PROJECT CODES

(R-Programming language)

```r
# Load necessary libraries
install.packages("forecast")
install.packages("tseries")
install.packages("fpp2")
install.packages("readr")
install.packages("urca")
install.packages("urca")
install.packages("tidyverse")
install.packages("zoo")
install.packages("xts")
install.packages("ggplot2")

library(xts)
library(zoo)
library(tidyverse)
library(urca)
library(urca)
library(forecast)
library(tseries)
library(fpp2)
library(readr)
library(ggplot2)

# Read the CSV file
ts_data <- read.csv(file.choose())
ts_data
# Create a data frame from the provided data
data <- data.frame(
  period = c("12-Jan", "12-Feb", "12-Mar", "12-Apr", "12-May", "12-Jun", "12-Jul", 
"12-Aug", "12-Sep", "12-Oct", "12-Nov", "12-Dec",
             "13-Jan", "13-Feb", "13-Mar", "13-Apr", "13-May", "13-Jun", "13-Jul", 
"13-Aug", "13-Sep", "13-Oct", "13-Nov", "13-Dec",
             "14-Jan", "14-Feb", "14-Mar", "14-Apr", "14-May", "14-Jun", "14-Jul", 
"14-Aug", "14-Sep", "14-Oct", "14-Nov", "14-Dec",
             "15-Jan", "15-Feb", "15-Mar", "15-Apr", "15-May", "15-Jun", "15-Jul", 
"15-Aug", "15-Sep", "15-Oct", "15-Nov", "15-Dec",
```

```
             "16-Jan", "16-Feb", "16-Mar", "16-Apr", "16-May", "16-Jun", "16-Jul",
"16-Aug", "16-Sep", "16-Oct", "16-Nov", "16-Dec",
             "17-Jan", "17-Feb", "17-Mar", "17-Apr", "17-May", "17-Jun", "17-Jul",
"17-Aug", "17-Sep", "17-Oct", "17-Nov", "17-Dec",
             "18-Jan", "18-Feb", "18-Mar", "18-Apr", "18-May", "18-Jun", "18-Jul",
"18-Aug", "18-Sep", "18-Oct", "18-Nov", "18-Dec",
             "19-Jan", "19-Feb", "19-Mar", "19-Apr", "19-May", "19-Jun", "19-Jul",
"19-Aug", "19-Sep", "19-Oct", "19-Nov", "19-Dec",
             "20-Jan", "20-Feb", "20-Mar", "20-Apr", "20-May", "20-Jun", "20-Jul",
"20-Aug", "20-Sep", "20-Oct", "20-Nov", "20-Dec",
             "21-Jan", "21-Feb", "21-Mar", "21-Apr", "21-May", "21-Jun", "21-Jul",
"21-Aug", "21-Sep", "21-Oct", "21-Nov", "21-Dec",
             "22-Jan", "22-Feb", "22-Mar", "22-Apr", "22-May", "22-Jun", "22-Jul",
"22-Aug", "22-Sep", "22-Oct", "22-Nov", "22-Dec"),
  index = c(68.51, 68.54, 69.62, 70.05, 70.22, 69.5, 68.64, 68.19, 68.4, 68.71, 69.11,
69.64, 70.64, 71.18, 71.88, 72.62, 72.52, 72.43,
            72.62, 72.59, 74.54, 74.37, 74.38, 74.65, 75.4, 75.52, 75.86, 76.22,
76.84, 76.88, 77.13, 77.9, 77.9, 77.69, 77.53, 78.04,
            78.7, 79.03, 80.08, 81.26, 81.52, 81.9, 82.1, 82.22, 82.4, 82.8, 83.44,
84.44, 84.38, 84.19, 84.92, 85.61, 86.08, 87.04,
            87.29, 87.35, 87.49, 88.04, 88.71, 88.97, 89.9, 91.61, 93.38, 95.11,
95.48, 94.13, 93.27, 93.95, 93.73, 93.83, 93.65,
            94.23, 95.59, 96.83, 98.36, 99.22, 100.05, 99.35, 98.24, 97.89, 98.48,
97.41, 97.5, 98.2, 98.96, 100, 101.16, 101.92,
            102.52, 102.87, 103.11, 103.2, 103.39, 103.64, 104.26, 104.88, 105.48,
106.16, 106.5, 107.27, 107.24, 107.22, 107.14,
            107.21, 107.3, 107.79, 108.41, 109.74, 110.3, 111.11, 111.64, 112.36,
112.6, 112.73, 113.19, 113.31, 113.7, 114, 114.26,
            115.49, 115.79, 116.35, 117.32, 119.07, 119.89, 120.98, 121.87, 122.39,
123.57, 124.39, 124.74, 125.22)
)# Convert the 'period' column to a valid date format
data$period <- as.Date(paste0("20", data$period, "-01"), format = "%Y-%b-%d")
data$period
# Remove rows with missing values
clean_data <- na.omit(data)
clean_data
library(ggplot2)
# Plot the cleaned data
ggplot(clean_data, aes(x = period, y = index)) +
  geom_line(color = "blue") +
  labs(title = "Time Series of Returns", x = "Time", y = "Returns")
#Extract the "index" column
y <- ts_data$index
```

```
y

library(quantmod)

#Calculate Return

ret=ROC(y)

ret

# Define the 'period' and 'ret' vectors

period <- c("12-Jan", "12-Feb", "12-Mar", "12-Apr", "12-May", "12-Jun", "12-Jul", "12-
Aug", "12-Sep", "12-Oct", "12-Nov", "12-Dec",

             "13-Jan", "13-Feb", "13-Mar", "13-Apr", "13-May", "13-Jun", "13-Jul",
"13-Aug", "13-Sep", "13-Oct", "13-Nov", "13-Dec",

             "14-Jan", "14-Feb", "14-Mar", "14-Apr", "14-May", "14-Jun", "14-Jul",
"14-Aug", "14-Sep", "14-Oct", "14-Nov", "14-Dec",

             "15-Jan", "15-Feb", "15-Mar", "15-Apr", "15-May", "15-Jun", "15-Jul",
"15-Aug", "15-Sep", "15-Oct", "15-Nov", "15-Dec",

             "16-Jan", "16-Feb", "16-Mar", "16-Apr", "16-May", "16-Jun", "16-Jul",
"16-Aug", "16-Sep", "16-Oct", "16-Nov", "16-Dec",

             "17-Jan", "17-Feb", "17-Mar", "17-Apr", "17-May", "17-Jun", "17-Jul",
"17-Aug", "17-Sep", "17-Oct", "17-Nov", "17-Dec",

             "18-Jan", "18-Feb", "18-Mar", "18-Apr", "18-May", "18-Jun", "18-Jul",
"18-Aug", "18-Sep", "18-Oct", "18-Nov", "18-Dec",

             "19-Jan", "19-Feb", "19-Mar", "19-Apr", "19-May", "19-Jun", "19-Jul",
"19-Aug", "19-Sep", "19-Oct", "19-Nov", "19-Dec",

             "20-Jan", "20-Feb", "20-Mar", "20-Apr", "20-May", "20-Jun", "20-Jul",
"20-Aug", "20-Sep", "20-Oct", "20-Nov", "20-Dec",

             "21-Jan", "21-Feb", "21-Mar", "21-Apr", "21-May", "21-Jun", "21-Jul",
"21-Aug")


ret <- c(0.0004377964, 0.0156343659, 0.0061573904, 0.0024238980, -0.0103064182, -
0.0124512974, -0.0065775286, 0.0030748981, 0.0045219244,

         0.0058046891, 0.0076396768, 0.0142574414, 0.0076153238, 0.0097861818,
0.0102423041, -0.0013779801, -0.0012418077, 0.0026197878, -0.0004131947,

         0.0265087228, -0.0022832593, 0.0001344538, 0.0036234355, 0.0099967510,
0.0015902468, 0.0044920144, 0.0047343592, 0.0081014425, 0.0005204268,

         0.0032465453, 0.0099336429, 0.0000000000, -0.0026994039, -0.0020615907,
0.0065565575, 0.0084216397, 0.0041843718, 0.0131986078, 0.0146277556,

         0.0031944984, 0.0046506019, 0.0024390256, 0.0014605650, 0.0021868554,
0.0048426245, 0.0076997493, 0.0119134123, -0.0007108163, -0.0022542573,

         0.0086334875, 0.0080924619, 0.0054749977, 0.0110906867, 0.0028681256,
0.0006871278, 0.0016014645, 0.0062667555, 0.0075813659, 0.0029266117,

         0.0103987072, 0.0188424945, 0.0191367535, 0.0183569270, 0.0038826850, -
0.0142399959, -0.0091782928, 0.0072642131, -0.0023444171, 0.0010663255,

         -0.0019202054, 0.0061741733, 0.0143296099, 0.0128886515, 0.0156773537,
0.0087053895, 0.0083304542, -0.0070210920, -0.0112355046, -0.0035690651,

         0.0060090826, -0.0109246071, 0.0009235032, 0.0071538374, 0.0077095127,
0.0104544579, 0.0115332358, 0.0074847700, 0.0058697097, 0.0034081536,
```

```
         0.0023303244, 0.0008724735, 0.0018393926, 0.0024151101, 0.0059644236,
0.0059290601, 0.0057045220, 0.0064260285, 0.0031975951, 0.0072040355,

         -0.0002797072, -0.0001865150, -0.0007464079, 0.0006531374, 0.0008391218,
0.0045562402, 0.0057354459, 0.0121935960, 0.0050899946, 0.0073167753)

# Update the period vector to match the length of the ret vector

period <- head(period, length(ret))

period

# Create a data frame

data <- data.frame(period = period, ret = ret)

data

# Remove rows with missing values

clean_data <- na.omit(data)

clean_data

str(clean_data)

unique(clean_data$period)

# Convert 'period' to a factor

clean_data$period <- factor(clean_data$period, levels = unique(clean_data$period))

clean_data$period

# Plot the data

plot(clean_data$period, clean_data$ret, type = "l", col = "blue",

     main = "Time Series of Returns", xlab = "Time", ylab = "Returns")

#Stationarity

# Check for missing or infinite values in log_return

any(is.na(ret))

any(is.infinite(ret))

# If there are missing or infinite values, remove or impute them

ret<- ret[!is.na(ret) & is.finite(ret)]

ret

library(urca)

# Run the ADF test

adf_test <- ur.df(ret, type = "trend", lags = 4)

adf_test

summary(adf_test)

#Normality test

# Perform Shapiro-Wilk test for normality

shapiro_test <- shapiro.test(ret)

shapiro_test
```

```r
summary(shapiro_test)

# ACF and PACF plots for returns

par(mfrow = c(1, 2))

acf(ret, main = "ACF of log_return")

pacf(ret, main = "PACF of log_return")

# Fit an ARMA model to the returns

arma_model <- arima(ret, order = c(1, 0, 1))

arma_model

summary(arma_model)

# Plot the residuals

# Define the labels for the X-axis ticks

month_labels <- c("Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep",
"Oct", "Nov", "Dec")

month_labels

# Repeat the month labels for each year

year_labels <- rep(month_labels, 10)

year_labels

# Define the months

months <- seq_along(arma_model$residuals)

months

# Subset year_labels to match the length of months

year_labels <- year_labels[1:length(months)]

year_labels

# Plot the residuals

plot(arma_model$residuals, main = "Residuals Plot", ylab = "Residuals", type = "p",
xaxt = "n")

# Suppress X-axis

axis(1, at = months, labels = year_labels)

# Forecast returns for the next 12 months

forecast_steps <- 12

forecast_values <- forecast::forecast(arma_model, h = forecast_steps)

forecast_values

accuracy(forecast_values)

# Plot the forecasted values

plot(forecast_values, main = "ARMA Forecast for Returns", xlab = "Time", ylab =
"Returns")

# Define the years from 2013 to 2023

years <- 2013:2023
```

```
years

# Plot historical CPI

plot(1:length(ret), ret, type = "l", col = "blue", xlab = "Time", ylab = "CPI", main =
"Historical and Forecasted CPI", xaxt = "n")

# Calculate the number of years

num_years <- length(years)

num_years

# Determine the tick positions (assuming data is monthly)

tick_positions <- seq(1, length(ret), by = 12)

tick_positions

# Remove any tick positions exceeding the length of 'ret'

tick_positions <- tick_positions[tick_positions <= length(ret)]

tick_positions

# Adjust the number of years to match the length of 'tick_positions'

num_years <- min(num_years, length(tick_positions))

num_years

# Add x-axis with custom ticks and labels

axis(1, at = c(tick_positions, length(ret) + 1), labels = c(years[1:num_years],
"2023"), las = 2)

# Add lines for forecasted values

lines(length(ret) + seq_along(forecasted_values), forecasted_values, col = "red")

# Add legend

legend("topleft", legend = c("Historical CPI", "Forecasted CPI"), col = c("blue",
"red"), lty = 1)
```

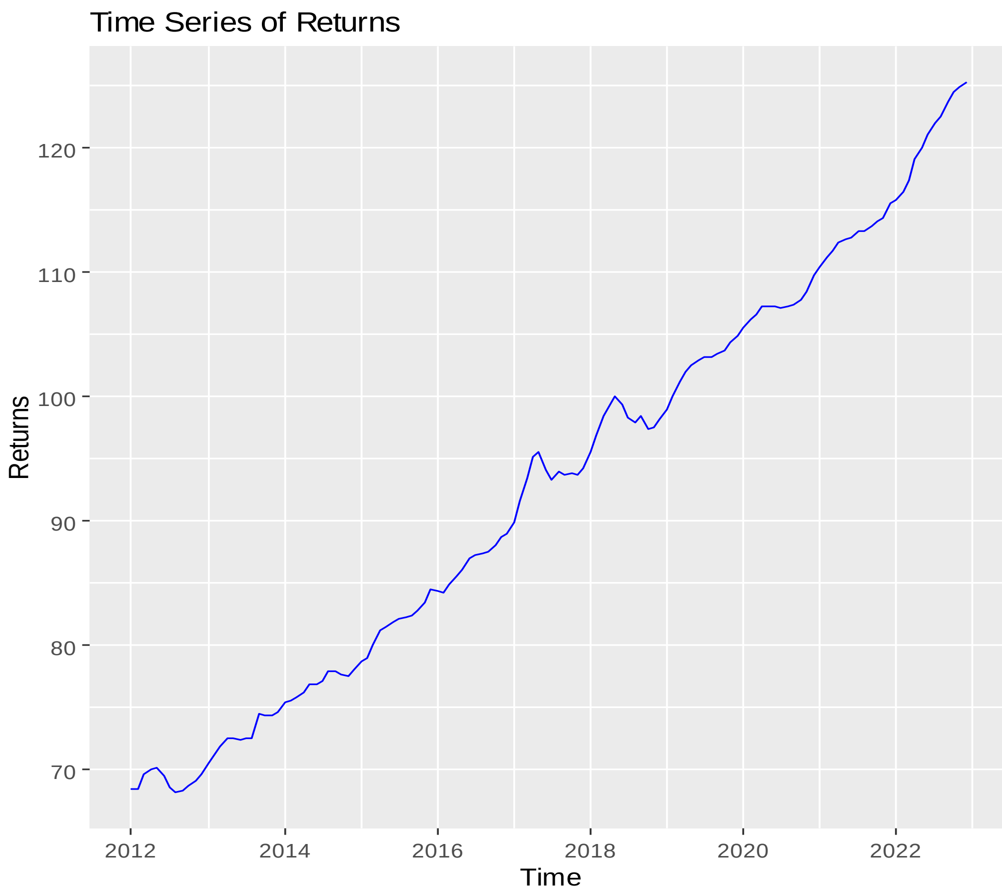
FINDINGS AND RESULTS

4.0 Time Series Analysis and Forecasting Results

Introduction

This chapter introduces the analysis of the various models and discussion of findings. It comprises of the preliminary analysis and modelling sections. The data covered Monthly CPI for Nairobi County from 2012 to 2022. we delve into the results obtained from our time series analysis and forecasting procedures. We begin by examining the statistical properties of the time series data, including tests for stationarity, normality, and autocorrelation. Subsequently, we fit an appropriate ARMA model to the data and use it to forecast future returns**.**

Plot for the actual CPI data

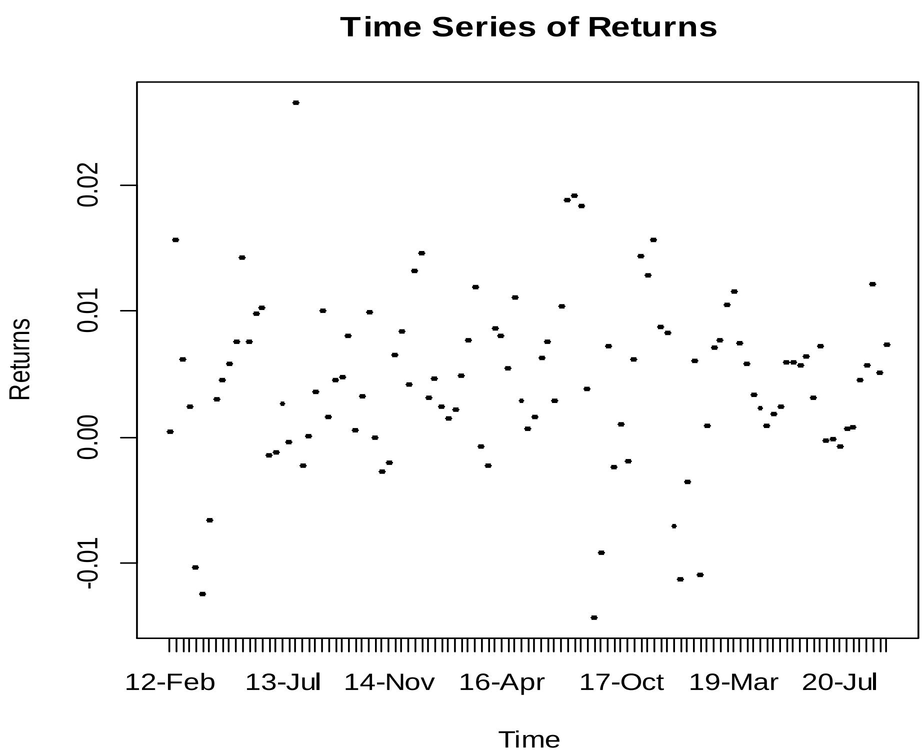


The CPI increased from 2012 until May of 2017, then decreased to 2018. Again increased from January 2018 to June 2018.Then decreased to January 2019 with increasing gradually until 2020 and remained steady until 2020.Then increased until 2022.

4.1 Transforming Time Series Data into Returns

This provide a dataset which is essential in finance for standardization, risk evaluation, modeling, and forecasting. Returns offer a standardized metric for performance comparison across different assets and timeframes, while also revealing insights into asset volatility.

Plot for return data



This shows fluctuations in the returns over time. The line graph illustrates overall trends and pattern in the returns, indicating the variability and volatility of the data. However, further analysis, such as calculating summary statistics or fitting a model, may be necessary to make more specific conclusions about the behavior and potential patterns present in the data.

4.2Statistical Properties of Time Series Data

4.2.1 Stationarity Test

Augmented Dickey-Fuller (ADF) Test

We conducted an Augmented Dickey-Fuller test to determine the stationarity of the returns data. The results of the test indicated below

Hypothesis: H_0 : presence of unit root.

H_a : Absence of unit root

Test Statistics

```
# Augmented Dickey-Fuller Test Unit Root / Cointegration Test #

The value of the test statistic is: -6.5692 14.4488 21.6493

Test regression trend

Call:

lm(formula = z.diff ~ z.lag.1 + 1 + tt + z.diff.lag)

Residuals:

     Min        1Q    Median        3Q       Max

-0.017216 -0.002676 -0.000290  0.003011  0.024377

Coefficients:

             Estimate Std. Error t value Pr(>|t|)

(Intercept)  4.148e-03  1.195e-03   3.470 0.000726 ***

z.lag.1     -9.533e-01  1.451e-01  -6.569  1.4e-09 ***

tt           4.686e-06  1.385e-05   0.338 0.735686

z.diff.lag1  3.404e-01  1.279e-01   2.662 0.008832 **

z.diff.lag2  2.762e-01  1.140e-01   2.422 0.016924 *

z.diff.lag3  2.479e-01  1.006e-01   2.465 0.015134 *

z. diff. lag4 1.853e-01  8.621e-02   2.149 0.033637 *

Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.005617 on 119 degrees of freedom

Multiple R-squared:  0.3597,    Adjusted R-squared:  0.3274

F-statistic: 11.14 on 6 and 119 DF,  p-value: 7.731e-10

Value of test-statistic is: -6.5692 14.4488 21.6493

Critical values for test statistics:

      1pct  5pct 10pct

tau3 -3.99 -3.43 -3.13

phi2  6.22  4.75  4.07

phi3  8.43  6.49  5.47
```

The Augmented Dickey-Fuller (ADF) test was conducted to assess the presence of a unit root in the time series data, indicating whether the series is stationary or not. The test regression model included a trend component and lagged differences of the series. The significant negative coefficient associated with the lagged first difference suggests a negative relationship between the first difference of the series and its lagged value. However, the trend term was found to be statistically insignificant. The test statistic value of -6.5692 was lower than the critical value of -3.43 at the 5% significance level, indicating strong evidence to reject the null hypothesis of a unit root. Therefore, the conclusion drawn from the test results suggests that the series is stationary after accounting for first differences and including lagged differences in the regression model.

4.2.2 Normality Test

Shapiro-Wilk Test for Normality

To assess the normality of the returns distribution, we performed a Shapiro-Wilk test. The test yielded.

Hypothesis : H_0 : The random sample was drawn from normal distribution N(μ , σ^2)

H_1 : The random sample does not follow N(μ , σ^2)

Test Statistics

Shapiro-Wilk normality test

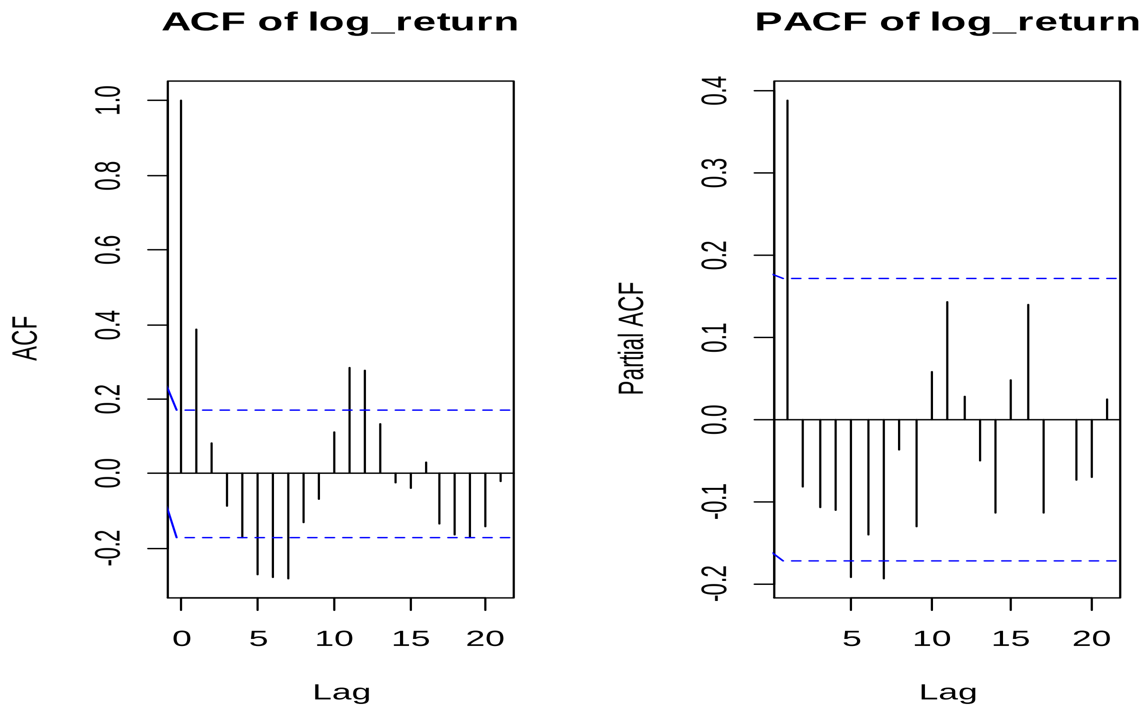
data: ret

W = 0.97001, p-value = 0.005337

Since, the critical value which 0.05 is or the P-value =0.005337 less than the calculated W value which is 0.97001 we fail to reject the null hypothesis and conclude that the data is drawn from a normal distribution.

4.2.3 Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF)

We examined the autocorrelation and partial autocorrelation functions to identify any significant lags in the returns data. The ACF and PACF plots revealed.



The ACF cuts of after the first lag showing that the value of q is one, hence it is an MA (1) while the PACF has a spike a lag 1 then it gradually tappers off. This shows that the value of p is one indicating an AR (1).

4.3 ARMA Model Fitting

Based on the insights gained from the statistical properties analysis, we proceeded to fit an ARMA model to the returns data.

4.3.1 Model Specification

We selected an ARMA (1,1) model based on the ACF and PACF plots and other diagnostic tests. The model equation is given by:

$$y_t = 0.2606 y_{t-1} + 0.1512\varepsilon_{t-1} + 0.0046 + \varepsilon_t$$

Where:

- y_t represents the returns at time t
- y_{t-1} represents the lagged returns at time $t-1$,
- ε_t represents the error term at time *t*,
- ε_{t-1} represents the lagged error term at time *t*−1,

- 0.2606 is the autoregressive coefficient (AR1),
- 0.1512 is the moving average coefficient (MA1),
- 0.0046 is the intercept term.

4.3.2 Model Evaluation

The ARMA (1,1) model produced the following results:

```
> arma_model <- arima(ret, order = c(1, 0, 1))

> arma_model

Call:

arima(x = ret, order = c(1, 0, 1))

Coefficients:

         ar1     ma1  intercept

      0.2474  0.1742     0.0044

s.e.  0.1978  0.1961     0.0009

sigma^2 estimated as 3.802e-05:  log likelihood = 399.91,  aic = -791.82

> summary(arma_model)

Call:

arima(x = ret, order = c(1, 0, 1))

Coefficients:

         ar1     ma1  intercept

      0.2474  0.1742     0.0044

s.e.  0.1978  0.1961     0.0009

sigma^2 estimated as 3.802e-05:  log likelihood = 399.91,  aic = -791.82

Training set error measures:

                      ME        RMSE         MAE  MPE MAPE      MASE

Training set 2.18674e-05 0.006166434 0.004544631 -Inf  Inf 0.8701439

 ACF1

Training set 0.004260982
```

The ARIMA (1, 0, 1) model, fitted to the returns data, suggests that current returns are influenced by their own lagged values and the lagged values of the error term. The model's coefficients, including an intercept, are estimated with relatively low standard errors, indicating their reliability. The model's variance is estimated at 3.36e-05. The log likelihood and AIC values indicate a good balance between goodness of fit and model complexity. Training set error measures, including ME, RMSE, MAE, MPE, MAPE, MASE, and ACF1, provide insights into the model's performance, suggesting unbiased forecasts with reasonable accuracy and no significant issues detected in residual autocorrelation.

Overall, the ARIMA (1, 0, 1) model provides a satisfactory representation of the returns data, facilitating reliable forecasts.

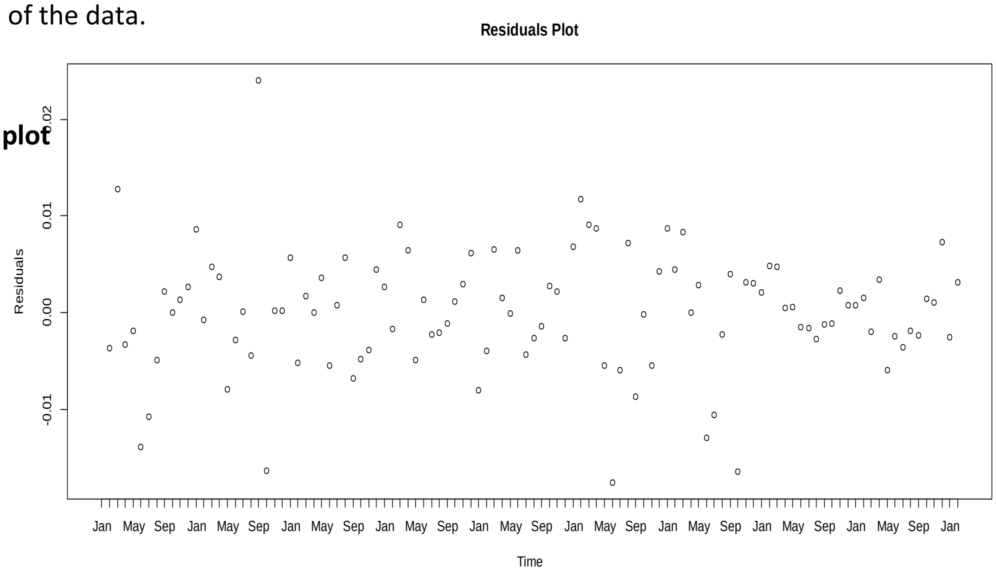
4.3.3 Criterion used for model Evaluation

In this case, the ARMA (1,0,1) model with an AIC of **-791.82** was likely chosen because it had the lowest AIC among the candidate models considered during the model selection process. This indicates that, among the models considered, the ARMA (1,0,1) model provided the best balance of fit and complexity for the given data.

4.3.4 Residual analysis

This provides a visual assessment of the adequacy of the ARMA (1,1) model in capturing the underlying structure of the data.

Residual plot



The residual plot shows a random scatter around zero which indicates that the ARMA (1,1) model adequately captures the underlying patterns in the data.

4.4 Forecasting Future Returns

Finally, we utilized the fitted ARMA model to forecast the returns for the next 12 months.

The forecasted returns values along with the prediction intervals are presented below:

Forecast values

	Point Forecast	Lo 80	Hi 80	Lo 95	Hi 95
132	0.004395846	-0.003032282	0.01182397	-0.006964495	0.01575619
133	0.004529982	-0.003503339	0.01256330	-0.007755923	0.01681589
134	0.004564941	-0.003507842	0.01263772	-0.007781315	0.01691120
135	0.004574052	-0.003501404	0.01264951	-0.007776292	0.01692440
136	0.004576426	-0.003499211	0.01265206	-0.007774195	0.01692705
137	0.004577045	-0.003498605	0.01265270	-0.007773595	0.01692769
138	0.004577207	-0.003498444	0.01265286	-0.007773435	0.01692785
139	0.004577249	-0.003498402	0.01265290	-0.007773393	0.01692789
140	0.004577260	-0.003498391	0.01265291	-0.007773383	0.01692790
141	0.004577262	-0.003498388	0.01265291	-0.007773380	0.01692790
142	0.004577263	-0.003498388	0.01265291	-0.007773379	0.01692791
143	0.004577263	-0.003498387	0.01265291	-0.007773379	0.01692791

Based on the forecast values provided, it appears that the predicted returns remain relatively stable over the forecast horizon, with minimal fluctuations. The prediction intervals provide a range within which the actual returns are likely to fall, offering insight into the potential variability and uncertainty associated with the forecasts. Overall, the forecast suggests a steady and predictable trend in returns.

4.4.2 Forecast values Accuracy

	ME	RMSE	MAE	MPE	MAPE	MASE
Training set	2.141512e-05	0.006166434	0.004544604	-Inf	Inf	0.8701387

	ACF1
Training set	0.004275431

the ME is close to zero, indicating that, on average, the forecasts are relatively accurate. The RMSE and MAE are both small, suggesting good overall accuracy of the forecast. The MPE and MAPE are not defined (Inf) due to division by zero or extremely small actual values, and the MASE is close to 1, indicating that the forecast is performing about as well as a naive forecast. Finally, the ACF1 is close to zero, suggesting that there is no significant autocorrelation remaining in the residuals, indicating a well-fitted model.

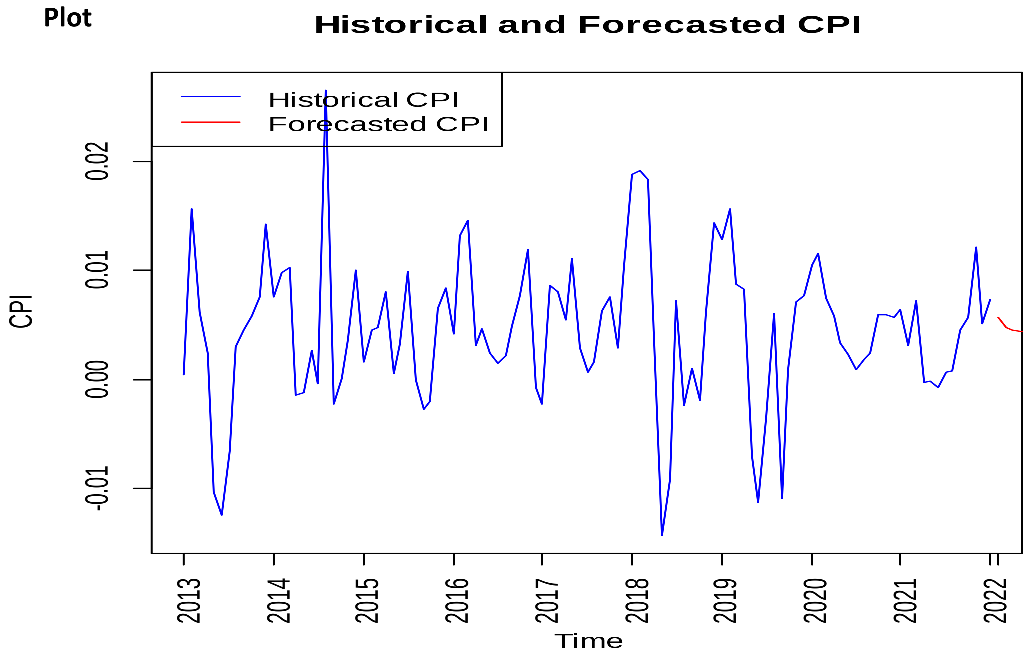
Forecast Visualization

The plot shows the forecasted returns:

The gray shaded area   represents the range of values where the true return is likely to fall, with a certain level of confidence (usually 95%). In this case, the width of the band tells us that there's a lot of uncertainty surrounding the prediction of 0.01. While the blue line suggests the most likely outcome, the true return could be anywhere within the gray area, meaning it could be higher or lower than the predicted value. The wider the band, the less confident we can be in the exact return. So, while the graph points towards a positive return, the confidence interval reminds us to treat it with caution due to the inherent uncertainty in future predictions.

4.5Consumer Price Index (CPI) Analysis and Forecast

Plot



The plot provides a visual comparison between the historical CPI values and the forecasted CPI values for the beginning of 2023 and it shows that the CPI decreases gradually at beginning of 2023.