Thanh Nguyen

Capstone Three Final Report:
"Predicting Stock Returns"

1) Introduction

Predicting stock returns is exciting but extremely challenging. In this project, we imagine the situation where we are at the end of the trading day. Having observed the closing stock price and trading volume, we would like to predict the next day's stock return. That is to predict stock price change from now until the next day's market close.

If we could predict returns, we would make trading decisions accordingly (buy or sell, for example) and generate a high return for our portfolio.

We will use historical stock data (price and trading volume) from Nasdaq Data Link. The data are at daily frequency. We will implement both traditional time series methods (ARIMA) and machine learning approaches to forecast stock returns.

Finance theory suggests it is nearly impossible to forecast returns based on historical prices and trading volumes. And the more liquid the stock is, the harder it is to predict returns. Stock of a bigger company tends to be more liquid than that of a smaller company. In our project, we will look at stocks of two banks: the bigger bank Fifth Third Bank (FITB, 17.8 billion in market capitalization) and the smaller Zions Bank (ZION, 4.4 billion in market capitalization)[1].

2) Dataset

Nasdaq Data Link offers a vast number of databases. The platform aims to provide a unified source of trusted data and analytics. We take advantage of its free WIKI Prices database. This database contains stock prices, dividends, and splits for 3000 US publicly traded companies (Lewinson, 2022). The data, however, is available only until the end of March 2018.

[1] According to companiesmarketcap.com, FITB ranks 302 and ZION ranks 689 in the US by market capitalization.

Figure 1 shows a preview of data downloaded from Nasdaq Data Link. The data are in both original and adjusted values. Our goal is to predict the next day returns. Thus, the timeseries of interest is daily stock returns that we calculate using Adjusted Close Prices. In addition to the return time series, we engineer additional features to improve our prediction.

Figure 1: Preview of Data for ZION from Nasdaq Data Link

Date	Open	High	Low	Close	Volume	Ex-Dividend	Split Ratio	Adj. Open	Adj. High	Adj. Low	Adj. Close	Adj. Volume
2000-01-03	59.03	59.12	53.44	55.50	1199600.0	0.0	1.0	46.614284	46.685355	42.200023	43.826745	1199600.0
2000-01-04	54.63	55.00	52.50	52.81	816100.0	0.0	1.0	43.139731	43.431910	41.457732	41.702530	816100.0
2000-01-05	52.75	53.25	51.06	53.06	1124700.0	0.0	1.0	41.655150	42.049985	40.320606	41.899948	1124700.0
2000-01-06	52.75	54.94	52.38	53.50	1112100.0	0.0	1.0	41.655150	43.384529	41.362971	42.247403	1112100.0
2000-01-07	53.75	54.25	53.31	53.63	782000.0	0.0	1.0	42.444821	42.839656	42.097366	42.350060	782000.0

WIKI Prices database has superior data quality. There are no missing values. Thus, we do not have to do much data wrangling. We discuss the calculation of the target (stock returns) and features in the next section.

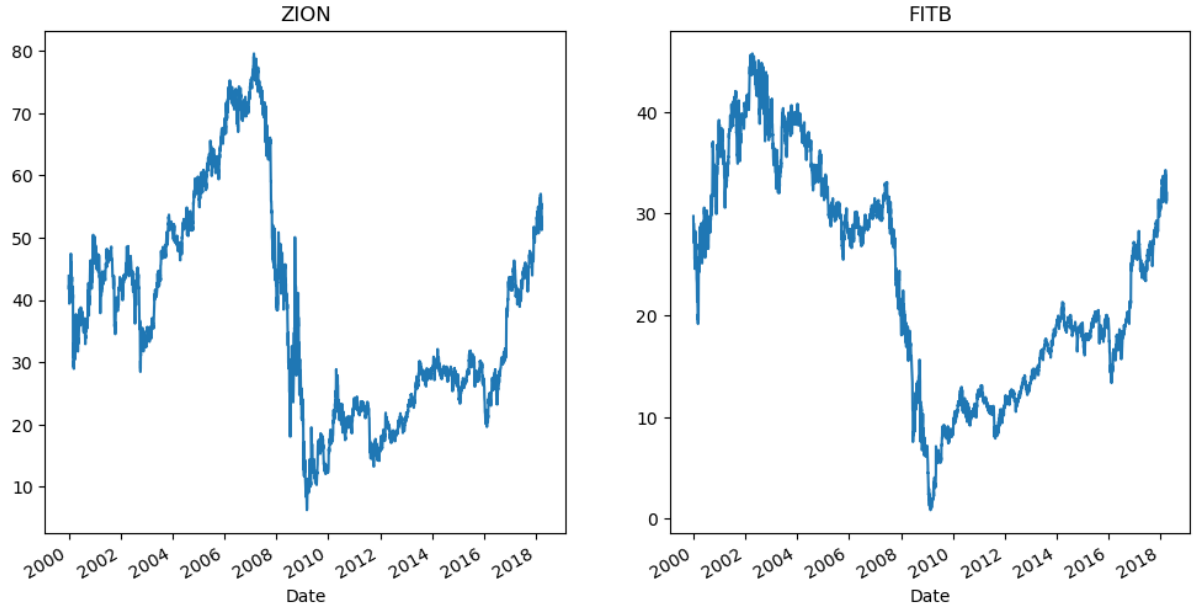
3) Exploratory data analysis

3.1) Stock returns

Figure 2 shows prices of our two stocks from 2000 to the end of March 2018. One can see the prices moved differently at the beginning of the period. ZION adjusted price peaked around 2007, probably like the time that broad market peaked before the great recession (late 2007 to middle 2009[2]), while FITB peaked around 2002. At the end of the period, since 2012, ZION and FITB prices seem to move together.

[2] The Great Recession | Federal Reserve History

Figure 2: ZION and FITB Prices



In time series forecasting, the length of the training period is a hyperparameter to be optimized. Long training periods tend to increase statistical power, but risk using old, might be less relevant data. We will consider a 5-year and 3-year training period. Thus, for the purpose of model training we use data from 2012 to 2017. All the statistics reported below are for this time period.

It is clear from Figure 2 that the stock prices are non-stationary. For our forecasting task, we will work with log returns which we calculate as following[3]:

$$log\_rtn_{\ t} = log\ (\frac{P_t}{P_{t-1}})$$

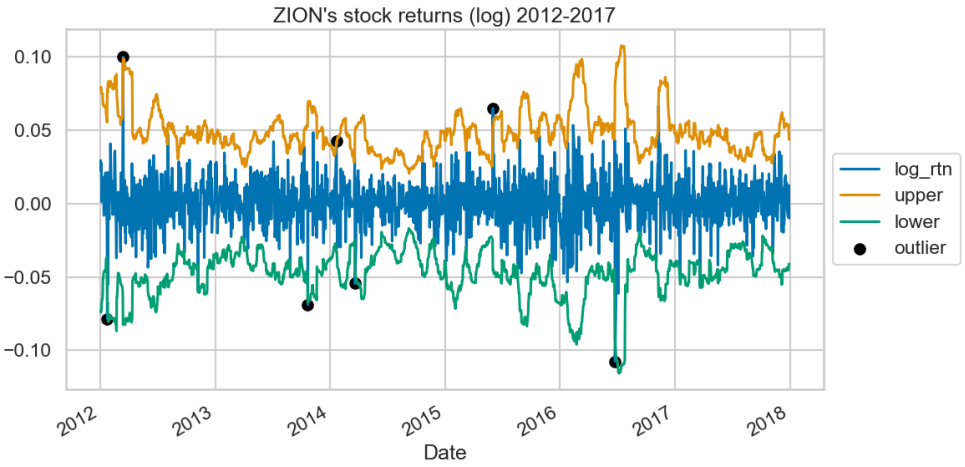
An alternative to log returns is simple returns calculated as $R_t = (P_t - P_{t-1})/P_{t-1}$. We prefer to use log returns for the following reason. Distribution of stock prices is close to log-normal and then the log returns are normally distributed. The normal distribution would work well with many statistical approaches to time series modeling (Lewinson, 2022).

[3] Our target feature is log return of the next trading day. Please see section 3.2 for more discussion.

Figure 3 shows log returns for both stocks together with upper and lower bounds for detecting outliers. The upper (lower) bounds are defined as three standard deviations above (below) the moving average. Standard deviations and moving averages are based on the 21-trading-day moving window.

There are two takeaways from figure 3. First, the returns seem to be stationary. Second, consistent with common understanding, stocks of smaller companies tend to be more volatile and to have more extreme values. ZION seems to have higher volatility and outliers than those of FITB. Indeed, the standard deviations of ZION and FITB are 0.0169 and 0.0147, respectively. It is not clear from the graph, but the correlation between the two stocks is 0.77, consistent with the fact that they are in the same industry.

Figure 3: ZION and FITB Returns



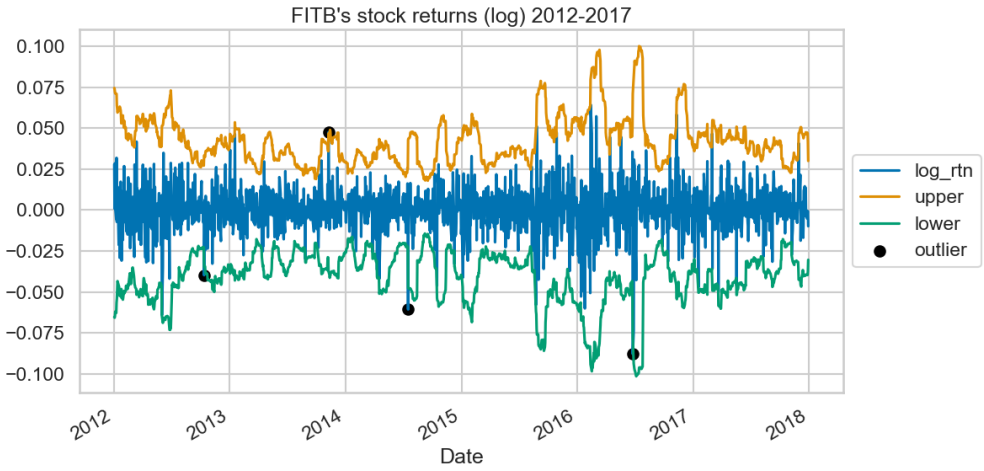
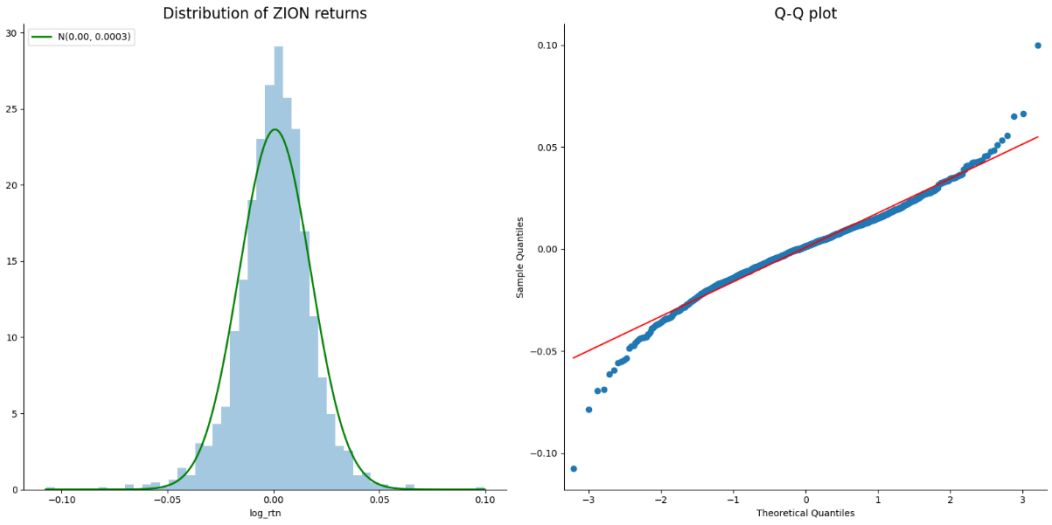
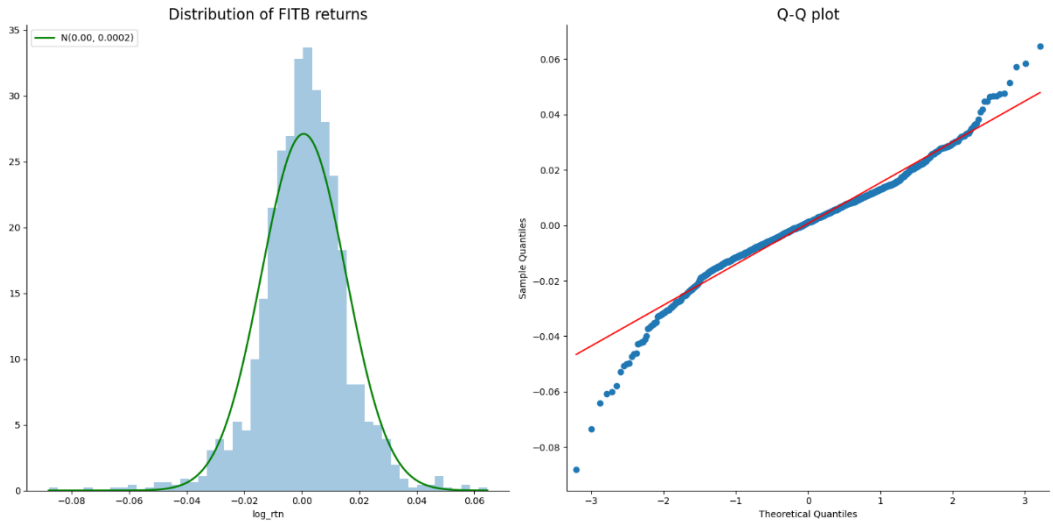


Figure 4 shows the distributions of returns. The return distributions are close to normal, except for the fat tails and negative skewness. The probabilities of having very low or very high returns are higher compared to the normal distribution and there are more very low returns than very high returns. The mean (median) of ZION returns is 0.00078 (0.0014). The number for FITB is 0.00067 (0.0012)

Figure 4: ZION and FITB Return Distribution

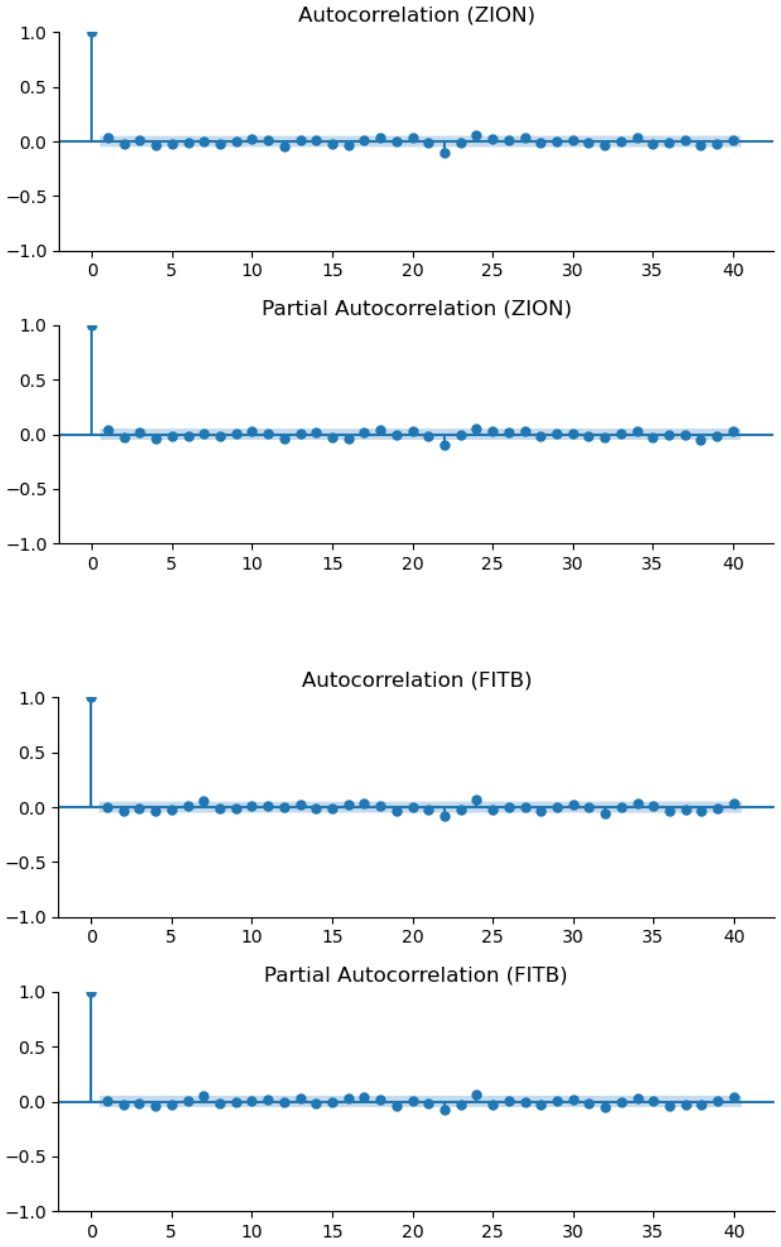




Next, we carry out official tests for stationarity. We implement two tests: Augmented Dickey-Fuller (ADF) and Kwiatkowski-Phillips-Schmidt-Shin (KPSS). Both tests confirm that the returns are stationary. The ADF test's null hypothesis is that the time series is not stationary; the test's p-values are much less than 0.001 for both stocks. The KPSS test's null hypothesis is that the time series is stationary; the test's p-values are higher than 0.1 for both stocks.

Figure 5 plots autocorrelation function (ACF) and partial autocorrelation functions (PACF) for both stocks. Consistent with stylized facts of asset returns, there is no autocorrelation in the returns. It is interesting that correlation at lag 22 is marginally significant for both ZION and FITB. It is unclear why this is but may merit further investigation.

Figure 5: Autocorrelation and Partial Autocorrelation Functions



3.2) Features

Our task is to forecast future returns. Traditional time series models such as ARIMA only use the past values of the time series to predict future values. In this section we will create additional features to be used in ARIMAX (ARIMA with explanatory variables) as well as in machine learning models. We engineer four groups of features.

The first group includes lags of return up to 14-day lag. The second group includes features based on trading volume change. We calculate one-day change in trading volume and 10-day moving average of the one-day change in trading volume.

The third group includes some technical analysis indicators. Traders have developed many technical analysis indicators that are believed to be able to predict stock returns. In our project, we choose two of them: moving average and relative strength index. We estimate these indicators for 14-day, 50-day, and 200-day periods.

Finally, the fourth group relates to the months of the year. The goal of these features is to capture seasonality in stock returns if the seasonality exists. We could use one-hot encoding, but we will lose the cyclical continuity of months. Thus, we implement cyclical encoding using sine and cosine transformation.

3.3) Target and its correlation with features

As mentioned before, we imagine the situation where we are at the end of the trading day. Having observed today's closing stock price and trading volume (together with all historical price, volume, and other public information), we would like to predict next day stock return; this is our target (variable 1d_future_log_rtn).

Figure 6 shows correlation between target and some lag returns. One can see that there is no significant correlation between future and lag returns. Figure 6 shows only from lag8 to lag14, but there is also no significant correlation when looking at other lags (lag0 to lag7) and the volume-based features[4]. The absence of correlation is consistent with the argument that if there is any significant correlation between future returns and historical data, the correlation is self-destroying. Traders will take advantage of the correlation, and the correlation will disappear as the result of trading activity.

[4] Heatmaps are not reported for the sake of brevity.

Figure 6: Correlation of future return and lag returns

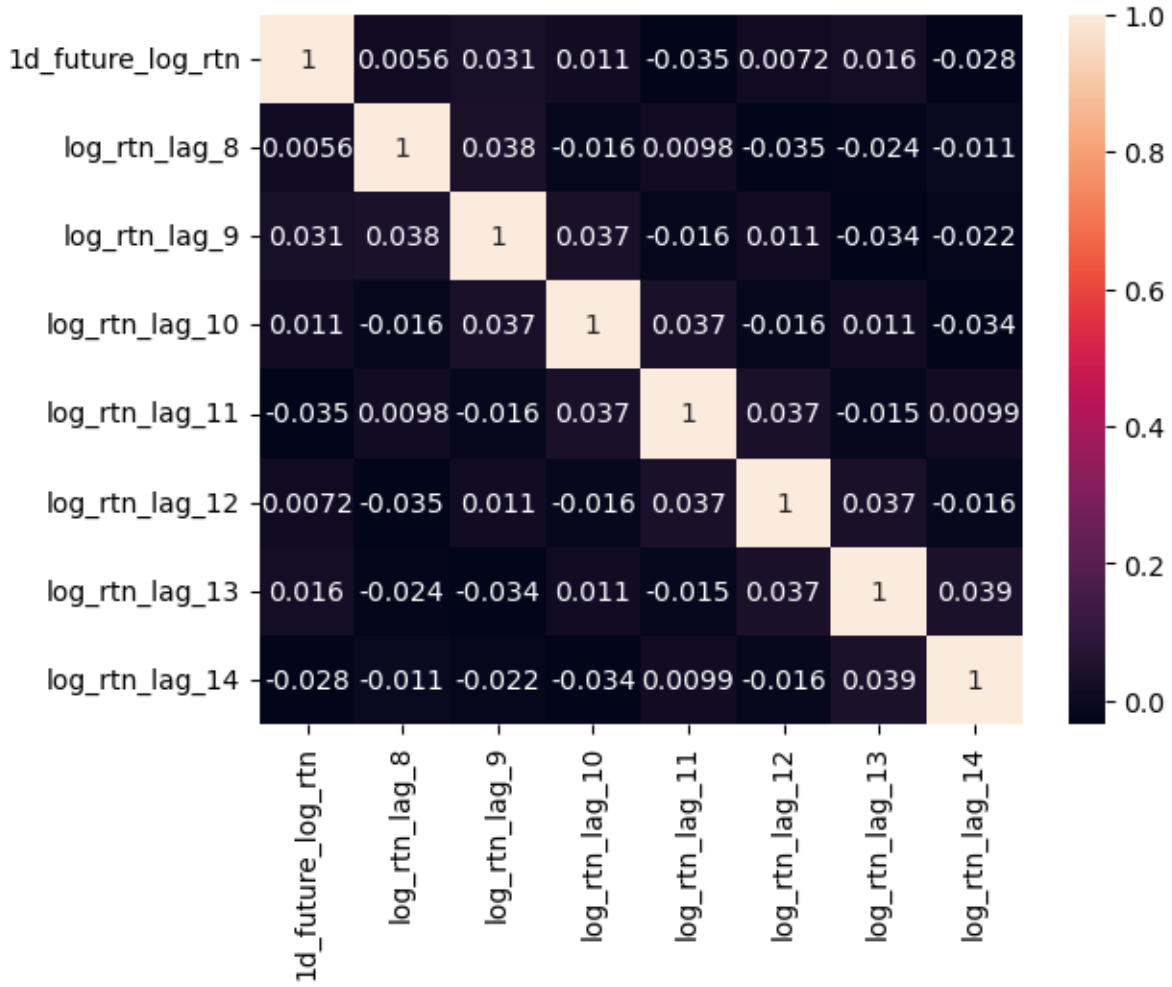
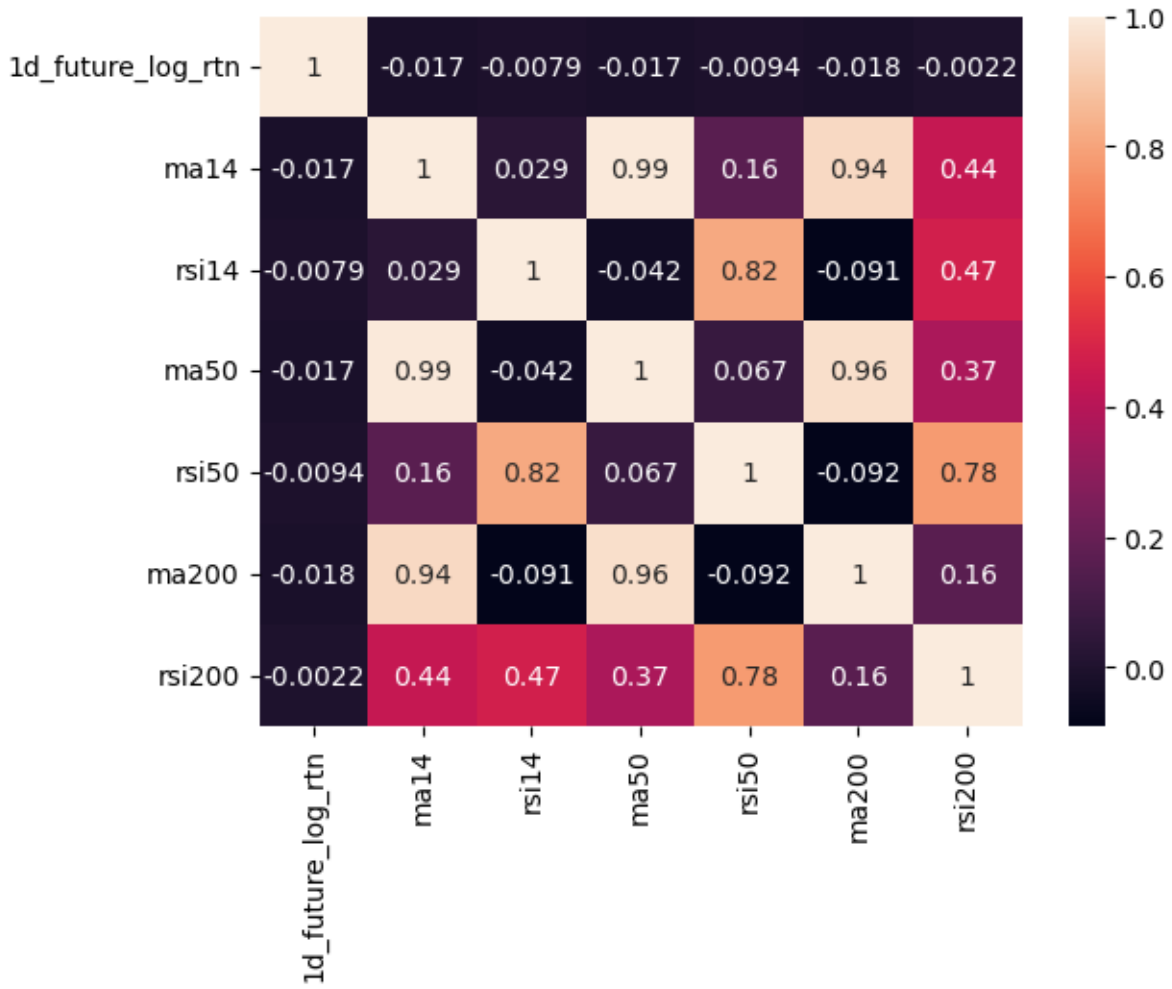


Figure 7 shows correlation between target and technical analysis indicators. The technical analysis indicators have no correlation with returns, but they have high correlation among themselves. For example, correlations among 14-day, 50-day, and 200-day moving averages are 0.99, 0.92 and 0.94. The correlations among three relative strength indicators are 0.81, 0.44, and 0.77. Thus, to avoid multicollinearity, in our regression, we will use only 50-day moving average and relative strength indicators.

Figure 7: Correlation of future return and technical analysis indicators



4) Modeling

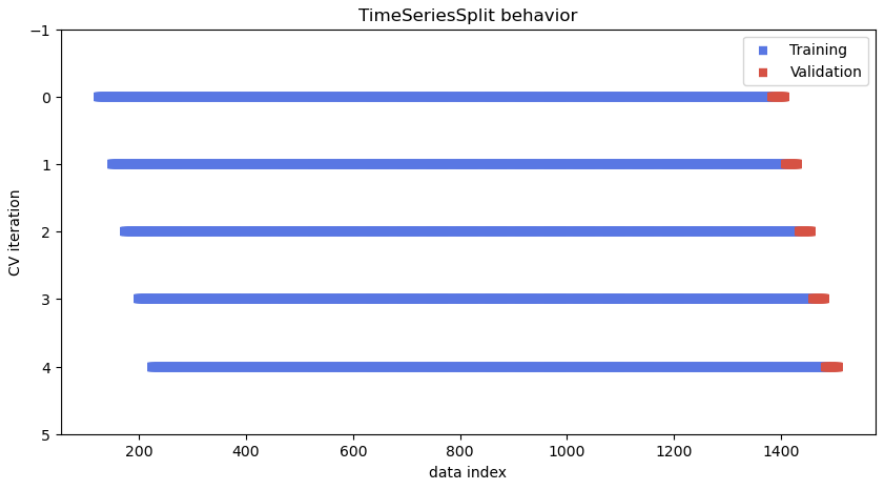
To evaluate models' performance, we use a cross-validation approach. The traditional k-fold cross-validation is not advised to use when dealing with time series models since it does not preserve time order. We use sliding window TimeSeriesSplit() cross-validation.

Figure 8 shows the train and validation sets that we use. Our data include six years from 2012 to 2017. However, for each split the train set includes only 5 years of data (1260 trading days). We also train our models using 3 years (756 trading days) of data. The validation set, as shown in the

figure, includes 25 trading days after train set[5]. The last day of the validation set of the last split is the last trading day of 2017.

Our test set is the first 25 trading days of 2018[6]. We choose the best forecasting model based on cross-validation performance. To evaluate the chosen model on the test set, we retrain it using 5 years of data (1260 trading days) up to the end of 2017 and report its performance comparing its forecasts with the test set.

Figure 8: Visualizing TimeSeriesSplit() behavior



The performance measure we use is Mean Squared Error (MSE). The other popular performance measure is Mean Absolute Percentage Error (MAPE). But since we are working with daily stock returns, the returns can be zero or very close to zero. MAPE might be undefined (divide to zero error) or very large (when denominators are very close to zero). This is why we prefer MSE.

Table 1 shows cross-validation results for the traditional time series model (ARIMA) and different machine learning algorithms predicting ZION returns. MSEs are very similar among the

[5] 25 trading days is about (a little more than) one trading month. What we have in mind is the chosen model will be re-trained monthly and used to predict returns for the following month. The exception is ARIMA models that will be re-trained every day.

[6] Nasdaq Data Link's WIKI Prices database data end at the end of March 2018. We could use the last 25 trading days of this data as the test set. We choose the first 25 trading days of 2018 for the ease of coding.

models. Gradient Boosting has the smallest MSE of 0.000171. Then Random Forest (5- and 3-year training period), Ridge (with and without month sine and cosine), and ARIMA all have MSE of 0.000172. KNN has the biggest MSE.

Interesting fact about KNN is that the model with month sine and cosine performs much better than the model without these variables. MSE with (without) month sine and cosine is 0.000177 (0.000185). KNN seems to be able to capture some seasonality in stock returns.

Table 1: Model training for predicting ZION returns

Model Name	**Length of train period**	**Best Parameters**	**Cross-Validation (mean squared error)**
Random Forest	5 years	'max_depth': 1 'max_features': 0.1 'n_estimators': 500	0.000172
Random Forest	3 years	'max_depth': 1 'max_features': 0.3 'n_estimators': 100	0.000172
Gradient Boosting	5 years	'learning_rate': 0.005 'max_depth': 6 'max_features': 0.1 'min_samples_split': 8 'n_estimators': 50 'subsample': 0.8	0.000171
Ridge	5 years (data are scaled)	alpha: 7250	0.000172
Ridge	5 years (with month sine and cosine, data are scaled)	alpha: 17000	0.000172
KNN	5 years (data are scaled)	n_neighbors: 14	0.000185
KNN	5 years (with month sine and cosine, data are scaled)	n_neighbors: 16	0.000177
ARIMA	5 years	(4, 1, 3)	0.000172

Table 2 shows cross-validation results for models predicting FITB returns. Again, MSEs are very similar among the models. KNN (with month sine and cosine) and Random Forest (5-year training period) have the smallest MSE (0.000128). Gradient Boosting has a little greater MSE (0.000129).

Table 2: Model training for predicting FITB returns

Model Name	**Length of train period**	**Best Parameters**	**Cross-Validation (mean squared error)**
Random Forest	5 years	'max_depth': 20 'max_features': 0.1 'n_estimators': 100	0.000128
Random Forest	3 years	'max_depth': 1 'max_features': 0.1 'n_estimators': 500	0.000130
Gradient Boosting	5 years	'learning_rate': 0.08 'max_depth': 3 'max_features': 0.1 'min_samples_split': 6 'n_estimators': 50 'subsample': 1	0.000129
Ridge	5 years (data are scaled)	'ridge__alpha': 40000.0	0.000130
KNN	5 years (data are scaled)	n_neighbors: 11	0.000132
KNN	5 years (with month sine and cosine, data are scaled)	n_neighbors: 6	0.000128
ARIMA	5 years	(1, 0, 2)	0.000131

5) Performance on test set

Table 3 shows our prediction performance on unseen data, the test set. Figure 9 and 10 plot actual returns, predicted returns, and historical average return for ZION and FITB, respectively.

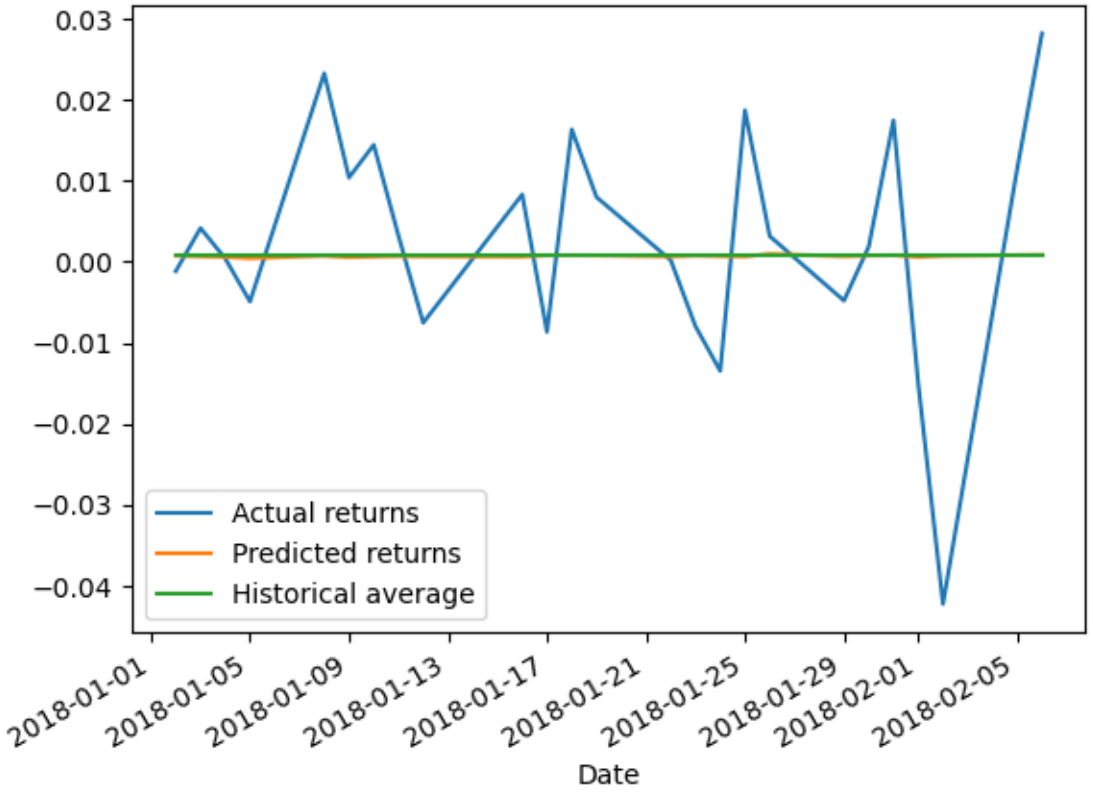
The best model for predicting ZION returns is Gradient Boosting. When we use this model to predict returns for the test set the MSE is 0.000209 and r_square of -0.0135. It is useful to compare our model performance with a naïve model where we use historical average return (5-year average

return) to predict returns for the test period. The test MSE and r-square of the naïve model are 0.000209 and -0.0160[7]. One can see that our best model does not perform much better than the naïve model.

Table 3: Models' performance on unseen data

	Best model performance		Naïve model performance	
	MSE	R-square	MSE	R-square
Predicting ZION returns	0.000209	-1.35%	0.000209	-1.60%
Predicting FITB returns	0.000147	1.28%	0.000152	-2.12%

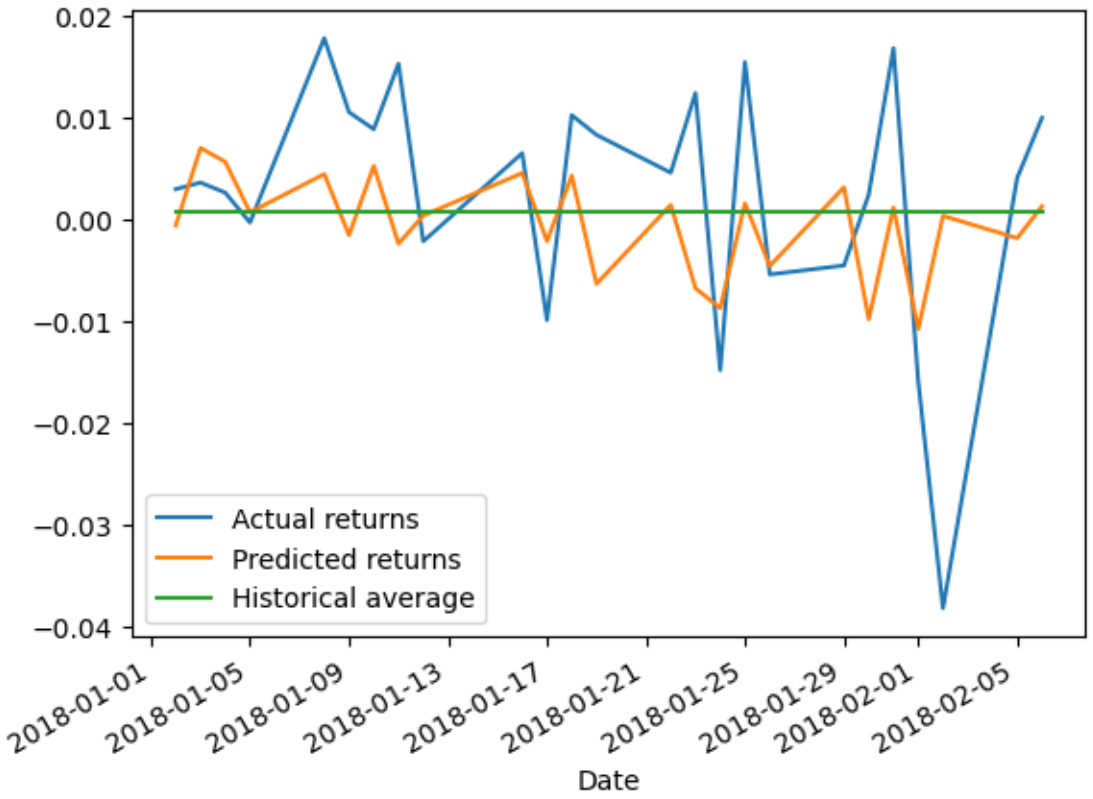
Figure 9: Actual returns, predicted returns and historical average returns (ZION)



[7] For ZION, MSEs of our model and naïve model are 0.000209 after rounding. The best model MSE is 0.0002087. This is a little smaller than MSE of the naïve model (0.0002092).

To predict FITB returns, Random Forest and KNN are the best models. For Random Forest, we have a concern that the parameter "max_depth" being 20 is too high. This might cause overfitting. Thus, we choose KNN with month sine and cosine as our final best model. KNN's test MSE (r-square) is 0.000147 (0.0128). Performance of the naïve model is 0.000152 (-0.0212). For FITB, our best model achieves some improvement over the naïve model.

Figure 10: Actual returns, predicted returns and historical average return (FITB)



There are interesting observations in figure 9 and 10. For ZION, our predicted returns are practically the same as the prediction of the naïve model. When stock price follows a random walk process then the best forecast about returns is the historical average; and our Gradient Boosting model produces the same forecast. On the other hand, for FITB, KNN model does a good job predicting direction of the returns. One can see that when the actual returns are above (below) the historical average, our prediction tends to be above (below) the average as well.

6) Conclusion and future work

We achieve improvement in predicting stock returns compared to the naïve model. The improvement is clearer when comparing r-squares. KNN models show potential to predict direction of price movement, either to go up or down. However, we believe our achieved improvement is not enough to justify the use of our models in assisting trading and portfolio management activities. In addition, we do not see big difference in our predicting performance between ZION and FITB. Might be that these two stocks have similar liquidity, and we should have chosen two stocks with much more difference in market capitalization.

Our result is consistent with the finance theory that it is very challenging or impossible to predict stock returns just based on the historical price and trading volume data. Even when using only historical price and trading volume data, our models are still very simple. We could engineer more features (for example other technical analysis indicators) to capture the temporal variation of the data.

There is work we could do to improve our predicting performance. First, we can try ARIMA model with seasonality and exogenous features. For seasonality, we need to find out how to specify seasonality for daily trading data where numbers of trading days in weeks, months, years are not the same over time.

Second, we can try to tune Random Forest models better to find the best hyperparameters that are less likely to cause overfitting. Third, we can train a Neural network model. Specifically, Recurrent Neural Networks (RNN). This model is designed to work with sequence data and, thus, is suitable to analyze and predict time series data such as stock returns.

References:

Lewinson, E. 2022. *Python for Finance Cookbook*. <packt>.