UNIVERSITY OF
FORWARD
THINKING
WESTMINSTER

WestminsterResearch
http://www.westminster.ac.uk/westminsterresearch

Essay on Volatility Clusters and Time Series Prediction

Urumov, G. and Chountas, P.

This is a copy of the author's accepted version of a paper subsequently published in the proceedings of the IEEE Intelligent Systems 2022, Warsaw 12 - 14 Oct 2022, IEEE.

The final published version will be be available online at:

https://doi.org/10.1109/IS57118.2022.10019619

© 2022 IEEE . Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

The WestminsterResearch online digital archive at the University of Westminster aims to make the research output of the University available to a wider audience. Copyright and Moral Rights remain with the authors and/or copyright owners.

ESSAY ON VOLATILITY CLUSTERS AND TIME SERIES PREDICTION

Georgy Urumov
School of Computer Science and Engineering
University of Westminster
115 New Cavendish Street, London
w1767944@westminster.ac.uk

Panagiotis Chountas
School of Computer Science and Engineering
University of Westminster
115 New Cavendish Street, London
p.i.chountas@westminster.ac.uk

***Abstract*— Volatility moves in financial markets are rare and sporadic i.e., periods of low volatility typically follow each other until the regime change due to technical and/or fundamental exogenous factors. Subsequently higher volatility tends to lead to even higher volatility and vice versa. Such dynamic systems require alternative ways of representation than a static multi-layer perceptron (MLP). The prevalent models assume L-stable distributions with independent and stable increments. This contrasts with what is observed in real world. L-stable distributions miss one of the main features of financial markets – the alternation of periods of large price changes with periods of small price changes. To correct for this deficiency another, self-affine process had been introduced – Fractional Brownian Motion (FBM). FBM does not capture fat tails or fluctuations in volatility that are unrelated to the predictability of future returns. In summary, both models have strong scale-invariance property, in which the distribution of returns over different sampling intervals are identical. This property is clearly at odds with empirical observations.**

The key idea in this paper is that Generative Adversarial Networks (GAN) combined with fractality of data is better suited to manage volatility time series between different shock events because it is structured to maintain a memory of older points in the time series and continuously learn from them. Episodic memory that is used in GANs maintains explicit record of past events. In order to make decision, the action is chosen that has the highest value based on the outcomes of past similar situations. There are other, older models such as time-delayed window input vector (TDNN nets) and recurrent structures such as 'Jordan' and 'Elman' but in this study we focus on a more modern way of looking at dynamic time series through semi-supervised framework.

Keywords: K-Means, FCM, IFCM, Intuitionistic fuzzy sets, Volatility, Neural Basis Expansion, NBEATS, Generative Adversarial Network, GAN.

I. INTRODUCTION

In order to understand the financial markets, it is critical to study the volatility of the market. This research report will help the reader in understanding the stock volatility and possible ways of predicting it. Volatility is a metric of the dispersion of returns for a given security or market index. In most cases, the higher the volatility, the riskier the security. Volatility is measured as either the standard deviation or variance between returns from that same security or market index.

In the securities markets, volatility is often associated with big swings in either direction. An asset's volatility is a key factor when pricing options contracts. 'Vega' or the sensitivity of the options' prices with respect to volatility is normally the largest component (or partial derivative in math parlance) of the options' price.

Volatility often refers to the amount of uncertainty or risk related to the size of changes in a security's value. This means that the price of the security can change dramatically over a short time period in either direction. A lower volatility means that a security's value does not fluctuate dramatically and tends to be steadier.

Market volatility can also be seen through the VIX or Volatility Index. The VIX was created by the Chicago Board Options Exchange as a metric to capture the 30-day expected volatility of the U.S. stock market derived from real-time quote prices of S&P 500 call and put options. It is effectively an instrument of future stakes investors and traders are making on the direction of the markets or individual securities. A high reading on the VIX implies a riskier market.

Also referred to as statistical volatility, historical volatility (HV) measures the fluctuations of underlying securities by measuring price changes over certain periods of time. It is the less widespread metric compared to implied volatility because it isn't forward-looking.

When there is a rise in historical volatility, a stock's price will also move more than normal. At this time, there is an expectation that something has changed. If the historical volatility is dropping, on the other hand, it means any uncertainty has been reduced significantly, so things return to the way they were.

CBOE Volatility Index (VIX) is a real-time measure of the market's expectations of the near-term changes in the S&P 500 index. The index is derived from the prices of at the money and out of the money SPX index options with near-term expiration dates. The calculation derives a 30-day forward projection of volatility. In financial markets, volatility is a measure of how fast prices change and measures the degree of fear or greed amongst the market participants. The formula for the VIX calculation [5] is:

$$\sigma^2 = \frac{2}{T}\sum_i \frac{\delta K_i}{K_i^2} e^{RT}\, \mathrm{Q}(K_i) - \frac{1}{T}\left[\frac{F}{K_0} - 1\right]^2 \quad (1)$$

where:

- VIX = σ * 100,
- T = Time to expiration,
- F = Forward index level derived from index option prices,
- K0 = First strike below the forward index level, F
- Ki = Strike price of ith out of the money option; a call if Ki > K0 and a put if Ki < K0; both put and call if Ki = K0.

$$\delta K_i = \frac{K_{i+1} - K_{i-1}}{2}$$

- R = Risk-free interest rate to expiration
- Q(Ki) = The midpoint of the bid-ask spread for each option with strike Ki.

In 2012, CBOE introduced a new index called VVIX. The VVIX index is a volatility of volatility measure as it represents the expected volatility of the 30-day forward price of the CBOE Volatility Index (the VIX). It is this expected volatility that drives the price of the VIX options. VVIX is calculated from the price of a portfolio of liquid at and out of the money VIX options.

As one of the parameters that influences the movement in stock prices, volatility is a crucial input to numerous financial market operations and it becomes an important tool to assess the risk of ruin for portfolio managers, investors and other interested parties [4, 6–9]. The modelling of volatility is a complex task, because it cannot be observed directly. Indeed, it can only be measured by looking at the extent of the movement of the option prices and deriving it mathematically from that.

The mathematical estimators are complicated also by volatility clusters, fat tails, non-normality of the distribution and structural breaks in the distribution of the returns. These features cannot be captured by simple classical models such as the autoregressive moving average (ARMA) process.

A. Clustering Volatility with FS and IFS

A major challenge posed by big data clustering applications is dealing with uncertainty in the formation of the feature vectors. Considering that feature values may be subject to uncertainty owing to imprecise measurements and noise, the distances that determine the membership of a feature vector to a cluster will also be subject to uncertainty. Therefore, the possibility of erroneous membership assignments in the clustering process is evident.

Current fuzzy clustering approaches do not utilise any information about uncertainty at the feature level. This paper accepts the challenge to deal with such kind of information and introduces some thoughts about a modification to the FCM. Features are represented by intuitionistic fuzzy values, i.e., elements of an intuitionistic fuzzy set. Intuitionistic fuzzy sets [6-8] that can be useful in coping with the hesitancy originating from imperfect or imprecise information.

B. Results and Discussion

The computation is coded in Google Colab, Python. We study daily dataset for VIX from January 1990 to April 2022. Our primary data source is Bloomberg Data services. In our view daily data for this study is preferable to other time frames as more short-term data such as minute or hourly data is far too noisy with little predictive value whereas data from longer timeframes is less useful.

There are many measures of volatility. One such measure is called the CBOE Volatility Index or VIX. This index was created by Menachem Brenner and Dan Galai in a series of papers starting in 1989. VIX measures 30-day expectation of volatility given by a weighted portfolio of at and out-the-money European options on the S&P 500 Index. In other words, it is a weighted average of implied volatilities as measured from the call and put prices.

Volatility moves in financial markets are rare and sporadic i.e., periods of low volatility typically follow each other until the regime change due to technical and/or fundamental exogenous factors. Subsequently higher volatility tends to lead to even higher volatility and vice versa.

We are going to use K-means and Fuzzy C-Means clustering techniques on the VIX. VIX is a real time index representing expected volatility over the coming 30 days in percentage terms based on S&P 500 index.

II. K-means

K-means clustering partitions data into K-clusters that minimise squared errors inside clusters using 'Euclidean' distances i.e., it minimises distances from centroids to data points inside clusters.

To find the number of centroids we use the 'Elbow' method and 'Silhouetter' analysis.

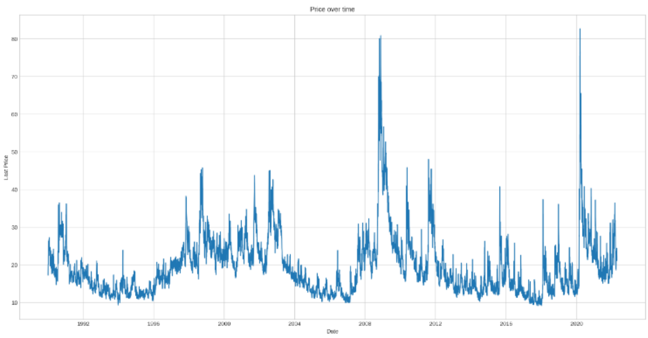


Figure 1. Plot of VIX from 1990 to 2022.

In order to normalise the data, we use 'StandardScaler' technique. We create a normalised data set with mean 0 and standard deviation equal to 1.

As we do not know how many clusters there are, we use 'for' loop with K in range from 1 to 10. This gives us a visual representation using 'Elbow' method.

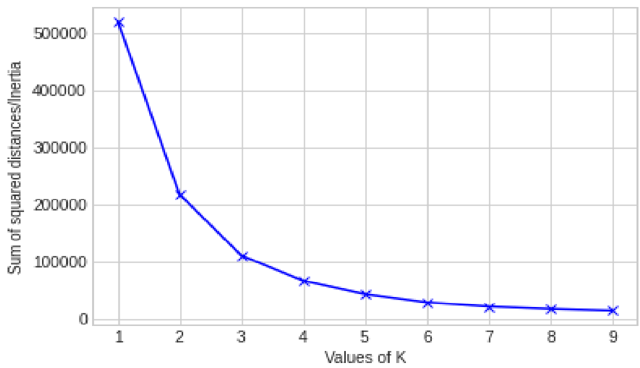


Figure 2. Elbow Method for optimal k

As the number of clusters increase, the error is minimised more and more. Visually, we are looking for an elbow of this curve. It appears in the region of 2–4 clusters.

The ‘Silhouette’ method is another method of finding the optimal number of clusters by computing the silhouette coefficients of each point that measures how much a point is similar to its own cluster compared to other clusters. Similar to ‘Elbow’ method, we train K-means clustering for each of the values of k. Plot of the graph shows the silhouette score on y-axis and the number of clusters on the x-axis.

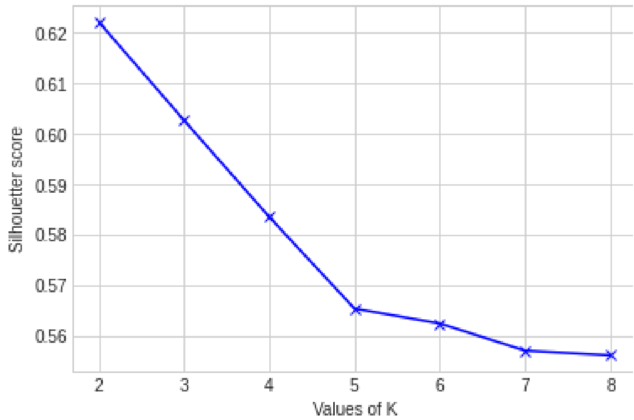


Figure 3. Silhouette analysis for optimal k

Prior to plotting the clusters, we find the centroids of the clusters: [26.81939959], [19.69309811], [13.48929072], [62.16890411], [38.19894207].

Next, we plot the graph with 2 and 5 clusters (Figure 4). The top graph shows two clusters identified by horizontal lines and the bottom graph with 5 clusters identified also by the horizontal lines. The top graph simply splits the data into periods of low and high volatility. The graph shows when the volatility is low it tends to stay low and when it is high it tends to remain high.

The bottom graph has 5 centroids which are represented as horizontal lines. The graph splits the data into periods of very low volatility, low volatility, medium volatility, high volatility, and extremely high volatility.

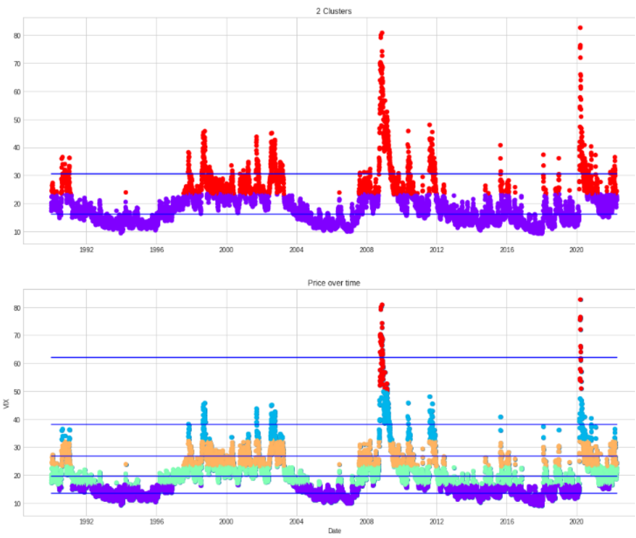


Figure 4. The graph with two clusters (top) and five clusters (bottom)

We find the silhouette score to check how well we fit the data into clusters. The silhouette score falls within the range [–1, 1]. The silhouette score of 1 means that the clusters are very dense and nicely separated. The score of 0 means that clusters are overlapping.

Our silhouette score is 0.554.

Lastly, we provide the transition matrix for the clusters. Table 1 shows how many times the VIX moved from one cluster to another. It is done in absolute terms where rows are starting cluster and columns are final cluster. For example, VIX was 3326 in Cluster 0 and stayed there. It moved 219 times in Cluster 3 from lower cluster to upper cluster. If there were no jumps, the matrix would be symmetric, but it is not.

Index		**Final cluster**				
		0	**1**	**2**	**3**	**4**
Starting culster	**0**	3326	0	0	219	0
	1	1	1231	67	171	0
	2	0	65	324	2	9
	3	218	174	0	2254	0
	4	0	0	9	0	65

Table 1. Transition matrix for the clusters

III. Fuzzy C-Means (FCM)

We are now going to repeat the same exercise using Fuzzy C-Means (FCM) clustering. This algorithm is considered to be better than K-means because unlike K-means where the data points exclusively belong to one cluster, in FCM algorithm, the data point can belong to more than one cluster which is where Fuzzy methods come in. FCM assigns membership grades which indicate the degree to which data points belong to each cluster. Plotting the Fuzzy C-Means Clusters 0–4 gives us the following results (Figure 5).

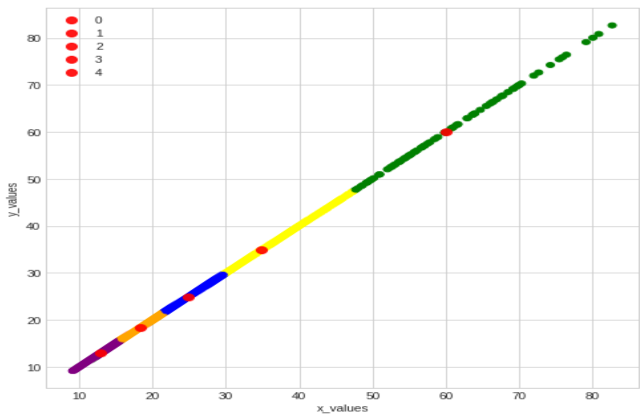


Figure 5. Plotting the Fuzzy C-Means. The red dots are centroids with clusters around them.

Plotting the best Cluster 5 gives 5 centroids which are represented as horizontal lines (Figure 6). The graph splits the data into periods of very low volatility, low volatility, medium volatility, high volatility, and extremely high volatility according to FCM algorithm.

The Silhouette Score is 0.555 which in this case is not very different to K-Means and is probably due to random state initialisation.

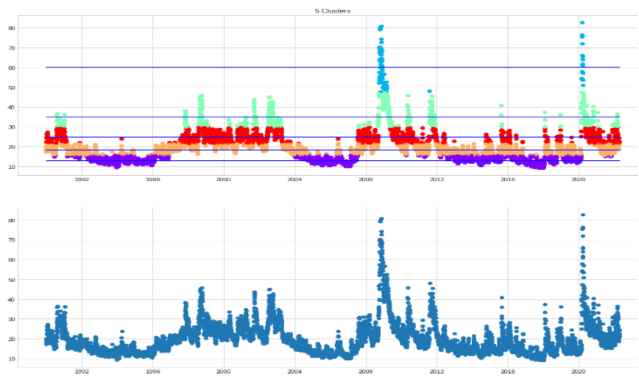


Figure 6. Volatility of Volatility based on Fuzzy C Means

IV. Intuitionistic Fuzzy Sets and IFCM Units

A fuzzy set non-membership value is calculated as a complement of the membership value to 1. However, in reality because of uncertainty, the non-membership is not always equal to one minus the membership value. To deal with this uncertainty, Atanassov proposed another higher order fuzzy set called IFS [1–3]. An IFS Ã in X is given by:

$$\overset{A}{A} \cong \{<x, \mu_A(x), V_A(x)|x€X>\}, \quad (2)$$

where X is a universe of discourse and $\mu\tilde{A}(x) : X \to [0, 1]$, $\nu\tilde{A}(x) : X \to [0, 1]$ with the condition

$0 \leqslant \mu\tilde{A}(x) + \nu\tilde{A}(x) \leqslant 1$; $\forall x \in X$ and $\mu\tilde{A}(x)$, $\nu\tilde{A}(x)$ denote membership and non membership degree, respectively.

For each IFS Ã in X, the hesitation degree should be considered. The hesitation degree of an element $x \in X$ is defined as:

$$\pi\tilde{A}(x) = 1 - \mu\tilde{A}(x) - \nu\tilde{A}(x), \quad (3)$$

where $\pi\tilde{A}(x)$ is hesitation degree and should satisfy the elementary condition of intuitionism, i.e.,

$0 \leqslant \pi\tilde{A}(x) = 1 - \mu\tilde{A}(x) - \nu\tilde{A}(x) \leqslant 1$. In the literature, two fuzzy complements or IFS generators are used to construct intuitionistic fuzzy set: Sugeno's and Yager's [15]. The fuzzy complement function is defined as:

$$N\big(\mu(x)\big) = g^{-1}(g(|) - g\big(\mu(x)\big)), \quad (4)$$

where g(.) is an increasing function and $g : [0, 1] \to [0, 1]$.

Yager's class can be generated by using the following function:

$$g(x) = x\lambda, \quad (5)$$

Non-membership values are calculated from Yager's intuitionistic fuzzy complement N(x). The IFSs using Yager's intuitionistic fuzzy complement become

$$A_\lambda^{IFS} = \left\{x, \mu_A(x), 1 - \mu_A(x))^{\frac{1}{\lambda}} | \, x€X\right\}, \quad (6)$$

Sugeno's negation can be generated using the following function:

$$g(x) \; = \frac{1-x}{1+\lambda x}, \qquad (7)$$

Non-membership values are calculated from Sugeno's intuitionistic fuzzy complement N(x). IFS constructed using Sugeno's fuzzy complement is as follows:

$$A_\lambda^{IFS} = \left\{x, \mu_A(x), \frac{1-\mu_A(x)}{1+\lambda\mu_A(x)} | \, x€X\right\}, \quad (8)$$

In soft clustering methods, the membership value is computed based on a distance function [14–20]. So distance metric plays an important role. In the literature, many distance metrics are proposed developed similarity measures of IFSs based on Hausdorff distance. Provisional results show that Hausdorff distance is simple and works better than other distance metrics. Hence, there is a need to take advantage of IFS and Hausdorff distance to increase the cluster's density and thus separability.

The authors are currently engaged in creating a Python code that can implement IFS in Python and apply it to volatility clustering. The next step will be to replace Yager and Sugeno functions with a different power law function and fuzzify the Hurst Exponent in the power law function. The remaining part to be completed is to merge the fBm with IFS to create a new process which we called Intuitionistic Fuzzy Fractal Brownian motion.

V. Neural Basis Expansion (NBEATS)

Oreshkin et al., [10] proposed a novel architecture for univariate time series forecasting. They applied it to a variety of TS forecasting problems using non-overlapping competition datasets: M4, M3 and Tourism. The results significantly outperformed traditional econometric forecasting techniques.

NBEATS is analysis for interpretable time series forecasting. The focus is on solving the univariate times series point forecasting problem using deep learning architecture based on backward and forward residual links and a very deep stack of fully connected layers. The architecture has several desirable properties, being interpretable, applicable without modification to a wide array of target domains, and fast to train.

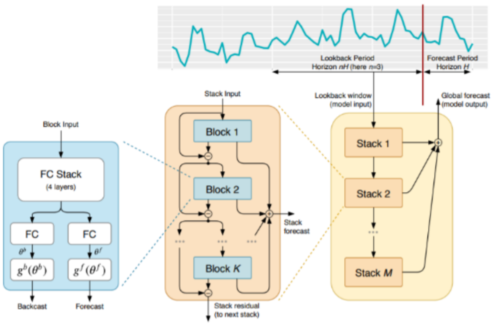
The model consists of a sequence of stacks, each of which combines multiple blocks. The blocks connect feedforward networks via forecast and backcast links. A block "removes the portion of the signal … it can approximate well" [9]. Then the block sets its focus on the residual error, which the preceding blocks could not disentangle. Each block generates a partial forecast, with its focus set on the local characteristics of the time series. The stack aggregates partial predictions on the blocks it consists of, and then passes the results on to the next stack. The purpose of the stack is to identify non-local patterns along the complete time axis by "looking back." Finally, the partial forecasts are aggregated into an overall forecast at the model level. N-BEATS takes as its hyperparameters:

i. The size of the input and output layers (constants INLEN and N_FC) must be sufficient to assign a node for each feature in the source data. The length of the input segment should not be less than the order of seasonality, otherwise the learning process will have more difficulty combining the segments. For efficient memory usage, set them to a power of 2.

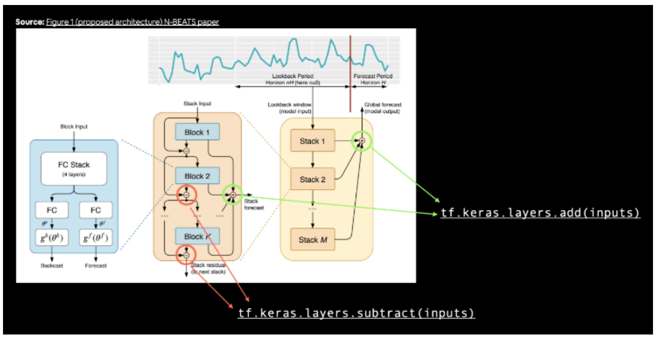
ii. The number of blocks in a stack (BLOCKS).

iii. The width of each fully connected layer in each block of a stack: its node count (LWIDTH).

The batch size determines the number of cases the model will process before updating its matrix weights. To effectively align it with the memory structure of your system, set it to a power of 2. Very large batch sizes can skew gradients down only in one direction, and the model can get stuck at a sub-optimal minimum. Smaller batch sizes will cause the gradient descent to bounce around in different directions and can result in lower accuracy, but they also tend to prevent the model from overfitting. The most frequent recommendation is to choose an initial batch size of 32. Since our dataset has a frequency of 24 hours per day, we set the batch size to the following binary limit that can handle 24 time steps: 32. The epochs tell the model the number of cycles the exercise it has to perform In each epoch, the model processes the entire training set, performing one forward pass and one back pass. Allowing for some oversimplification, the product of these hyperparameters defines the tensor size of the model. Large parameter values can cause it to reach the system's memory limit and cause exponentially longer processing times. While small parameter values may not be sufficient to reflect complex patterns in the source data. We obtain a probabilistic forecast using quantile regression. This is an option we can exercise in all deep forecasting models. The loss function of a neural network can be constructed using quantile loss function. Then quantile regression will not only compute a central forecast value at each time step, a point estimate, but will draw uncertainty bands about it. Pairs of quantiles like 1%/99% or 10%/90% express the range over which the forecast value can vary, above or below the central value. In future work, we are planning to use transformers which is a relatively new concept in neural networks. Unlike univariate time series, transformer will learn to integrate various factors that influence VIX as regressors and use multivariate time series with multiple time frames. For example, we could split the time intervals into minutes, hours and days. This would provide a complex time series for a neural network to learn from.



Source: https://medium.com/@jonathanbechtel/kerasbeats-an-easy-way-to-use-n-beats-in-keras-395b24c5cc28

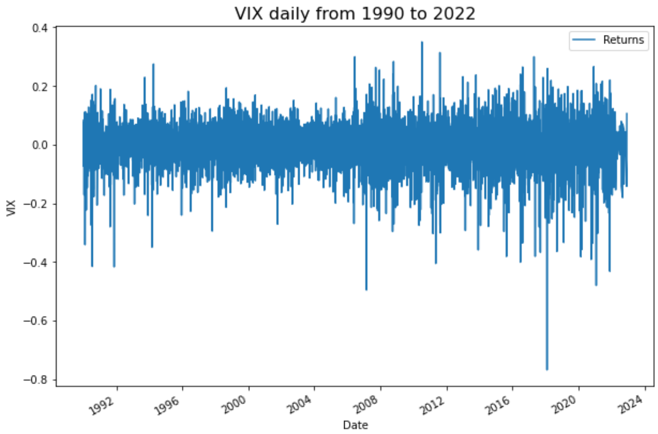


(Boris N. Oreshkin, 24 May 2019)

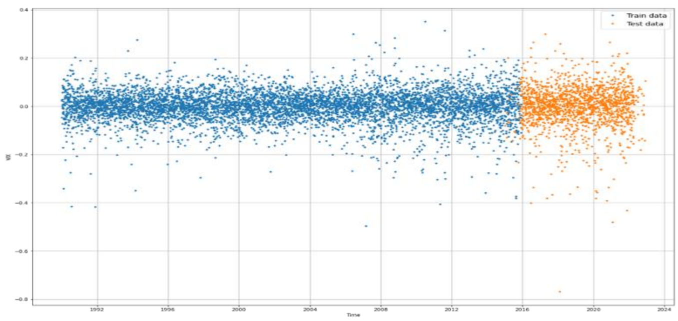
Next, we will detrend data and remove seasonality. We will compare NBEATS against naïve model and some other Deep learning models.

A. RESULTS

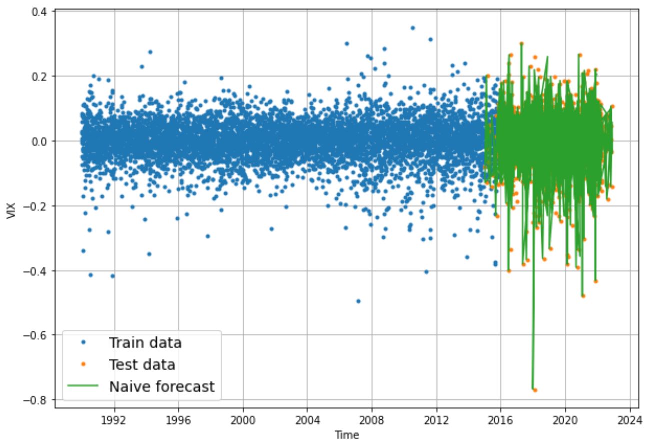
As was already mentioned, we ran data from January 1990 to April 2022. Graph below shows scaled log returns of VIX data series.



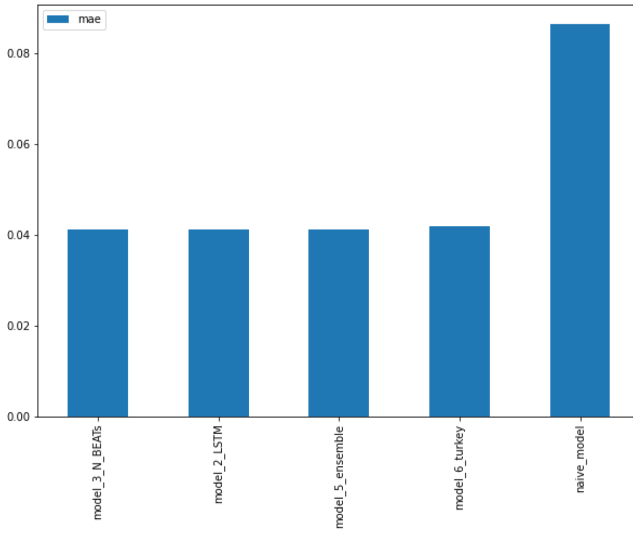
Train and test split



Naïve forecast



NBEATS model seems to do quite well compared to naïve model, but it is not a lot better than other deep learning models such as Conv1d and LSTM.



	mae	mse	rmse	mape	mase
naive_model	0.086481	0.015316	0.086481	421354.18	0.999
model_2_LSTM	0.041248	0.003253	0.057037	59694.81	0.672
model_3_N_BEATs	0.041183	0.003258	0.057076	56545.72	0.671
model_5_ensemble	0.041321	0.003274	0.057219	46762.38	0.673
model_6_turkey	0.041824	0.003423	0.042492	33452.46	0.681

VI. A Generative Adversarial Network (GAN)

A generative adversarial network (GAN) [9] is a class of machine learning frameworks designed by Ian Goodfellow and his colleagues in June 2014. Two neural networks compete in a zero-sum game, where one agent's gain is another agent's loss.

With a training set, this technique learns to generate new data with the same statistics as the training set. For example, an image-trained GAN can produce new images that appear realistic at least to a human observer, with many realistic features. Though originally proposed as a form of generative model for unsupervised learning, GANs have also proved useful for semi-supervised learning, fully supervised learning, and reinforcement learning.

The core idea of a GAN is based on the "indirect" training through the discriminator, another neural network that can tell how much an input is "realistic", which itself is also being updated dynamically [4]. This basically means that the generator is not trained to minimize the distance to a particular frame, but rather to fool the discriminator. This enables the model to learn in an unsupervised manner.

GANs are like mimicry in evolutionary biology, with an evolutionary arms race between both networks. To learn the generator's distribution p(g) over data x, we define a prior on input noise variables p(z) then represent a mapping to data space as G(z; ϴg), where G is a differentiable function represented by a multiplayer perceptron with parameters ϴg. We also define a second multilayer perceptron D (x; ϴd) that outputs a single scalar. D(x) represents probability that x came from the data rather than p(g). We train Discriminator (D) to maximise the probability of assigning the correct label to both training examples and samples from G. Simultaneously we train G to minimise log(1 – D(G(z))):

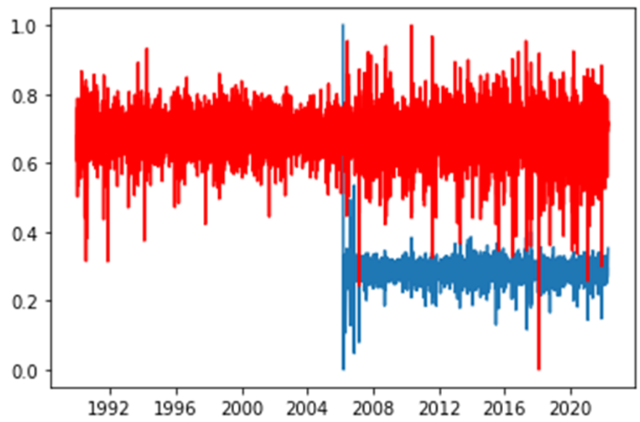
So in other words, D and G play the following two-player minimax game with value function V(G, D):

$$\min_{G}\max_{D} V(D, G) = E_{x\sim p_{data}(x)}\left[\log D(x)\right] + E_{z\sim p_z(z)}\left[\log(1 - D(G(z)))\right] \quad (9)$$

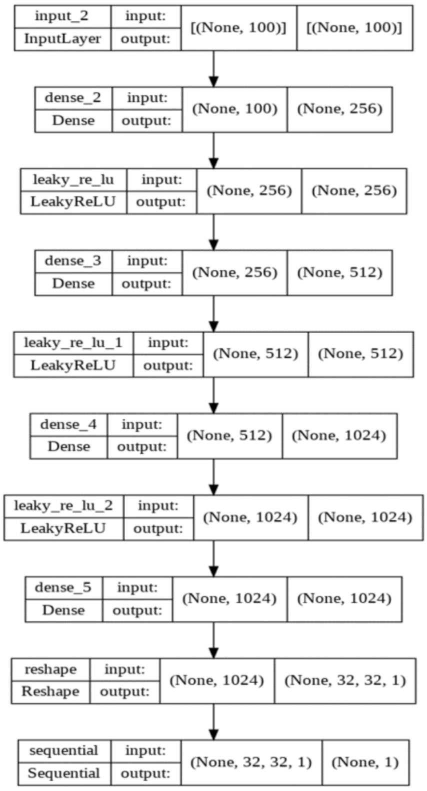
(Ian J. Goodfellow, 2014)

Kang Zhang et al. [22] proposed a novel architecture of GAN with Multi-Layer Perceptron (MLP) as the discriminator and LSTM as the generator for forecasting the closing stock prices. The generator is built by LSTM to mine the data distribution of stocks from given data and generate data in the same distributions, whereas the discriminator designed MLP to discriminate between real and fake data. Their results showed that the novel GAN structure gave promising results compared with other models in machine learning and deep learning. HungChun Lin et al., [11] proposed stock price prediction model using GAN with Gated Recurrent Units (GRU) used as a generator that inputs historical stock price and generates future stock price and Convolutional Neural Network (CNN) as a discriminator to discriminate between real and fake distributions. Different to one step ahead forecasts only, the authors used deep learning to make multi-step ahead predictions more accurately. The authors also found that Wasserstein GAN performed better during unexpected events like COVID whereas conventional GAN performed better during normal times. They also showed that including RNN into a GAN makes it unstable because it is challenging to tune hyperparameters and without suitable parameters the results are poor. The key is tune parameters in each layer to make the whole model more accurate. Other researchers use Rainbow method based on Q-learning for hyperparameter optimisation. Martin Erdman et al., [7] used adversarial network with the Wasserstein distance to generate simulated detector data. The authors investigated two variants of GANs for detector simulations. In both cases, the transfer of probability distributions from one data set to another using GAN training worked well using the Wasserstein distance in the loss function. Instead of training a deep network with simulations that differ in detail from data, simulations can be adapted to match data prior to network training. With this method, authors showed that results are better compared to training with the originally simulated traces.

In this paper, we will use normalised log scale return VIX as our train data and normalised log scale return VVIX as our validation data.



We will then build ARIMA on scaled data and predict the fitted values on validation data. We also calculate the future trend growth with ARIMA. We will then make a dataframe of the fitted values and split it into train and test sets for our GAN model. We will add CNN conv1 layer as our discriminator and choose accuracy as our metrics. Then in generator we use two layers of CNN and LSTM together and again we choose accuracy as our metrics. For training the model, we compile discriminator and generator together and use the learning rate of 0.001 and beta of 0.5. A plot of the model is also created, and we can see that the model expects a 100-element point in laten space as input and will predict a single output.



A. Results:

We ran the model for a total of 1200 epochs and it gave us the accuracy rate of 93.3%. We then use the fitted (predicted) values of GAN as features/inputs in SVM and RF. We define the RF model and use train features for training and fitting and test features for predicting. The results for MAPE and MAE are quite encouraging. For SVM, we use grid search and fit the model to get a score of 98.5% on fitting.

RF

MAPE: 0.0074763204010184145

MAE: 0.005130449308450033

SVM

MAPE: 0.005861350065975159

MAE: 0.004021246354831941

VII. Conclusions

Central Banks in particular pay very close attention to volatility and volatility of volatility. In fact, we know that most Central bank models include volatility of volatility as a factor of how loose or tight the financial conditions are. Hence, it has direct impact on things that most people care about such as their mortgage rates, inflation and unemployment. With a very few exceptions, unlike pure sciences and math, systems in economics exhibit inter-dependencies, hesitation and reflexivity i.e., participation in experiments influences its outcomes and vice versa. To this extent IFS based unsupervised techniques for data analysis seem to have an advantage.

References

[1] Atanassov, K. T. (1983). Intuitionistic fuzzy sets, *VII ITKR's Session*, Sofia, 1983 (Deposed in Central Science–Technology Library of Bulgaria Academy of Science –1697/84).

[2] Atanassov, K. T. (1999). Intuitionistic Fuzzy Sets: Past, Present and Future. In: Wagenknecht, M., & Hampel, R. (eds) *3rd Conference of the European Society for Fuzzy Logic and Technology*, Zittau, Germany (10.09.2013–12.09.2003), 12–19.

[3] Atanassov, K. T. (1999). Intuitionistic Fuzzy Sets: Theory and Applications. Physica-Verlag, New York.

[4] Adaloglou, N. (2020). GANs in computer vision – Introduction to generative learning. AI Summer.

[5] Bucci, A. (2020). Realized Volatility Forecasting with Neural Networks. *Journal of Financial Econometrics*, 18(3), 502–531.

[6] Demeterfi, K., Derman, E., Kamal, M., & Zou, J. (1999). *More Than You Ever Wanted to Know about Volatility Swaps*. Goldman Sachs Quantitative Strategies Research Notes, March 1999.

[7] Erdmann, M., Geiger, L., Glombitza. & Schmidt, D. (2018). Generating and refining particle detector simulations using the Wasserstein distance in adversarial networks. Computer and Software for Big Science 2 (1), 4.

[8] Francq, C., & Zakoian, J.-M. (2010). *GARCH Models: Structure, Statistical Inference and Financial Applications*. 1st edition ed. Chichester: John Wiley & Sons.

[9] Goodfellow, I., Pouget-Abadie J., Mirza M., Xu B., Warde-Farley D., Ozair S., Courville A. & Bengio Y. (2014). *Generative Adversarial Nets*, University of Montreal .

[10] Oreshkin B, Carpov D, Chapados N & Bengio Y. (2019). N-BEATS: *Neural Basis Expansion analysis for interpretable time series forecasting*. Cornell University.

[11] Lin, H., Chen, C., Huang, G & Jafari, A. (2021). Stock price prediction using Generative Adversarial Networks. Journal of Computer Science, 17(3), 188-196.

[12] Poon, S.-H., & Granger, C. W. J. (2003). Forecasting Volatility in Financial Markets: A Review. *Journal of Economic Literature*, 41(2), 478–539.

[13] Pradeepkumar, D., & Ravi, V. (2017). Forecasting financial time series volatility using Particle Swarm Optimisation trained Quantile Regression Neural Network. *Applied Soft Computing*, 58, 35–52.

[14] Site, A., Birant, D., & Isik, Z. (2019). Stock Market Forecasting Using Machine Learning Models. *IEEE 2019 Innovations in Intelligent Systems and Applications Conference (ASYU)* - Izmir, Turkey (31.10.2019 – 02.11.2019), 318–323.

[15] Sugeno, M., & Terano, T. (1977). A model of learning based on fuzzy information, *Kybernetes*, 6, 157–166.

[16] Szmidt, E., & Kacprzyk, J. (1997). On measuring distances between intuitionistic fuzzy sets. *Notes on Intuitionistic Fuzzy Sets*, 3(4), 1–3.

[17] Szmidt, E., & Kacprzyk, J. (2000). Distances between intuitionistic fuzzy sets, *Fuzzy Sets and Systems*, 114(3), 505–518.

[18] Szmidt, E., & Kacprzyk, J. (2001). Intuitionistic fuzzy sets in some medical applications, *Notes on Intuitionistic Fuzzy Sets*, 7(4), 58–64.

[19] Van Lung, H., & Kim, J.-M. (2009). A generalized spatial fuzzy *C*-means algorithm for medical image segmentation. In: FUZZ-IEEE'09: Proceedings of the 18th international conference on Fuzzy Systems, Jeju Island, Korea, 20.08.2009 – 24.08.2009, 409–414.

[20] Wang, W., & Zhang, Y. (2007). On fuzzy cluster validity indices. *Fuzzy Sets and Systems*, 158(19), 2095–2117.

[21] Wang, Z., Xu, Z., Liu, S., & Tang, J. (2011). A netting clustering analysis method under intuitionistic fuzzy environment. *Applied Soft Computing*, 11(8), 5558–5564.

[22] Wang, Z., Xu, Z., Liu, S., & Yao, Z. (2014). Direct clustering analysis based on intuitionistic fuzzy implication. *Applied Soft Computing*, 23, 1–8.

[23] Zhang, K., Zhong, G., Dong, J., Wang, S. & Wang, Y. (2018). Stock Market Prediction Based on Generative Adversarial Network. International Conference on Identification, Information and Knowledge in the Internet of Things, IIKI 2018, Procedia Computer Science, 147 400-406.