. Forecasts produced by this method are weighted averages of past observations where the weights decay exponentially. The main idea behind using this model is that the most recent observations should have a higher impact on the current observations compared to observations months ago. It focuses on the estimation of the trend and seasonality component in the data.

Another approach for modelling time series is based on the description of the autocorrelations in the data, i.e. ARIMA models. These models assume stationary data, i.e. the properties of the time series do not depend on the observation time. In the real world, financial data are rarely stationary. Here is where differencing technique comes in. Specifically, I will explore SARIMA and a simpler ARIMA model.

The last explored model is a neural network based model, namely Temporal Fusion Transformer. It is a deep learning model that can be used for time series forecasting, and its architecture consists of both encoder and decoder parts. The encoder part is used to extract features from the input data, and the decoder part is used to generate the predictions.

Assumptions

For all the traditional time series models (all but TFT), we have the following assumptions on the residuals:

- the residuals are normally distributed, $\varepsilon_t \sim N(0, \sigma^2)\ \forall t$
- the appropriatness of the regression function, i.e. $\mathbb{E}(\varepsilon_t) = 0\ \forall t$
- the residuals are not autocorrelated, $cov(\varepsilon_t, \varepsilon_s) = 0\ \forall t, s : t \neq s$
- the residuals have constant variance, $var(\varepsilon_t) = \sigma^2\ \forall t$, i.e. we assume homoscedastic residuals

These assumptions are necessary for the validity of the statistical tests and confidence intervals for the model parameters. In the real world, these assumptions are rarely met. Therefore we should be cautious of the model predictions and confidence intervals. The predictions may not be accurate, the model parameters estimates may be biased as well as confidence intervals or statistical tests. Also, outliers may have a significant impact on the model performance. If the assumptions are violated and the model performance is poor, we should consider using a different model (non-parametric models such as lowess or GAMs), or suitable data transformations (e.g. Box-Cox transformation). The assumptions may be violated and still we can obtain a decent model, but its predictions may be less accurate. It then depends on the application whether we can use the model or choose a different approach.

3.1 Reference Model

The reference model is defined as $\hat{\mathbb{E}}[v_{d+1}] = v_d$. Evaluation of the model performance was made on the time period: January, 1st 2017 till December, 31st 2018. I have obtained residual sum of squares $SSE = 0.9 \times 10^{18}$, and $R^2 = 0.12$. Figure **??** shows the predictions of the reference model on the test data set. We can see that extreme values are not captured by the model properly. Predicting the previous value for the next day leads to a major increase in squared residual errors when the series is highly fluctuating. In the next subsection, I will try to improve the model performance by using more sophisticated methods.

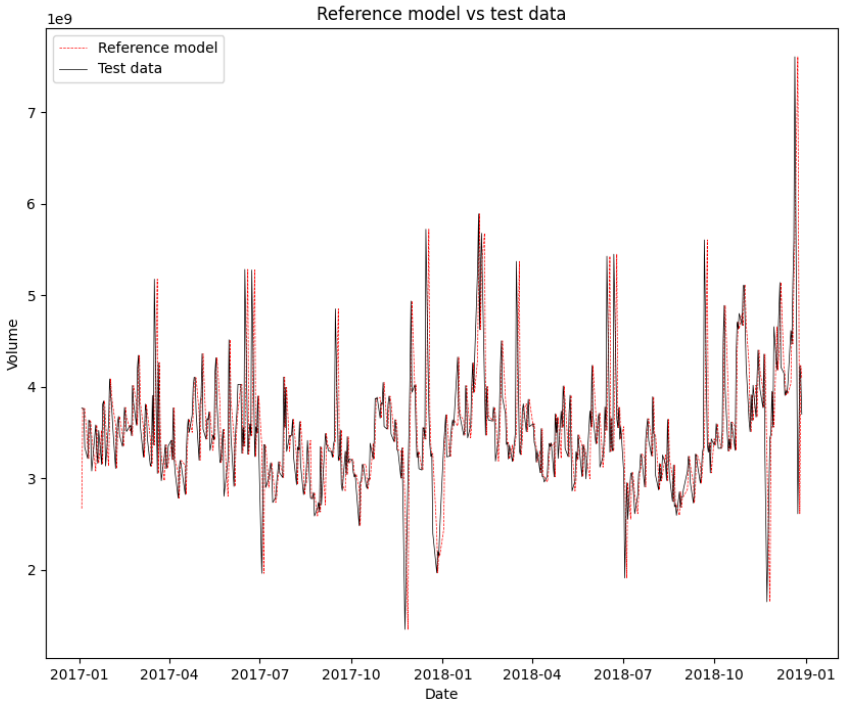

Figure 3: Predictions of the reference model on the test data set with observed data.

Table **??** shows basic statistics of the residuals. Figure **??** shows a scatter plot of residuals. We can see that residuals are scattered around 0, with outliers around 2 - 5 billion. These outliers drive the SSE up.

min	**max**	**mean**	**median**	**std**
−4 995.08	2 293.89	2.05	−2.70	608.97

Table 3: Basic statistics of the residuals of the reference model (in millions, 10^6).

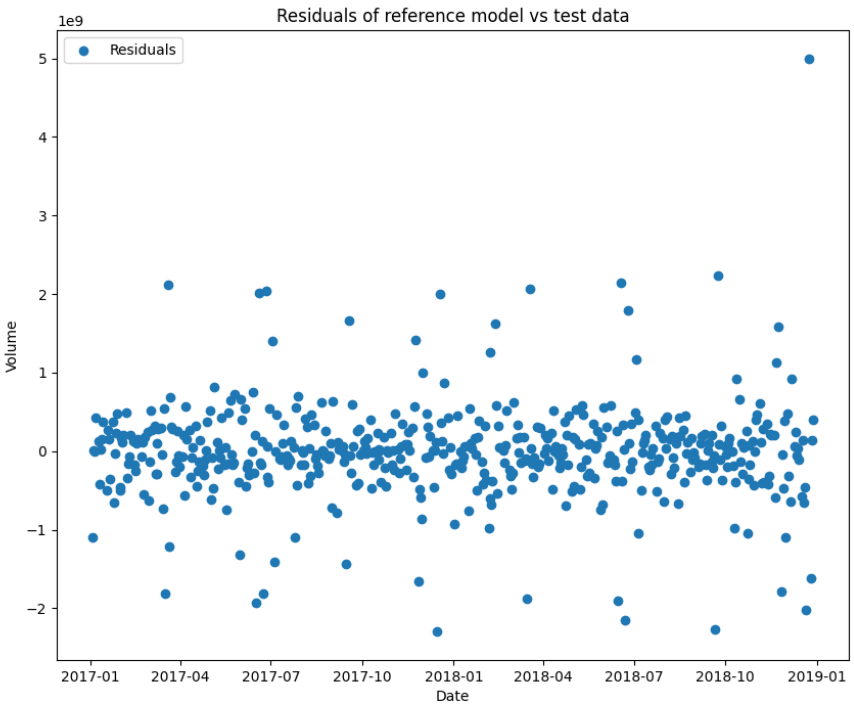

Figure 4: Scatter plot of residuals of the reference model.

3.2 Linear Regression

Linear regression assumes a linear model. In general, it is defined as

$$y_{t+1} = \beta_0 + \sum_{i=1}^{p} \beta_i x_i + \varepsilon_{t+1}, \tag{1}$$

where $\beta_0, \beta_1, \ldots, \beta_p$ are the regression coefficients, $x_1, \ldots, x_p$ are independent variables (features), and ε_{t+1} is the residual. The regression coefficients are estimated by minimizing the sum of squared residuals

$$SSE = \sum_{t=1}^{n-1} \varepsilon_{t+1}^2 = \sum_{t=1}^{n-1} (y_{t+1} - \beta_0 - \sum_{i=1}^{p} \beta_i x_i)^2. \tag{2}$$

Linear regression assumes that the residuals are normally distributed, homoscedastic and not correlated, the data are independent and identically distributed. Our time series data do not satisfy these assumptions (the variance increased over time). To handle the increased variance, a common practice is to log transform the data. Figure **??** shows the log-transformed volume of the data set. We can see that the variance is more or less constant over time. I also present boxplots of the log-transformed volume grouped by quarter and month in Figure **??**. These variables can be used as features.

Regarding quarters, we can see that the median of the log-transformed volume is the highest in the first quarter, and the lowest in the fourth quarter. Zooming into months, we can see that the median of the log-transformed volume is the highest in January, and the lowest in December. Reasoning behind this behavior may be that in January, there is year-end rebalancing of portfolios and the release of corporate earnings reports. On the other hand, December is the month where there is a lot of holiday days and therefore stock market is closed for those days. Also, in August there is a reduced trading volume which can be attributed to summer season when people are on vacation, and there is a lower news flow.

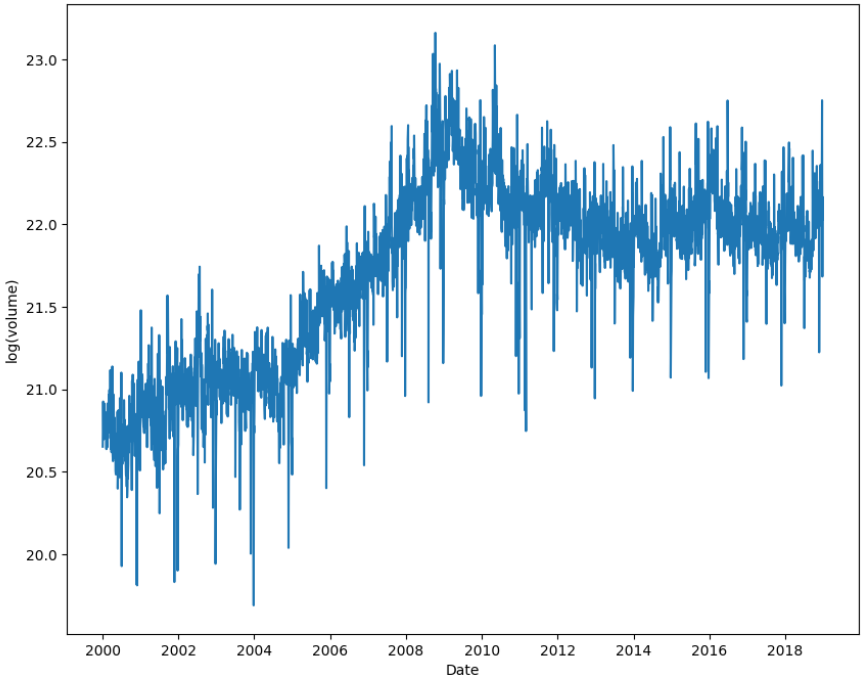

Figure 5: Log-transformed data.

Decomposing the log-transformed data into trend, seasonality and noise, I obtain Figure **??**. I chose additive decomposition since I log-transformed the data (it is equivalent to multiplicative decomposition of the original values). I will model the trend with a polynomial of degree three and a logarithm (based on experiments, this transformation works better than only a polynomial

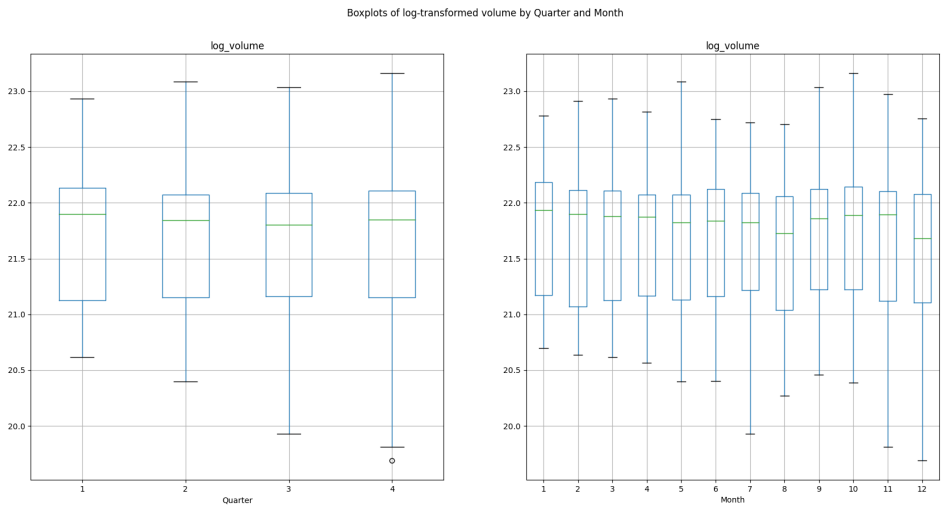

Figure 6: Boxplot of the log-transformed volume grouped by quarter.

or only a logarithm). Regarding the seasonality, I will consider quarter, month, week, day of the week and day of the month.

One of the assumptions of linear regression is independence of the features. Since quarter, month and week are correlated with each other, I will decide to keep in our set of features only month (based on the boxplots, it gave us enough information). I obtain the final list of features: time $t \in \{0, 1, \dots\}, t^2, t^3$, $\log(t)$, month $m \in \{1, 2, \dots, 12\}$, day of the week $d_w \in \{0 = \text{monday}, 1, \dots, 4 = \text{friday}\}$, and day of the month $d_m \in \{1, 2, \dots, 31\}$. Categorical variables m, d_w and d_m were one-hot encoded. The target variable is log-transformed volume.

I obtained $R^2 = -0.005$ and $SSE = 7 \times 10^{18}$. The model is not able to beat the reference model. I therefore considered incorporating previous values of the target variable as features (shift by 1 to 4 days). I obtained $R^2 = 0.42$ and $SSE = 0.6 \times 10^{18}$, which is a significant improvement compared to the reference model.[1] Figure **??** shows the predictions of the linear regression model on the test data set with confidence interval on the level of $\alpha = 0.05$. We can see that the model is not able to capture the extremes, since it assumes normality of the residuals.

Table **??** shows basic statistics of the residuals. Figure **??** shows residuals diagnostics plot (residuals vs date, autocorrelation, residuals vs fitted values and QQplot). We can see that residuals are scattered around 0, with outliers around 1 - 3 billion (in absolute values). We do not reject the hypothesis of not autocorrelated errors, since the autocorrelation plot does not show any significant autocorrelation (for lag 7 and 12 it is significant, but overall it is not). I do not reject the hypothesis of homoscedasticity based on the residuals vs fitted values figure, since it seems that residuals are distributed around 0 uniformely. We can see that the residuals are not normally distributed, since they do not follow the red line (the distribution has heavy tails).

min	**max**	**mean**	**median**	**std**
−2 686.048	3 141.85	14.58	61.15	509.20

Table 4: Basic statistics of the residuals of the linear regression model (in millions, 10^6).

[1] I have tried also lasso and ridge regression, but it had a negative impact on both metrics.

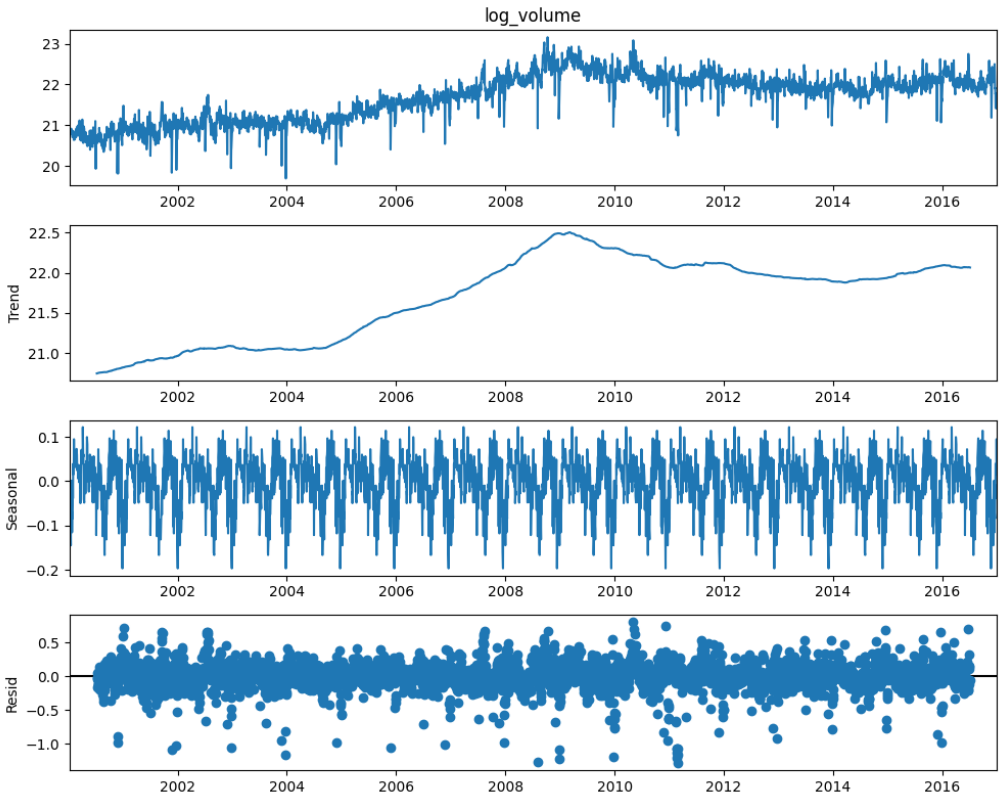

Figure 7: Additive decomposition of the log-transformed data.

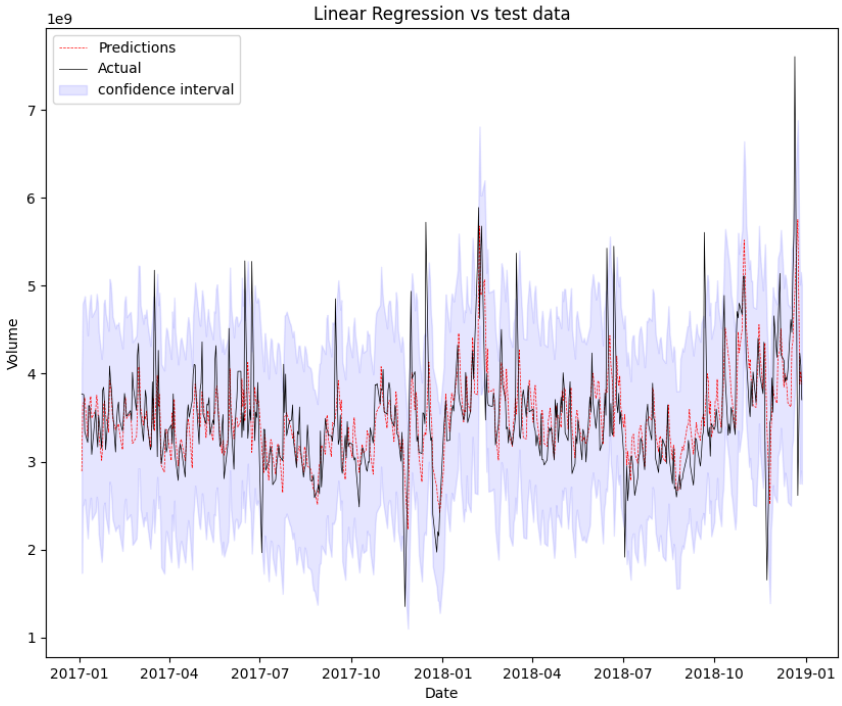

Figure 8: Predictions of the linear regression model on the test data set with observed data.

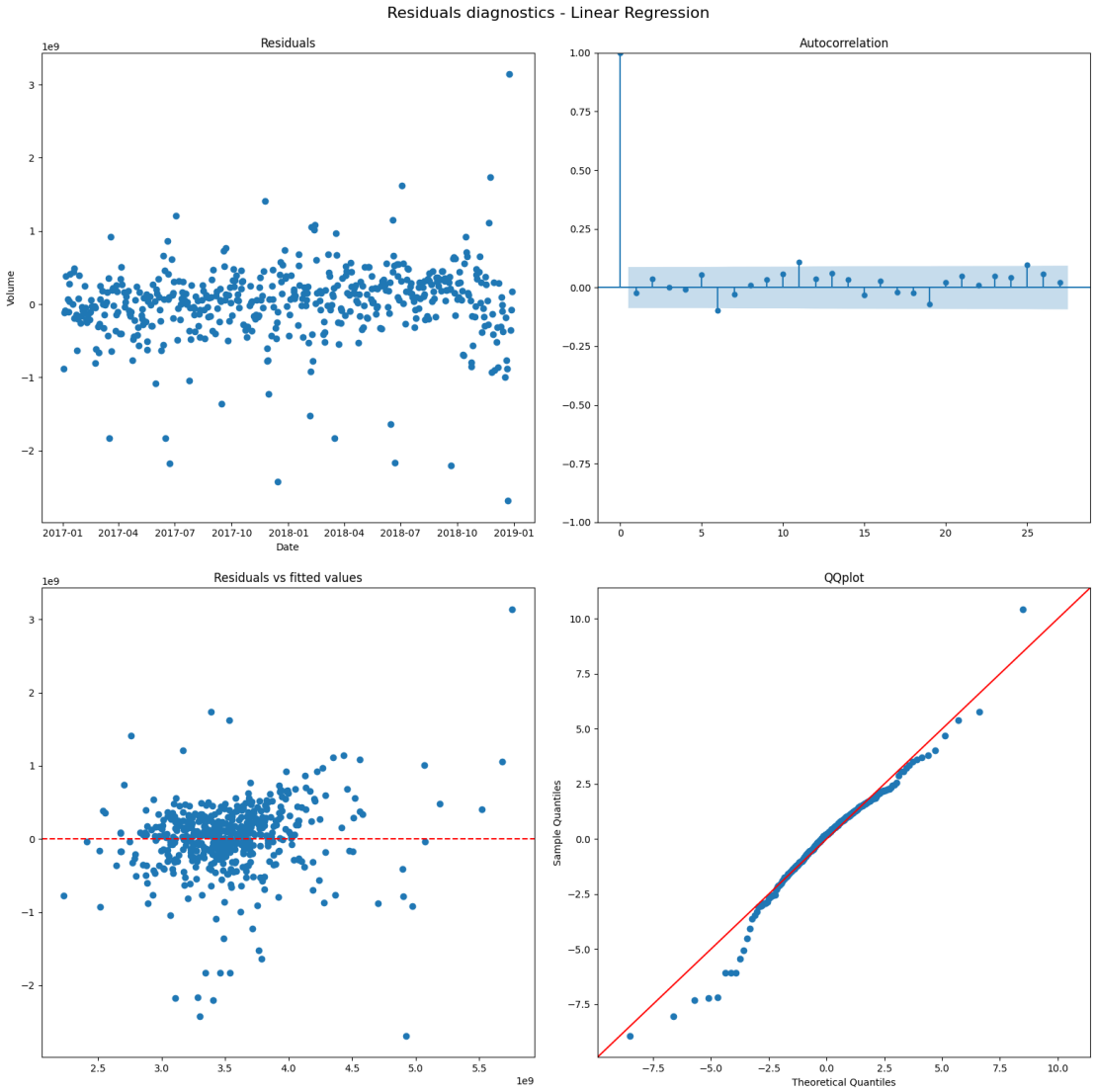

Figure 9: Residual diagnostics plot for linear regression model.

3.3 Exponential Smoothing

Next model is exponential smoothing model introduced by Holt and Winters. This model puts higher weight to the recent observations than to the older observations. The weights decrease exponentially as observations come from further in the past. The specific model I will use is called Holt-Winters seasonal method with additive trend and seasonal component. The model is described by the following equations:

$$\hat{y}_{t+h|t} = l_t + hb_t + s_{t-m+h_m^+}$$

$$l_t = \alpha(y_t - s_{t-m}) + (1-\alpha)(l_{t-1} + b_{t-1})$$

$$b_t = \beta(l_t - l_{t-1}) + (1-\beta)b_{t-1}$$

$$s_t = \gamma(y_t - l_{t-1} - b_{t-1}) + (1-\gamma)s_{t-m}$$

where l_t is the level component, b_t is the trend component, s_t is the seasonal component, α, β, γ are the smoothing parameters, m is the number of seasons, h is the forecast horizon (in our case, $h = 1$), h_m^+ is the seasonality component of the forecast horizon h.

I have analyzed simple (no trend nor seasonality), double (trend only) and triple (trend and seasonality) exponential smoothing. I have explored all combinations for trend and seasonality, i.e. both additive, both multiplicative, or one multiplicative and the other component additive. I have also considered dampened trend (i.e. trend component is dampened by a factor $\phi \in \{0.99, 0.98, 0.7, 0.5, 0.3, 0.1\}$). I have also considered the possibility of using Box-Cox transformation to stabilize the variance of the time series. I have also considered log transformed volume. I have arrived to the best model (in terms of R^2) using the following parameters: additive trend, additive seasonality, no dampening, no Box-Cox transformation. The model is described by the provided equations with parameters $h = 1$, $\alpha = 0.4646$, $\beta = 0.0001$, $\gamma = 0.0001$, $m = 250$.

The statistics of the model predictive performance are $R^2 = 0.23$ and $SSE = 146 \times 10^{18}$. This model performs worse than the linear regression model in one-day ahead forecasts, but it performs better compared to the reference model in terms of R^2.

Figure **??** shows model's predictions on the test data set with confidence interval. We can see that the model can quite well predict the trend and seasonality in the data, but rather lacks the ability to predict the extreme values.

Table **??** shows basic statistics of the residuals. Residual diagnostics plot is shown in Figure **??**. Outliers are scattered around 1-4 billion in absolute values, which drives the SSE up. We can see that the residuals are correlated with lags 3, 4 and 6, while for other lags the autocorrelation is not statistically significant. The residuals are uniformely scattered around zero, and their magnitude does not seem to depend on the fitted values, which is a good sign. Based on QQ plot, I would not reject the hypothesis of normality of the residuals.

min	**max**	**mean**	**median**	**std**
−2 578.83	3 722.83	−4.78	11.46	542.02

Table 5: Basic statistics of the residuals of the Holt-Winters model (in millions, 10^6).

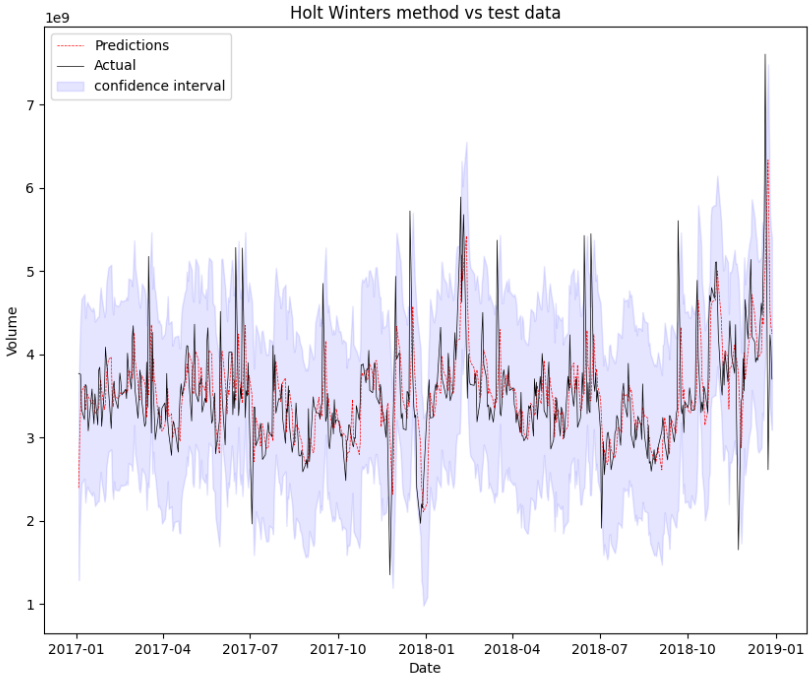

Figure 10: Holt-Winters predictions on the test data set.

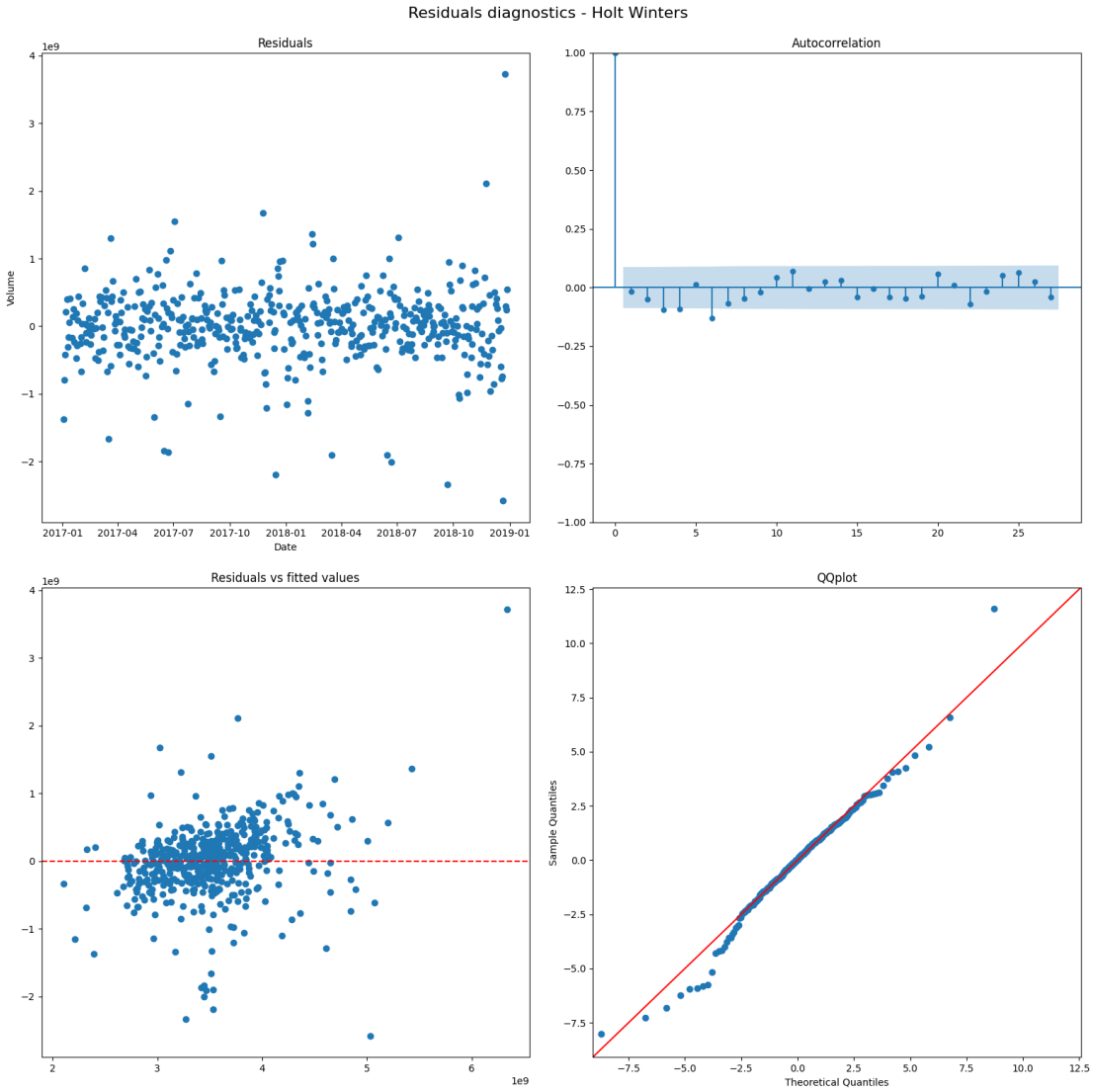

Figure 11: Residual diagnostics plot for Holt-Winters model.

3.4 SARIMA

ARIMA models are another common approach to time series modelling and forecasting. ARIMA models aim to describe the autocorrelation in the data, while exponential smoothing models aim to describe the trend and seasonality in the data.

ARIMA models are a generalization of ARMA models, where the time series is differenced d times to make it stationary. The model is then denoted as ARIMA(p, d, q), where p is the order of the autoregressive part, d is the number of differences, and q is the order of the moving average part.

We will focus on seasonal ARIMA models, denoted as SARIMA$(p, d, q)(P, D, Q)_m$, where m is the number of periods in each season, P is the order of the seasonal autoregressive part, D is the number of seasonal differences, and Q is the order of the seasonal moving average part. The model is then defined as:

$$\phi_p(B)\Phi_P(B^m)\nabla^d\nabla_m^D y_t = \theta_q(B)\Theta_Q(B^m)\epsilon_t \tag{3}$$

where $\phi_p(B)$ and $\theta_q(B)$ are the autoregressive and moving average polynomials, respectively, $\Phi_P(B^m)$ and $\Theta_Q(B^m)$ are the seasonal autoregressive and moving average polynomials, respectively, $\nabla^d\nabla_m^D$ is the seasonal difference operator, and ϵ_t is the white noise process. The model is then fitted using maximum likelihood estimation.

Parameters p, and q are determined by the autocorrelation and partial autocorrelation functions of the time series, while P and Q are determined by the seasonal autocorrelation and partial autocorrelation functions of the time series. Parameter d is determined by the number of differences needed to make the time series stationary, while D is determined by the number of seasonal differences needed to make the time series stationary.

The original time series is not stationary, therefore I consider first differences (and higher if it is needed). Figure **??** shows first differences of volume and log-transformed volume. We can see that the volume time series is not stationary, while log-transformed volume seems to have a steadier variance. Therefore I chose to use log-transformed volume in the SARIMA model. There seems to be rather no seasonality in the data, therefore I will also consider a simple ARIMA model in the experiments. For SARIMA model, zooming to the data, I have chosen seasonality $m = 125$ trading days.

Figure **??** shows ACF and PACF of the 1st differences of log-transformed volume time series (top row is without seasonal differencing, bottom row is with seasonal differencing). Based on these figures, I have chosen the following models to analyze:

- SARIMA: all combinations of $(p, d, q)(P, D, Q)_m$, where $p \in \{0, 1, 2, 3\}$, $d = 1$, $q \in \{0, 1, 2, 3\}$, $P \in \{0, 1, 2, 3\}$, $D = 1$, $Q \in \{0, 1, 2, 3\}$, $m = 125$
- ARIMA: all combinations of (p, d, q), where $p \in \{0, 1, 2, 3\}$, $d = 1$, $q \in \{0, 1, 2, 3\}$, $P = 0$, $D = 0$, $Q = 0$

I chose the best model from SARIMA and ARIMA class based on the AIC criterion. The best SARIMA model obtained $AIC = -109.24$ and is denoted as SARIMA(3, 1, 3)$(1, 1, 3)_{125}$. The best ARIMA model obtained $AIC = -3\ 578.23$ and is denoted as ARIMA(3, 1, 2). I therefore chose the ARIMA model as the final candidate for test data predictions.

Figure **??** shows the test data predictions of the ARIMA model and the actual data. We can see that the model predicts the data quite well. It obtained $R^2 = 0.30$ and $SSE = 134.19 \times 10^{18}$. ARIMA model outperformes Holt-Winters model in terms of both R^2 and SSE. The linear Regression is still the best model in terms of R^2 and SSE.

Table **??** shows the basic statistics of the residuals of the ARIMA model and figure **??** shows the residuals diagnostics plots. The residuals are scattered around zero, they are not correlated and assumption of normality seems to be satisfied.

min	**max**	**mean**	**median**	**std**
−2 777.85	3 085.82	−42.05	9.43	516.34

Table 6: Basic statistics of the residuals of ARIMA model (in millions, 10^6).

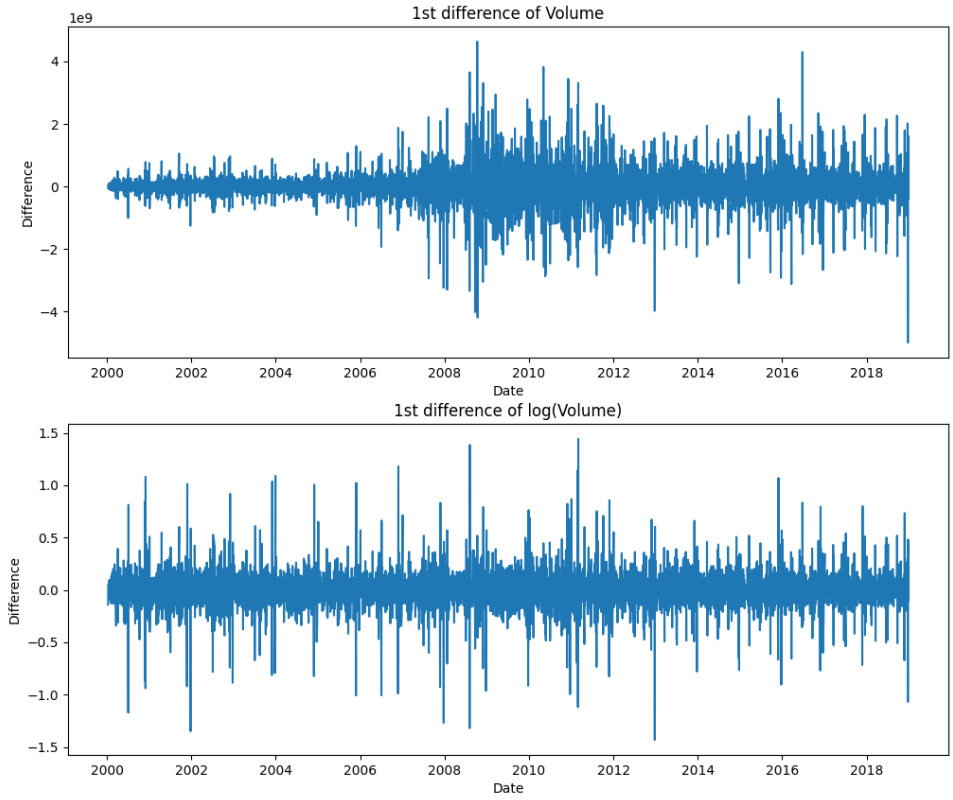

Figure 12: First differences of volume and log-transformed volume.

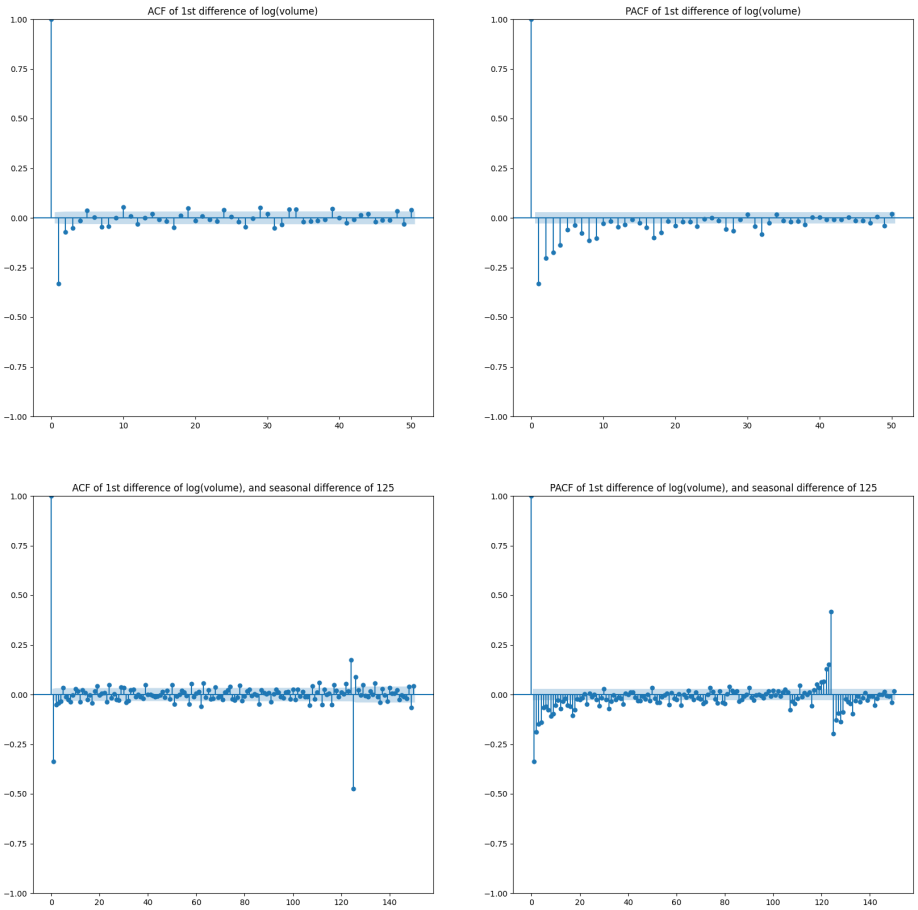

Figure 13: ACF and PACF of the 1st differences of log-transformed volume time series (with and without seasonal differencing).

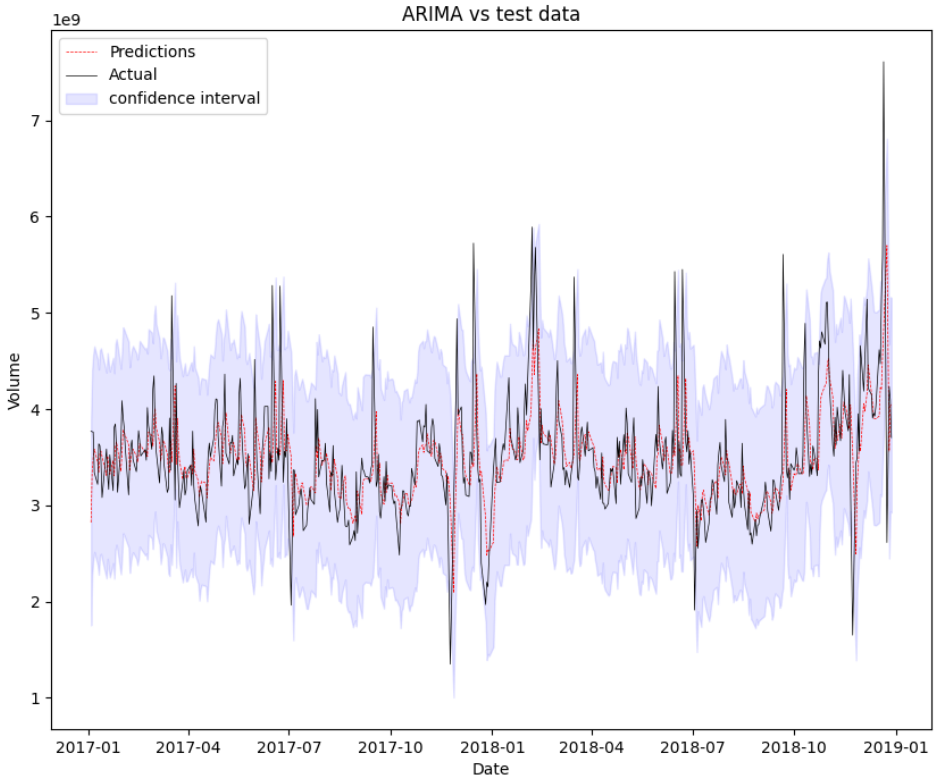

Figure 14: Predictions of the ARIMA model on the test data set.

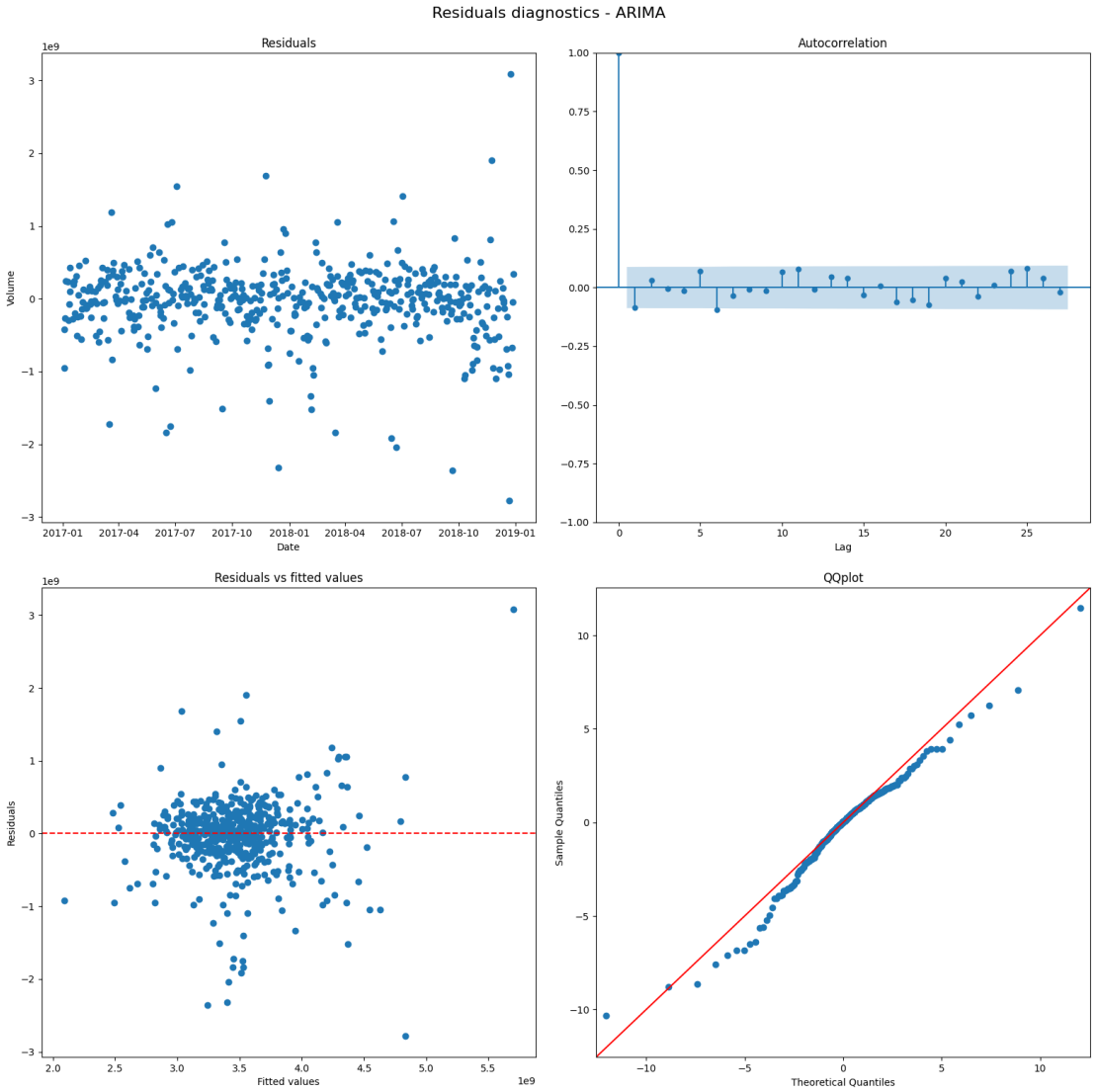

Figure 15: Residuals diagnostics plots of the ARIMA model.

3.5 Temporal Fusion Transformer

Finally, I decided to explore Temporal Fusion Transformer (hereinafter as TFT). TFT model is a deep learning model that can be used for multi-horizon time series forecasting. It was introduced in the paper *Temporal Fusion Transformers for Interpretable Multi-horizon Time Series Forecasting* (https://arxiv.org/pdf/1912.09363.pdf). It is a encoder-decoder model with attention-based deep neural network architecture. The encoder is used to extract features from the input time series, while the decoder is used to generate the predictions. It combines several concepts, such as self-attention layers, recurrent layers or gating layers. The authors also highlighted that TFT model enables three interpretability use cases (what the important variables are, what are the persistent temporal patterns and what are the significant events). More on this can be found in the paper. It also mentions that deep neural networks demonstrated strong performance improvements over traditional time series models. Another advantage of TFT model is that it produces quantile predictions, which can be used to construct prediction intervals. Though, Transformer models require a lot of data to train. However, our train dataset contains only 4 277 data points, which may not lead to strong performance. Nevertheless, I decided to try this model and see how it performs.

I used the implementation of TFT model from the *Darts* library. I kept default parameters of the model, except for:

- `input_chunk_length` $\in \{16, 32, 64, 128, 256, 512\}$
- `output_chunk_length` $\in \{1, 8, 16, 32, 64, 256, 512\}$
- `hidden_size` $\in \{16, 32, 64, 128, 256\}$
- `hidden_continuous_size` $\in \{8, 64\}$
- `lstm_layers` $\in \{1, 2\}$
- `num_attention_heads` $\in \{1, 2, 3\}$
- `dropout` $\in \{0, 0.05, 0.1\}$
- `batch_size` $\in \{16, 32, 64\}$
- `n_epochs` $\in \{10, 20, 30\}$

Considered features were year, quarter, month, week, day of week, day of month and time. The training was different compared to previous models. Previously, I recursively predicted the next value of the time series. In this case, I predicted the whole test data set at once. Training a neural network recursively and predicting one step ahead prediction at a time would be time-consuming and computationally expensive.

The best model obtained $R^2 = 0.05$ and $SSE = 182 \times 10^{18}$. Its configuration was:

- `input_chunk_length` $= 32$
- `output_chunk_length` $= 8$
- `hidden_size` $= 64$
- `hidden_continuous_size` $= 64$
- `lstm_layers` $= 1$
- `num_attention_heads` $= 3$
- `dropout` $= 0$
- `batch_size` $= 16$
- `n_epochs` $= 30$

It is barely better than a simple average, but it is still worse than the reference model. The root cause for this poor performance is very likely the lack of data as well as insufficient amount of computing power. Figure **??** shows the predictions by TFT model and test data, with confidence intervals. The model lacks the ability to predict extremes.

Table **??** shows the basic statistics of the residuals of the TFT model and figure **??** shows the residuals diagnostics plots. We do not have any assumptions on residuals for TFT model, but I

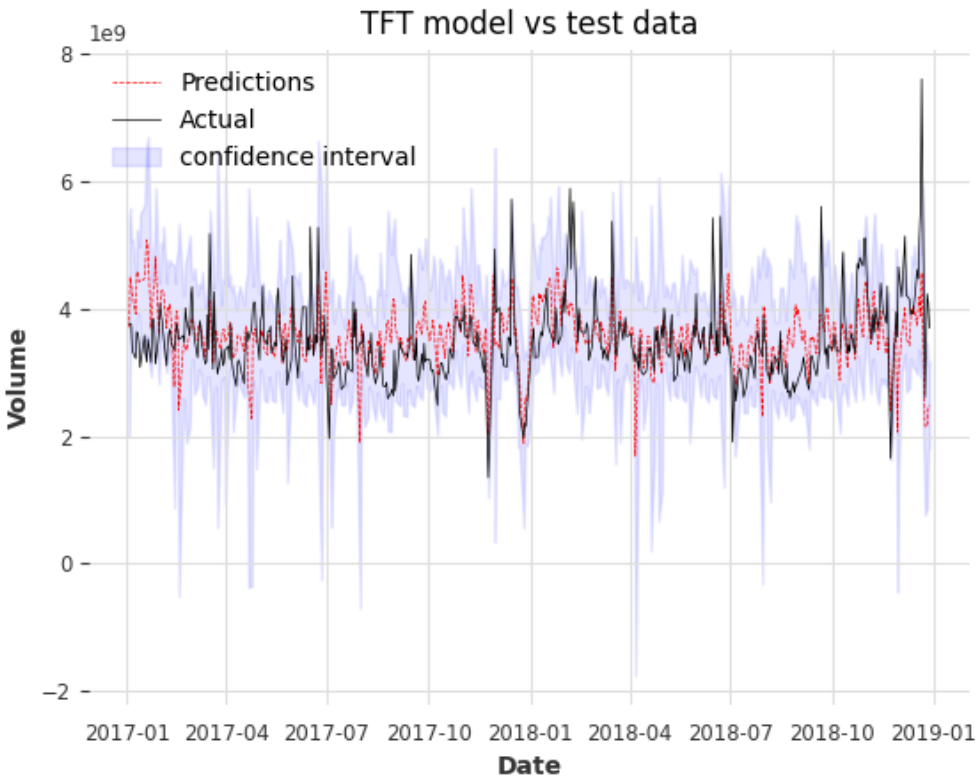

Figure 16: Predictions of the TFT model on the test data set.

present the same diagnostic plots to compare them with the rest of the models. Residuals are scattered around zero, with outliers mainly around 1 to 3 billion (in absolute values). There are more negative residuals lower than -1 billion (compared to higher than suggests that the model undercuts the actual values (we can see that also in the Figure **??**). The residuals seem to be correlated based on the ACF plot upto lag 6. Looking at the residuals vs fitted values plot, the residuals are scattered around zero and it seems as that with higher fitted values come higher residual (i.e. with fitted values of 4.5 billion and up, the prediction is further from the actual value with higher prediction, and vice versa for fitted values of 3 billion and down). Normality of these residuals is not rejected.

min	**max**	**mean**	**median**	**std**
−3 042.59	1 914.53	115.27	202.28	591.79

Table 7: Basic statistics of the residuals of TFT model (in millions, 10^6).

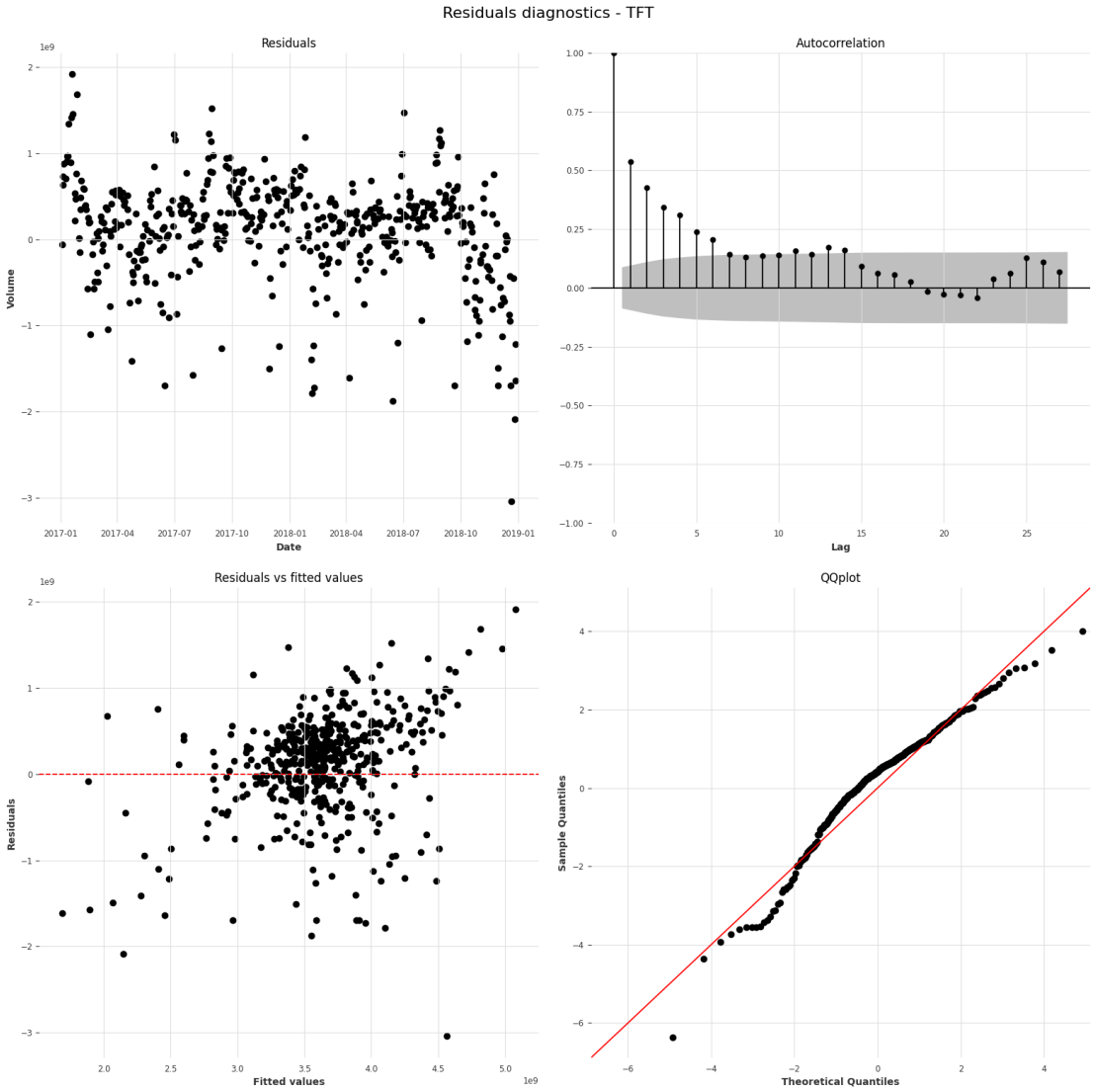

Figure 17: Residuals diagnostics plots of the TFT model.

4 Conclusion

In this homework, I have analyzed the time series of the daily trading volume of the S&P 500 index from 2000 to 2018. I have used several time series forecasting models to predict the future values of the time series in the period from 2017 to 2018. I have explored the following models: linear regression, Holt-Winters model, SARIMA model and ARIMA model. I have also used the Temporal Fusion Transformer model. I have evaluated the models based on the R^2 and SSE metrics. The reference model obtained $R^2 = 0.12$ and $SSE = 0.9 \times 10^{18}$. I was beaten by all models, but TFT model, in terms of R^2. In terms of SSE metric, it was beaten by only linear regression. The best model in terms of R^2 and SSE was the linear regression model ($R^2 = 0.42$, $SSE = 0.6 \times 10^{18}$). The ARIMA was the second best model in terms of R^2 ($R^2 = 0.30$, $SSE = 134 \times 10^{18}$) and the Holt-Winters model was the third one in terms of R^2 ($R^2 = 0.23$, $SSE = 146 \times 10^{18}$). The Temporal Fusion Transformer model performed the worst in terms of both metrics ($R^2 = 0.05$, $SSE = 181 \times 10^{18}$). Small training data set size was probably the root cause of the poor performance, since transformer models require a lot of data to train.

There are several areas for improvement. Firstly, I considered only a moving window of training data. It would be interesting to investigate the effect of different window lengths. Secondly, the dataset consisted only of target data. Incorporating also other features (not just time related) could improve the models' performance. Lastly, another model candidates could be explored, such as other deep learning models (e.g. LSTM, GRU, etc.), or different parameters for TFT model could be tweaked. Also, in case of using deep learning models, it would make sense to enlarge the dataset (e.g. by using data from other stock indices and by increasing the time horizon).