
**Alexandria National University**

# EGX30 Under the Lens: A Statistical Perspective

## Advisors:

Prof. Dr. Ahmed Tayel

Prof. Dr. Ahmed Yahya

# Group members

| | |
|---|---|
| 1-Asem Wael Nassar | 2204043 |
| 2-Yahya Mahrous Mohammed | 2204112 |
| 3-Mazen Hossam Eldin Ibrahim | 2204062 |
| 4-Omar Abed Ahmed | 2204032 |
| 5-Diaa Eldin Essam | 2204042 |

# *Introduction*

*The EGX30 index fund, representing the top 30 companies listed on the Egyptian Stock Exchange (EGX),it plays a pivotal role in the Egyptian financial landscape. As investors seek to optimize their portfolios, understanding the historical behavior of this index becomes paramount. Our project aims to provide actionable insights to guide investment decisions, emphasizing both descriptive statistics and predictive modeling.*

***Data Collection and Analysis:***

1. Data Source:
    - We collected a substantial dataset comprising 800 data points from the EGX30 index fund. These data points track the indicator's performance from 2021 to 2024.
    - The data was sourced directly from the **Egyptian Stock Exchange (EGX)** website.
2. Statistical Tools Used:
    - We leveraged **MATLAB** for data analysis.
    - Specifically, we utilized the Statistics and Machine Learning Toolbox to explore and model the EGX30 index behavior.

***Key Metrics and Predictions:***

Our analysis focused on the following metrics:

1. Mean (Average):
    - The mean value provides insight into the central tendency of the EGX30 index returns.
2. Variance:
    - Variance measures the dispersion or variability of the index fund's returns.
3. Standard Deviation(ó):
    - Standard deviation quantifies the level of risk associated with investing in the EGX30.
4. Probability Density Function (PDF):
    - The PDF describes the likelihood of different index values occurring.
5. Cumulative Distribution Function (CDF):
    - The CDF shows the probability that the index value falls below a specific threshold.
6. Future Predictions (Next Three Years):

Using our statistical models, we forecast the EGX30 index's behavior for the upcoming three years.

# *1. Mean, Variance, Standard Deviation:*

*1.1.Mean:* The **arithmetic mean** (also known as the average) represents a central value in a set of data. It is calculated by dividing the **sum of all observations** by the **total number of observations**. Mathematically, the formula for the arithmetic mean is:

$$\text{Mean} = \frac{\text{sum of observatio}}{\text{Total number of observation}}$$

Code line: meanValue = mean(values);

*1.2. Variance:* the **variance** measures the **dispersion** or **spread** of values within a dataset. Specifically, it quantifies how much individual data points deviate from the **mean** (average) value.

$$\text{Variance} = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2$$

where:

- (N) is the number of data points.
- (x_i) represents each data point.
- (x') is the mean value.

Code line: variance = var(values);

*1.3. Standard Deviation:* The **standard deviation** is the square root of the variance.

$Standard\ Deviation = \sqrt{Variance}$ Code line: standardDeviation = std(values);

```
>> untitled69
Statistics for Value:
Mean: 15083.94
Variance: 35295459.15
Standard Deviation: 5941.00

Statistics for Max Value:
Mean: 15241.36
Variance: 37168606.80
Standard Deviation: 6096.61

Statistics for Min Value:
Mean: 14937.62
Variance: 33982878.22
Standard Deviation: 5829.48
```

**fig 1**

# *2.PDF, CDF:*

*2.1. PDF:* The **Probability Density Function (PDF)** describes the probability distribution for a random, continuous variable. It provides the likelihood that the value of this variable will fall within a specific range of values that you specify. The formula:

$$fX(x) = dFX(x)\ dx = F\ '\ X(x)$$

code line: [counts, edges] = histcounts(valuesInterp, 'Normalization', 'pdf');

*2.2. CDF:* The **Cumulative Distribution Function (CDF)** completely describes the probability distribution of a real-valued random variable. It represents the probability that the variable will take a value less than or equal to a specific point. The formula:

$$F_X(x) = \int_{-\infty}^{x} f_X(t)dt$$

Code line: y = cdf(x, 10%);

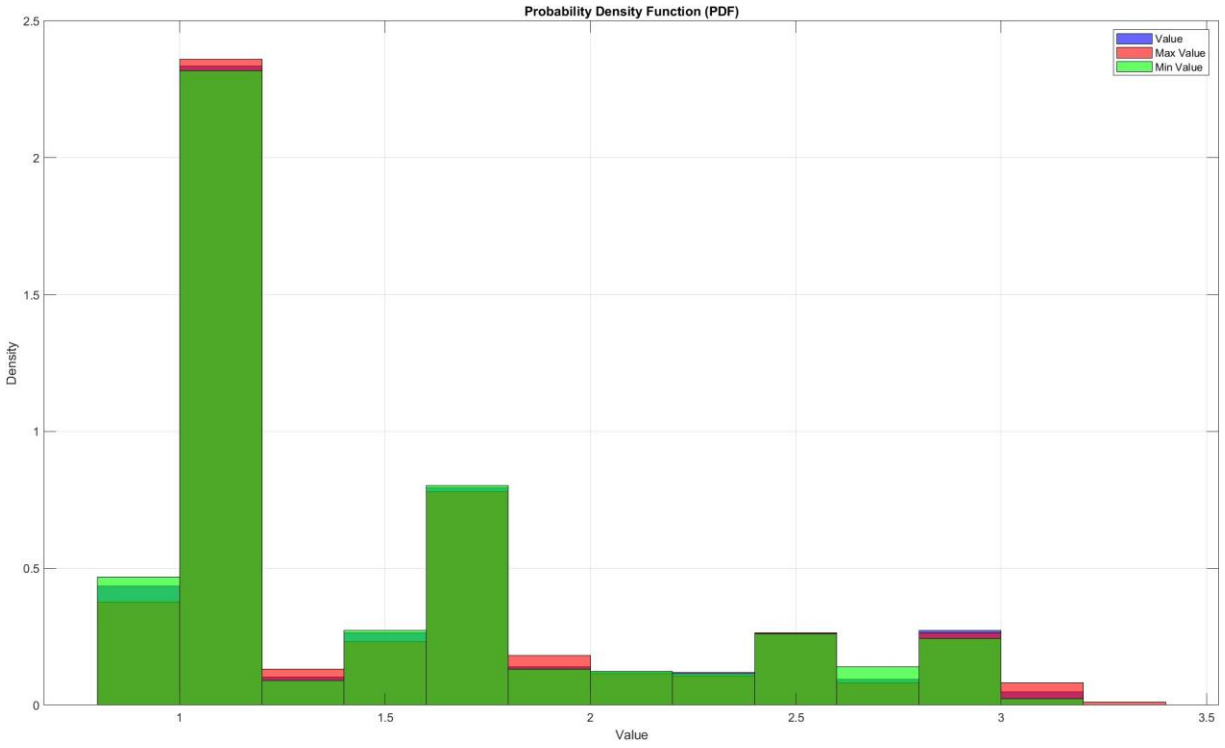


**fig 2.1**

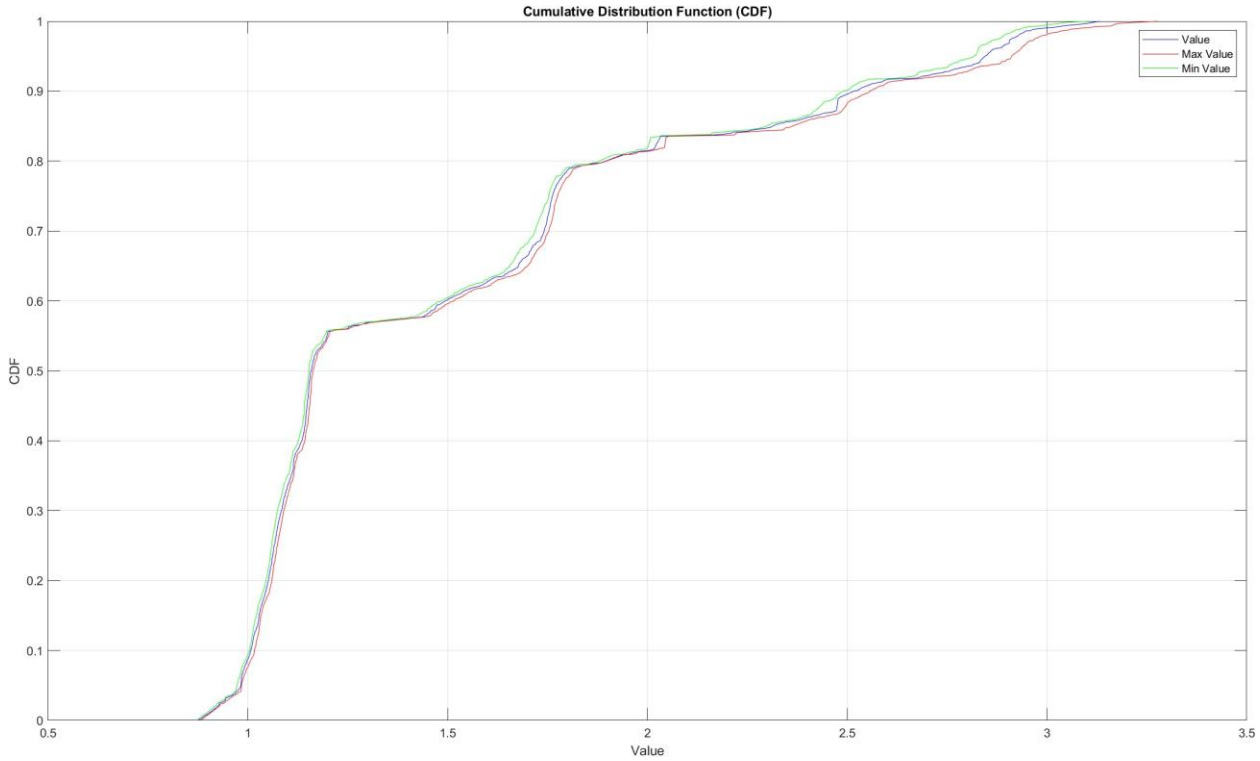
Cumulative Distribution Function (CDF)
Value
Max Value
Min Value
CDF
Value
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0.5
1
1.5
2
2.5
3
3.5

**fig 2.2**

# *3.future prediction(next three years):*

Formula: $P(X=k)=\frac{e-\lambda\lambda k}{(1-e-\lambda)k!}$

Code line: % Time series forecasting with linear regression % Define forecast horizon (3 years) forecastHorizon = 365 * 3; % Prepare the regression model X = (1:length(valuesInterp))'; y = valuesInterp; mdl = fitlm(X, y); % Create future dates and indices for prediction futureX = (length(valuesInterp) + 1 : length(valuesInterp) + forecastHorizon)'; [futureValues, futureCI] = predict(mdl, futureX); % Create future dates futureDates = (fullDates(end) + caldays(1:forecastHorizon))'; % Plot forecasted values figure; plot(fullDates, valuesInterp, 'b', 'DisplayName', 'Historical Data'); hold on; plot(futureDates, futureValues, 'r--', 'DisplayName', 'Forecast'); plot(futureDates, futureCI(:,1), 'k--', 'DisplayName', 'Confidence Interval'); plot(futureDates, futureCI(:,2), 'k--', 'HandleVisibility', 'off');

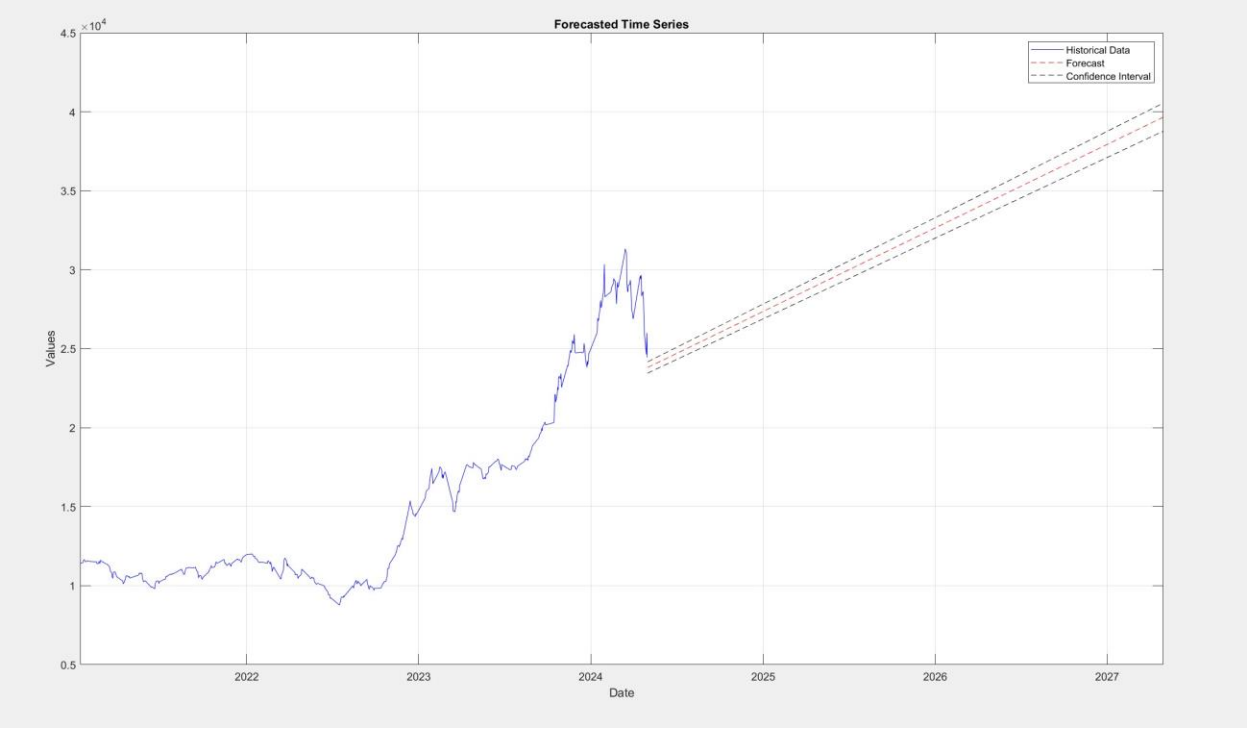


Fig 3

# *4. Code:*

```
% Extract the columns from the imported table
dates = datetime(Book91.Date, 'InputFormat', 'dd/MM/yyyy'); % Convert strings to
datetime
values = Book91.Value;
maxValues = Book91.MaxValue;
minValues = Book91.MinValue;
% Convert datetime to numeric for interpolation
numericDates = datenum(dates);
% Check for non-finite values and remove them
finiteIdx = isfinite(numericDates) & isfinite(values) & isfinite(maxValues) &
isfinite(minValues);
numericDates = numericDates(finiteIdx);
values = values(finiteIdx);
maxValues = maxValues(finiteIdx);
minValues = minValues(finiteIdx);
% Interpolating data to fill gaps if necessary
% Create a full range of dates
fullNumericDates = min(numericDates):max(numericDates);
valuesInterp = interp1(numericDates, values, fullNumericDates, 'linear');
maxValuesInterp = interp1(numericDates, maxValues, fullNumericDates, 'linear');
minValuesInterp = interp1(numericDates, minValues, fullNumericDates, 'linear');
% Convert numeric dates back to datetime
fullDates = datetime(fullNumericDates, 'ConvertFrom', 'datenum');
% Plotting the Time Series
figure;
plot(fullDates, valuesInterp, 'Color', 'b', 'DisplayName', 'Value');
hold on;
plot(fullDates, maxValuesInterp, 'Color', 'r', 'DisplayName', 'Max Value');
plot(fullDates, minValuesInterp, 'Color', 'g', 'DisplayName', 'Min Value');
xlabel('Date');
ylabel('Values');
title('Time Series Plot');
legend('show');
grid on;
hold off;
% Manual calculation of CDF
figure;
[sortedValues, sortIdx] = sort(valuesInterp);
cdfValues = (1:length(sortedValues)) / length(sortedValues);
plot(sortedValues, cdfValues, 'b', 'DisplayName', 'Value');
hold on;
[sortedMaxValues, sortMaxIdx] = sort(maxValuesInterp);
cdfMaxValues = (1:length(sortedMaxValues)) / length(sortedMaxValues);
plot(sortedMaxValues, cdfMaxValues, 'r', 'DisplayName', 'Max Value');
[sortedMinValues, sortMinIdx] = sort(minValuesInterp);
cdfMinValues = (1:length(sortedMinValues)) / length(sortedMinValues);
plot(sortedMinValues, cdfMinValues, 'g', 'DisplayName', 'Min Value');
legend('Value', 'Max Value', 'Min Value');
```

```
xlabel('Value');
ylabel('CDF');
title('Cumulative Distribution Function (CDF)');
grid on;
hold off;
% Manual calculation of PDF using histogram
figure;
histogram(valuesInterp, 'Normalization', 'pdf', 'DisplayName', 'Value', 'FaceColor',
'b');
hold on;
histogram(maxValuesInterp, 'Normalization', 'pdf', 'DisplayName', 'Max Value',
'FaceColor', 'r');
histogram(minValuesInterp, 'Normalization', 'pdf', 'DisplayName', 'Min Value',
'FaceColor', 'g');
xlabel('Value');
ylabel('Density');
title('Probability Density Function (PDF)');
legend('show');
grid on;
hold off;
% Calculate Mean, Variance, and Standard Deviation
meanValue = mean(valuesInterp);
meanMaxValue = mean(maxValuesInterp);
meanMinValue = mean(minValuesInterp);
varValue = var(valuesInterp);
varMaxValue = var(maxValuesInterp);
varMinValue = var(minValuesInterp);
stdValue = std(valuesInterp);
stdMaxValue = std(maxValuesInterp);
stdMinValue = std(minValuesInterp);
% Display the calculated statistics
fprintf('Statistics for Value:\n');
fprintf('Mean: %.2f\n', meanValue);
fprintf('Variance: %.2f\n', varValue);
fprintf('Standard Deviation: %.2f\n\n', stdValue);
fprintf('Statistics for Max Value:\n');
fprintf('Mean: %.2f\n', meanMaxValue);
fprintf('Variance: %.2f\n', varMaxValue);
fprintf('Standard Deviation: %.2f\n\n', stdMaxValue);
fprintf('Statistics for Min Value:\n');
fprintf('Mean: %.2f\n', meanMinValue);
fprintf('Variance: %.2f\n', varMinValue);
fprintf('Standard Deviation: %.2f\n', stdMinValue);
% Time series forecasting with linear regression
% Define forecast horizon (3 years)
forecastHorizon = 365 * 3;
```

```
% Prepare the regression model
X = (1:length(valuesInterp))';
y = valuesInterp;
mdl = fitlm(X, y);


% Create future dates and indices for prediction
futureX = (length(valuesInterp) + 1 : length(valuesInterp) + forecastHorizon)';
[futureValues, futureCI] = predict(mdl, futureX);
% Create future dates
futureDates = (fullDates(end) + caldays(1:forecastHorizon))';
% Plot forecasted values
figure;
plot(fullDates, valuesInterp, 'b', 'DisplayName', 'Historical Data');
hold on;
plot(futureDates, futureValues, 'r--', 'DisplayName', 'Forecast');
plot(futureDates, futureCI(:,1), 'k--', 'DisplayName', 'Confidence Interval');
plot(futureDates, futureCI(:,2), 'k--', 'HandleVisibility', 'off');
xlabel('Date');
ylabel('Values');
title('Forecasted Time Series');
legend('show');
grid on;
hold off;
```