B.Tech. Major Project Presentation - 1

Volatility Model Analysis using Exponential Generalized Autoregressive Conditional Heteroskedasticity (Egarch) Model And GJR Garch On Indian Cryptocurrencies

Under the supervision of Professor

Dr. R. Srivastava

Department of Applied Mathematics

Kunal Sharma 2K18/MC/060

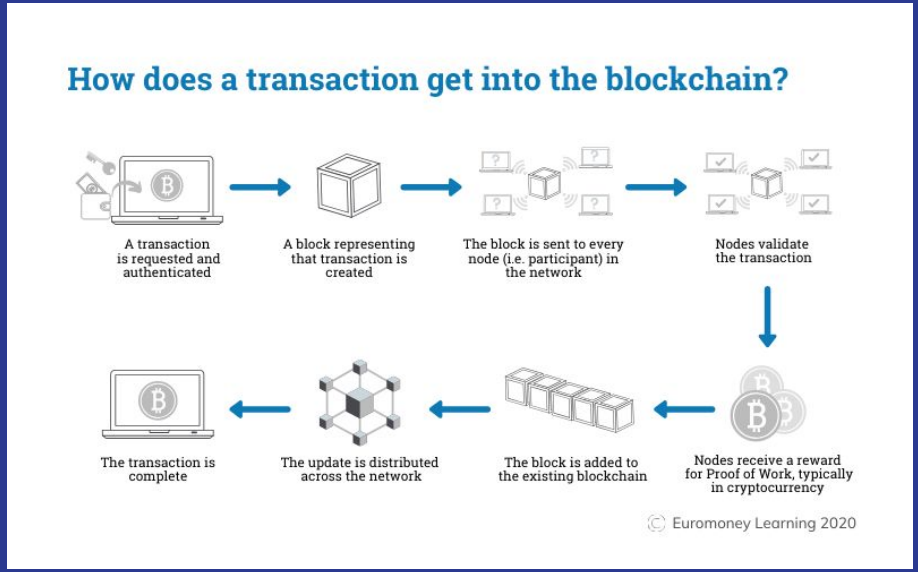
Rahul Sharma 2K18/MC/087

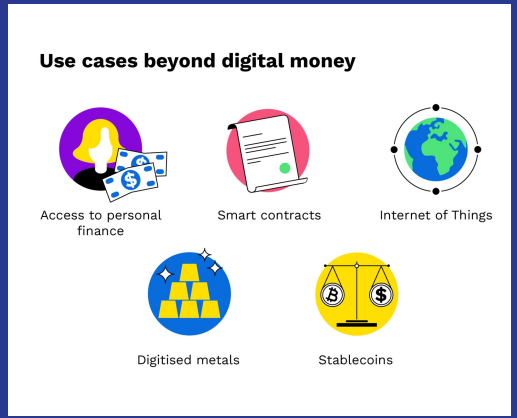
A. ABSTRACT

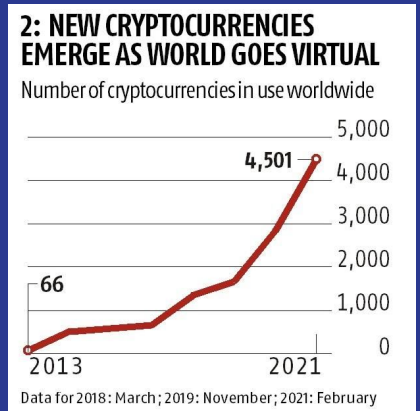
Recent development in the field of Digital Currencies such as Cryptocurrencies like Bitcoin, Etherium, DogeCoin etc have piqued the attention and interest of Investors, Financial Researchers and Institutions, The concept of Cryptocurrencies' volatility is a burning issue at this time. This project besides determining the high rate volatility in cryptocurrency prices fits the Exponential Generalized Autoregressive Conditional Heteroskedasticity (EGARCH) and other similar models to the volatility of the digital currency. Model selection criteria will be carried out using Bayesian Information Criterion

B. MOTIVATION

The increased interest in this form of digital assets has a pressing need for quantification of variation. As compared to the traditional currencies, these digital assets are highly volatile in nature and vary rapidly so much so that we cannot assume their exchange rate to be IID or Independently and Identically Distributed variables. Hence the motivation of the project lies in modelling the volatility of these currencies using GARCH and its variant model for understanding the nature of these digital assets.







PROBLEM STATEMENT

A.The Problem modelled here is the problem of prediction and forecasting of the volatility on the exchange rate of cryptocurrencies, very little to no work has been done on the Indian developed cryptocurrencies, that is why here instead of considering the biggest crypto by market cap, we consider the two biggest digital assets considered in India, namely MATIC coin developed by POLYGON and WRX developed by online brokerage WazirX. The cryptocurrency returns are in essence uncorrelated with traditional asset classes such as bonds or shares of stock.

B. **POLYGON MATIC/INR**
Ethereum token that helps the network of Polygon, which is a scaling solution for Ethereum. The objective of Polygon is to help provide more reliable and swift transactions on Ethereum using 2nd Layer Sidechains.

C. WAZIRX WRX/INR
This is a form of utility token of the company WazirX, based entirely upon the Binance Blockchain Technology And total supply of this right now is approximately 1 Billion.

.

OBJECTIVE

These cryptocurrency demonstrate some positive relation between volatility and shock in the Pre-crash period, but now since they are expanded their use and idolization creates a pressing issue of measuring volatility. We try to find the best fitting GARCH model as described by BIC (Bayesian Inference Criterion). The objective of this project is to understand which conditional heteroskedasticity model can explain the diversified digital cryptocurrencies rate volatility.

Methodology: Generalized Autoregressive Conditional Heteroskedasticity Model

Autoregressive Model and their variants - AR models are a type of random processes that are used to describe time-varying processes in financial engineering. The component of autoregression specifies that output variable is linearly related to some of its previous values depending upon the parameters of the AR model.

$$X_t = c + \sum_{i=1}^{p} \varphi_i X_{t-i} + \varepsilon_t \qquad X_t = c + \sum_{i=1}^{p} \varphi_i B^i X_t + \varepsilon_t$$

ARCH is the type of statistical model used in Time-Series Datasets that helps in understanding the variance of Error Term as a reason for the size of previous time periods and its error terms. These models are used in modelling of Time series data and volatility and clustering, which are time periods of ups and downs, ARCH belongs to a family of volatility models.

$\epsilon_t = \sigma_t z_t$ and the series of time dependent variance

$$\sigma_t^2 = \alpha_0 + \alpha_1 \epsilon_{t-1}^2 + \cdots + \alpha_q \epsilon_{t-q}^2 = \alpha_0 + \sum_{i=1}^{q} \alpha_i \epsilon_{t-i}^2$$

GARCH - If the AR model is coupled with Moving Average generating ARMA and then we assume that for the error term, the model becomes Generalized ARCH, hence having two parameters namely p and q.

$$\sigma_t^2 = \omega + \alpha_1 \epsilon_{t-1}^2 + \cdots + \alpha_q \epsilon_{t-q}^2 + \beta_1 \sigma_{t-1}^2 + \cdots + \beta_p \sigma_{t-p}^2 = \omega + \sum_{i=1}^{q} \alpha_i \epsilon_{t-i}^2 + \sum_{i=1}^{p} \beta_i \sigma_{t-i}^2$$

P is the lag length of Generalized ARCH models and q is the order of the model, similarly there are multiple models that are available for modelling volatility, here we will consider EGARCH and GJR GARCH.

$$\rho = \frac{\sum_{t=i+1}^{T} (\hat{\epsilon}_t^2 - \hat{\sigma}_t^2)(\hat{\epsilon}_{t-1}^2 - \hat{\sigma}_{t-1}^2)}{\sum_{t=1}^{T} (\hat{\epsilon}_t^2 - \hat{\sigma}_t^2)^2}$$

AUTOCORRELATION

$$\log \sigma_t^2 = \omega + \sum_{k=1}^{q} \beta_k g(Z_{t-k}) + \sum_{k=1}^{p} \alpha_k \log \sigma_{t-k}^2$$

EGARCH

$$\sigma_t^2 = K + \delta \sigma_{t-1}^2 + \alpha \epsilon_{t-1}^2 + \phi \epsilon_{t-1}^2 I_{t-1}$$

where $I_{t-1} = 0$ if $\epsilon_{t-1} \geq 0$, and $I_{t-1} = 1$ if $\epsilon_{t-1} < 0$.

GJR GARCH

Date	Price	Open	High	Low
Oct 31, 2021	155.000	147.360	158.200	145.014
Oct 30, 2021	147.360	159.021	159.021	147.071
Oct 29, 2021	159.021	166.969	174.427	142.000
Oct 28, 2021	166.969	142.546	169.650	142.458
Oct 27, 2021	142.546	136.445	150.300	127.604
...	...	...	...	...
Aug 05, 2021	81.199	80.534	81.600	78.100
Aug 04, 2021	80.534	78.970	81.899	78.013
Aug 03, 2021	78.970	80.001	81.474	77.129
Aug 02, 2021	80.001	81.117	82.872	78.610
Aug 01, 2021	81.117	82.168	86.050	80.300

[92 rows x 6 columns]
<class 'pandas.core.frame.DataFrame'>

	Date	Price	Open	High	Low
0	Oct 31, 2021	1.4300	1.3820	1.5430	1.3710
1	Oct 30, 2021	1.3780	1.3590	1.3800	1.3070
2	Oct 29, 2021	1.3560	1.3440	1.3830	1.3350
3	Oct 28, 2021	1.3430	1.3030	1.3840	1.2590
4	Oct 27, 2021	1.3100	1.4060	1.4680	1.2670
..	...	...	...	...	...
87	Aug 05, 2021	1.1085	1.0916	1.1129	1.0526
88	Aug 04, 2021	1.0900	1.0518	1.1054	1.0320
89	Aug 03, 2021	1.0478	1.0881	1.0926	1.0373
90	Aug 02, 2021	1.0869	1.0883	1.1266	1.0696
91	Aug 01, 2021	1.0855	1.1283	1.1600	1.0760

[92 rows x 7 columns]
<class 'pandas.core.frame.DataFrame'>
<class 'pandas.core.frame.DataFrame'>

The data preprocessing stage involved cleaning of the data and extracting the relevant information from the .csv data file. Since all of the data contained is time-series data or time varying random/stochastic process, we need Hurst Exponent Function which is famous measure used in statistics for classifying the time series data and deriving the inferences of the level of difficulty in choosing apt model for series and carrying out predictions. Random Series - For Exponent in the vicinity of 0.5, Mean Reverting Series for value near 0 and Trending series value near 1.

```python
def hurst(ts):
    """Returns the Hurst Exponent of the time series vector ts"""
    # Create the range of lag values
    lags = range(2, 100)
    # Calculate the array of the variances of the lagged differences
    tau = [sqrt(std(subtract(ts[lag:], ts[:-lag]))) for lag in lags]
    # Use a linear fit to estimate the Hurst Exponent
    poly = polyfit(log(lags), log(tau), 1)
    # Return the Hurst exponent from the polyfit output
    return poly[0]*2.0
```

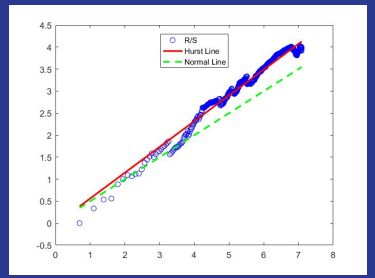
```python
def plot_correlogram(x, lags=None, title=None):
    lags = min(10, int(len(x)/5)) if lags is None else lags
    fig, axes = plt.subplots(nrows=2, ncols=2, figsize=(12, 8))
    x.plot(ax=axes[0][0])
    q_p = np.max(q_stat(acf(x, nlags=lags), len(x))[1])
    stats = f'Q-Stat: {np.max(q_p):>8.2f}\nADF: {adfuller(x)[1]:>11.2f} \nHurst:
     {round(hurst(x.values),2)}'
    axes[0][0].text(x=.02, y=.85, s=stats, transform=axes[0][0].transAxes)
    probplot(x, plot=axes[0][1])
    mean, var, skew, kurtosis = moment(x, moment=[1, 2, 3, 4])
    s = f'Mean: {mean:>12.2f}\nSD: {np.sqrt(var):>16.2f}\nSkew: {skew:12.2f}
    \nKurtosis:{kurtosis:9.2f}'
    axes[0][1].text(x=.02, y=.75, s=s, transform=axes[0][1].transAxes)
    plot_acf(x=x, lags=lags, zero=False, ax=axes[1][0])
    plot_pacf(x, lags=lags, zero=False, ax=axes[1][1])
    axes[1][0].set_xlabel('Lag')
    axes[1][1].set_xlabel('Lag')
    fig.suptitle(title, fontsize=20)
    fig.tight_layout()
    fig.subplots_adjust(top=.9)
```

```
print(maticDF.describe())
```

	Return	Low	High	Open	Price
count	91.000000	91.000000	91.000000	91.000000	91.000000
mean	-0.641882	99.252363	111.517615	106.164769	106.881165
std	6.143578	15.811043	18.039246	16.570331	16.926747
min	-22.297415	65.000000	81.474000	78.970000	78.970000
25%	-3.024194	86.250500	99.667000	96.999500	97.050000
50%	-0.146725	98.412000	110.784000	104.871000	105.159000
75%	2.921881	110.100000	119.899500	115.542000	115.917000
max	14.582280	147.071000	174.427000	166.969000	166.969000

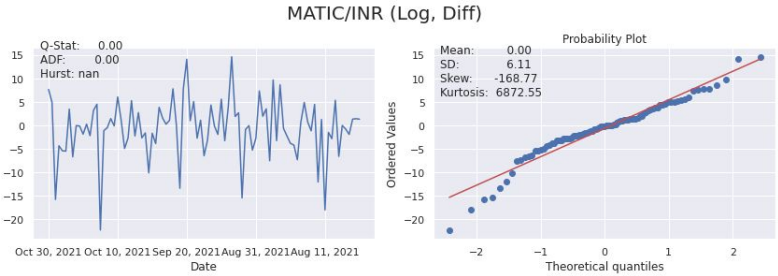
```
[67] print(wrxDF.describe())
```

	Price	Open	High	Low	Return
count	91.000000	91.000000	91.000000	91.000000	91.000000
mean	1.282068	1.279454	1.339871	1.227203	-0.222879
std	0.163305	0.163272	0.171948	0.154054	5.513484
min	0.999000	0.995000	1.030000	0.962000	-20.245724
25%	1.160000	1.148950	1.210500	1.108000	-3.607066
50%	1.252000	1.252000	1.347000	1.200000	-0.279051
75%	1.410500	1.410000	1.491500	1.358850	2.409976
max	1.665400	1.662400	1.745400	1.490000	22.869692

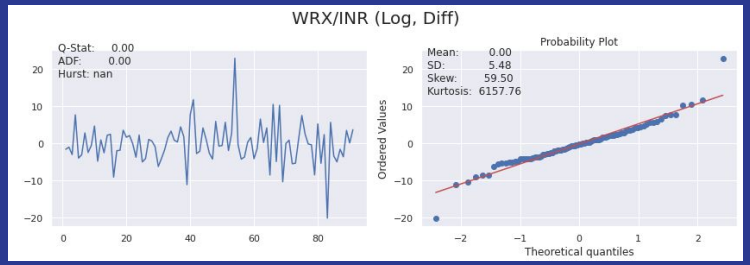


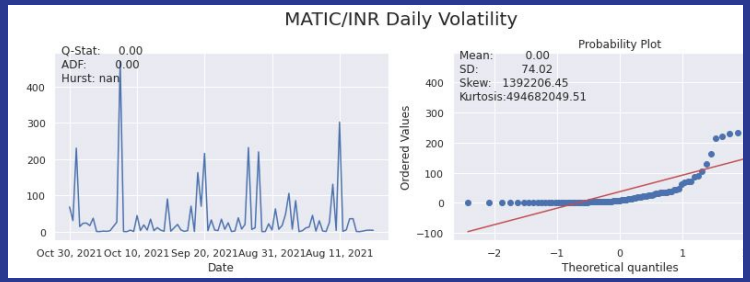
EXPERIMENTAL CODING

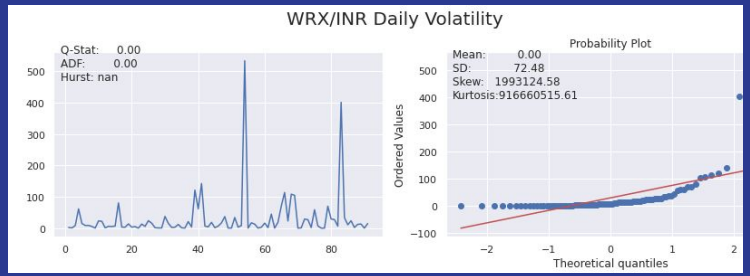
After basic preprocessing of the data and passing the time series through Hurst Exponent, we then plot the Correlogram charts containing information regarding the Autocorrelation and Partial Autocorrelation, Probability plot and finally Q-stat and ADF for both of our crypto under consideration. We then apply the GARCH, EGARCH and finally the GJR-GARCH model on both of our cryptocurrency data to find the short term investment plan by deriving relation between observed volatility of the stock and the volatility as observed from the model (GARCH, EGARCH, GJR GARCH).Basic parameters of the Volatility model like Omega, Alpha, Gamma and Beta were found and the covariance estimate was robust in nature. Finally basic descriptive statistics are obtained for understanding some of the general properties of cryptocurrency data like Quartiles, Mean, Std, interquartile ranges, skewness and kurtosis of MATIC/INR and WRX/INR.



The following are the logs. Diff charts and Daily Volatility for MATIC/INR and WRX/INR Cryptocurrency which shows us the underlying Time - Series data, Probability plot, Autocorrelation and Partial Autocorrelation.





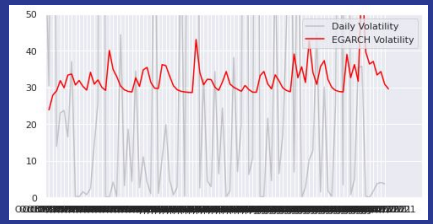


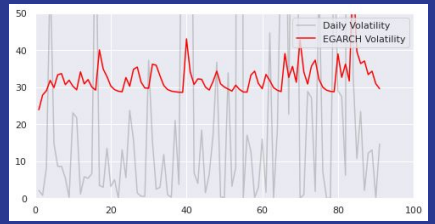
CODE SNIPPET FOR EGARCH ON MATIC AND WRX

```
# Specify GJR-GARCH model assumptions
gjr_gm = arch_model(maticDF['Return'], p = 1, q = 1, o = 1, vol = 'GARCH',
                    dist = 't')
# Fit the model
gjrgm_result = gjr_gm.fit(disp = 'off')
# Print model fitting summary
print(gjrgm_result.summary())

# Specify EGARCH model assumptions
egarch_gm = arch_model(maticDF['Return'], p = 1, q = 1, o = 1, vol = 'EGARCH',
                       dist = 't')
# Fit the model
egarch_result = egarch_gm.fit(disp = 'off')
# Print model fitting summary
print(egarch_result.summary())
```

```
# Specify GJR-GARCH model assumptions
gjr_gm = arch_model(wrxDF['Return'], p = 1, q = 1, o = 1, vol = 'GARCH',
                    dist = 't')
# Fit the model
gjrgm_result = gjr_gm.fit(disp = 'off')
print(gjrgm_result.summary())
egarch_gm = arch_model(wrxDF['Return'], p = 1, q = 1, o = 1, vol = 'EGARCH',
                       dist = 't')
egarch_result = egarch_gm.fit(disp = 'off')
print(egarch_result.summary())
gjrgm_vol = gjrgm_result.conditional_volatility
egarch_vol = egarch_result.conditional_volatility
# Plot the actual returns
plt.axis([0, 100, 2, 8])
plt.plot(wrxDF['Return'], color = 'grey', alpha = 0.4, label = 'Price Returns')
plt.plot(egarch_vol, color = 'red', label = 'EGARCH Volatility')
plt.legend(loc = 'upper right')
plt.show()
# Print each models BIC
print(f'GJR-GARCH BIC: {gjrgm_result.bic}')
print(f'\nEGARCH BIC: {egarch_result.bic}')
```

```
                     Constant Mean - EGARCH Model Results
==============================================================================
Dep. Variable:                    Return   R-squared:                       0.000
Mean Model:                Constant Mean   Adj. R-squared:                  0.000
Vol Model:                        EGARCH   Log-Likelihood:               -279.741
Distribution:      Standardized Student's t   AIC:                        571.482
Method:                 Maximum Likelihood   BIC:                        586.548
                                             No. Observations:                   91
Date:                  Thu, Nov 25 2021      Df Residuals:                       90
Time:                          10:52:12      Df Model:                           1
                            Mean Model
==============================================================================
                 coef    std err          t      P>|t|      95.0% Conf. Int.
------------------------------------------------------------------------------
mu            -0.1638  3.602e-02     -4.547  5.436e-06 [ -0.234,-9.320e-02]
                              Volatility Model
==============================================================================
                 coef    std err          t      P>|t|    95.0% Conf. Int.
------------------------------------------------------------------------------
omega         0.2606  4.593e-06  5.674e+04      0.000   [  0.261,  0.261]
alpha[1]     -0.3848  2.534e-03   -151.841      0.000   [ -0.390, -0.380]
gamma[1]      0.0981  5.144e-03     19.066  4.804e-81 [8.800e-02,  0.108]
beta[1]       0.9258  8.264e-07  1.120e+06      0.000   [  0.926,  0.926]
                              Distribution
==============================================================================
                 coef    std err          t      P>|t|  95.0% Conf. Int.
------------------------------------------------------------------------------
nu            6.8365  5.469e-02    125.000      0.000 [  6.729,  6.944]
==============================================================================

Covariance estimator: robust
```





CODE SNIPPET FOR GJR GARCH ON MATIC AND WRX

```
# Specify GJR-GARCH model assumptions
gjr_gm = arch_model(maticDF['Return'], p = 1, q = 1, o = 1, vol = 'GARCH',
                    dist = 't')
# Fit the model
gjrgm_result = gjr_gm.fit(disp = 'off')
# Print model fitting summary
print(gjrgm_result.summary())

# Specify EGARCH model assumptions
egarch_gm = arch_model(maticDF['Return'], p = 1, q = 1, o = 1, vol = 'EGARCH',
                       dist = 't')
# Fit the model
egarch_result = egarch_gm.fit(disp = 'off')
# Print model fitting summary
print(egarch_result.summary())
```

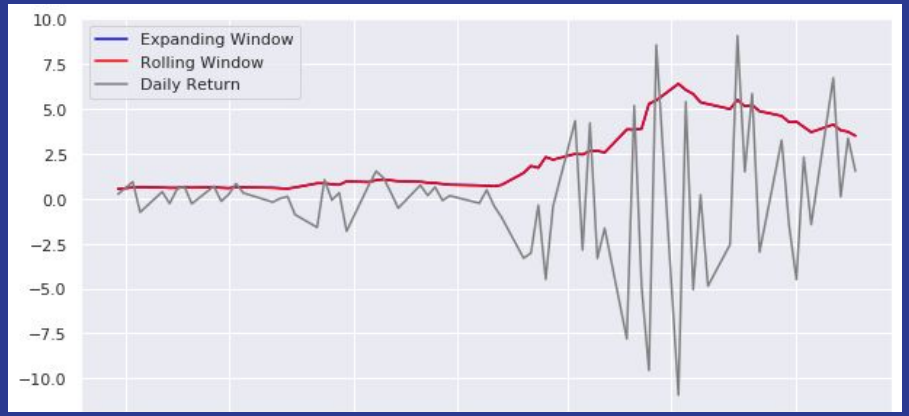
```
# Specify GJR-GARCH model assumptions
gjr_gm = arch_model(wrxDF['Return'], p = 1, q = 1, o = 1, vol = 'GARCH',
                    dist = 't')
# Fit the model
gjrgm_result = gjr_gm.fit(disp = 'off')
print(gjrgm_result.summary())
egarch_gm = arch_model(wrxDF['Return'], p = 1, q = 1, o = 1, vol = 'EGARCH',
                       dist = 't')
egarch_result = egarch_gm.fit(disp = 'off')
print(egarch_result.summary())
gjrgm_vol = gjrgm_result.conditional_volatility
egarch_vol = egarch_result.conditional_volatility
# Plot the actual returns
plt.axis([0, 100, 2, 8])
plt.plot(wrxDF['Return'], color = 'grey', alpha = 0.4, label = 'Price Returns')
plt.plot(egarch_vol, color = 'red', label = 'EGARCH Volatility')
plt.legend(loc = 'upper right')
plt.show()
# Print each models BIC
print(f'GJR-GARCH BIC: {gjrgm_result.bic}')
print(f'\nEGARCH BIC: {egarch_result.bic}')
```

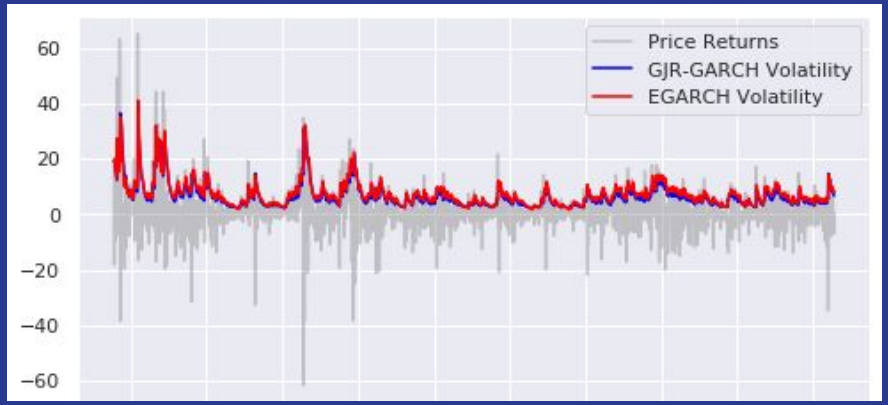
```
                     coef    std err          t      P>|t|  95.0% Conf. Int.
--------------------------------------------------------------------------------
mu            -0.1575      0.509     -0.310      0.757 [ -1.154,  0.839]
                             Volatility Model
                     coef    std err          t      P>|t|  95.0% Conf. Int.
--------------------------------------------------------------------------------
omega          3.3322      4.199      0.794      0.427 [ -4.898, 11.563]
alpha[1]       0.2118      0.589      0.359      0.719 [ -0.944,  1.367]
gamma[1]      -0.2118      0.622     -0.341      0.733 [ -1.431,  1.007]
beta[1]        0.8465      0.217      3.901  9.573e-05 [  0.421,  1.272]
                               Distribution
                     coef    std err          t      P>|t|  95.0% Conf. Int.
--------------------------------------------------------------------------------
nu             3.4663      1.321      2.623  8.710e-03 [  0.876,  6.056]

Covariance estimator: robust
```

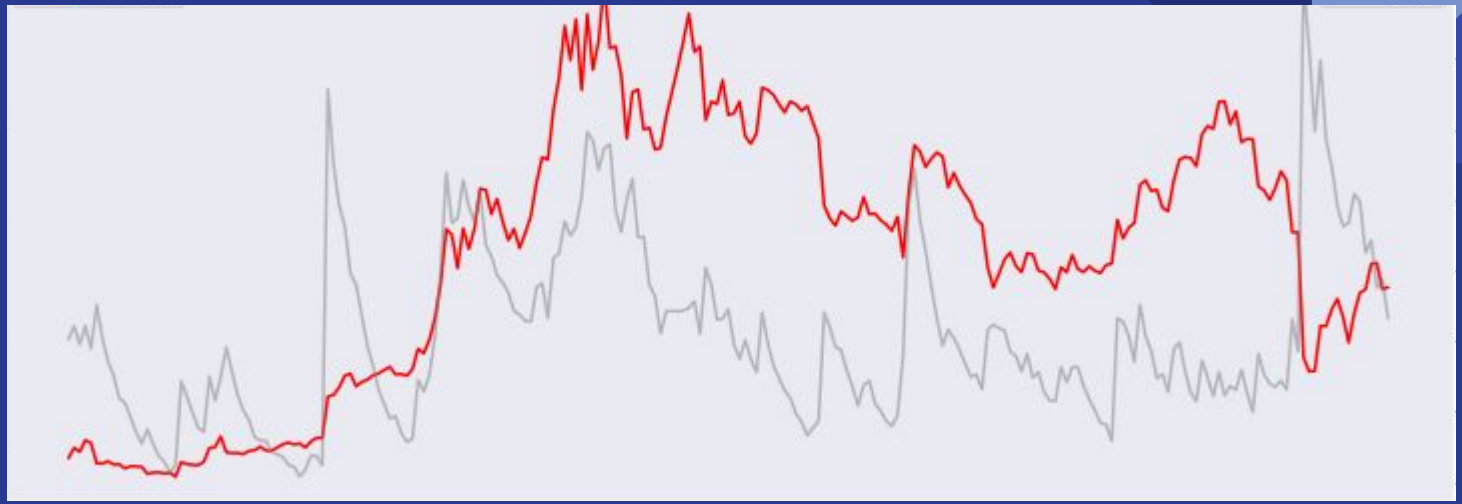
```
                     coef    std err          t      P>|t|  95.0% Conf. Int.
--------------------------------------------------------------------------------
mu            -0.4445      0.545     -0.816      0.415 [ -1.512,  0.623]
                             Volatility Model
                     coef    std err          t      P>|t|       95.0% Conf. Int.
--------------------------------------------------------------------------------
omega          0.3464      1.090      0.318      0.751      [ -1.791,  2.484]
alpha[1]    6.8307e-14  4.436e-02  1.540e-12      1.000 [-8.695e-02,8.695e-02]
gamma[1]   -2.7903e-13  9.091e-02 -3.069e-12      1.000      [ -0.178,  0.178]
beta[1]        1.0000  5.977e-02     16.731  7.744e-63      [  0.883,  1.117]
                               Distribution
                     coef    std err          t      P>|t|  95.0% Conf. Int.
--------------------------------------------------------------------------------
nu             4.1056      1.538      2.669  7.598e-03 [  1.091,  7.120]

Covariance estimator: robust
```

ANALYSIS OF RESULTS

We can analyse from the graphs that whenever the actual volatility of MATIC or WRX changes (upticks or downticks),a similar change can be observed in the graph of EGARCH and GJR GARCH, both the cryptos are slightly better modeled using GARCH with higher BIC score. The underlying grey curve was the actual market and the red curve is the predicted, hence explaining the volatility of cryptos namely MATIC and WRX using Generalized Autoregressive Conditional Heteroskedasticity.







In this project we proposed and compared two different Generalized Autoregressive Conditional Heteroskedasticity models for modelling volatility of value of indian cryptocurrencies MATIC and WRX. The datasets for short term investment for 3 months was used and final results were displayed in terms of actual vs expected volatility, which shows positive correlation amongst them, hence proving that we can use Exponential Generalized Autoregressive Conditional Heteroskedasticity and GJR Generalized Autoregressive Conditional Heteroskedasticity for modelling the indian cryptos. Autocorrelation and Partial Correlations were obtained along with daily volatility and volatility over 3 months. The final results suggest that the aim of the project has been fulfilled. For future scope of work and for Major Project - II, we are planning to take these variants of GARCH models and create forecasting models that can do predictive modelling and give predictions not only for upcoming days but a general trend of the values, finally we want to attain an optimum portfolio approach, in which we can find what cryptos to short and which to hold for getting maximum profits.

REFERENCES

[1]. Gerald P. Dwyer, "The economics of Bitcoin and similar private digital currencies", Journal of Financial Stability, Volume 17, April 2017.

[2]. Paraskevi Katsiampa, "Volatility estimation for Bitcoin: A comparison of GARCH models", Economics Letters, Volume 158, September 2017.

[3]. Dyhrberg, Anne Haubo. 2016, " Bitcoin, gold and the dollar—A GARCH volatility analysis", Finance Research Letters 16: 85–92.

[4]. Balcilar, Mehmet, Elie Bouri, Rangan Gupta, and David Roubaud. 2017, "Can volume predict Bitcoin returns and volatility? A quantiles-based approach", Economic Modelling 64: 74–81.

[5]. Bouri, Elie, Georges Azzi, and Anne Haubo Dyhrberg. 2017, "On the return-volatility relationship in the Bitcoin Market around the price crash of 2013", Economics 11: 1–16.

Thanks Everyone

Kunal Sharma 2K18/MC/060
Rahul Sharma 2K18/MC/087