TIME_SERIES_ANALYSIS

21122024_JAGANNATH V V

17 November 2022

==

== INTRODUCTION:

Wheat is the main cereal crop in India. The total area under the crop is about 29.8 million hectares in the country. The production of wheat in the country has increased significantly from 75.81 million MT in 2006-07 to an all time record high of 94.88 million MT in 2011-12. The productivity of wheat which was 2602 kg/hectare in 2004-05 has increased to 3140 kg/hectare in 2011-12. The major increase in the productivity of wheat has been observed in the states of Haryana, Punjab and Uttar Pradesh.

The production and productivity of Wheat crop were quite low, when India became independent in 1947. The production of Wheat was only 6.46 million tonnes and productivity was merely 663 kg per hectare during 1950-51, which was not sufficient to feed the Indian population. The Country used to import Wheat in large quantities for fulfilling the needs of our people from many countries like USA under PL-480. The reasons of low production and productivity of Wheat at that time was (a) the tall growing plant habit resulting in lodging, when grown under fertile soils, (b) the poor tillering and low sink capacity of the varieties used, (c) higher susceptibility to diseases, (d) the higher sensitivity to thermo & photo variations, etc., resulting in poor adaptability, and (e) longer crop duration resulting in a long exposure of plants to the climatic variations and insect pest / disease attacks.

MODELS:

1)Auto-Regressive Integrated Moving Average (ARIMA) Model

The ARIMA model analyzes and forecasts equally spaced univariate time series data. An ARIMA model predicts a value in a response time series as a linear combination of its past importance. The ARIMA approach was first popularized by Box and Jenkins (1976), and ARIMA models are often referred to as Box-Jenkins models. This process is referred to as ARIMA (p, d, q), where p and q guide the number of AR and MA terms, and d refers to the order of differencing required for making the series a stationary

2)Holts Exponential smoothing

Holt’s two-parameter model, also known as linear, exponential smoothing, is a popular smoothing model for forecasting data with trends. Holt’s model has three separate equations that work together to generate a final forecast. The first is a basic smoothing equation that directly adjusts the last smoothed value for the previous trend. The trend itself is updated over time through the second equation, where the movement is expressed as the difference between the last two smoothed values.

3)KNN

KNN algorithm can be used for both classification and regression problems. The KNN algorithm uses 'feature similarity' to predict the values of any new data points. This means that the new point is assigned a value based on how closely it resembles the points in the training set.

OBJECTIVE:

A comparative study is performed to forecast the WHEAT production in India for the next five years using ARIMA and Holts exponential ,KNN.

DATA SOURCE:

A time-series approach is used in the yearly WHEAT production data, in thousand tones from 1951 to 2022, as per the availability the time series data related to the annual WHEAT productions data collected from Reserve Bank of India Government ministry.

ANALYSIS:

A comparative study is performed in this paper. We imported the data, and then we could see that there existed 70 observations.

```
library(readxl)
j<-read_excel("C:/Users/jagan/OneDrive/Desktop/time2.xlsx")
j
```

```
## # A tibble: 72 × 7 ## YEAR RICE WHEAT `Coarse Cereals` `Total Cereals` Pulses
`Total Foodgrains` ## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> ## 1 1951 206. 64.6
154. 424. 84.1 508.
## 2 1952 213 61.8 161. 436. 84.2 520. ## 3 1953 229 75 196. 500. 91.9 592 ## 4
1954 282. 80.2 230. 592 106. 698. ## 5 1955 252. 90.4 228. 571. 110. 680. ## 6 1956
276. 87.6 195. 558. 110. 668. ## 7 1957 290. 94 199. 583. 116. 699. ## 8 1958 255.
79.9 212. 548. 95.6 643. ## 9 1959 308. 99.6 232. 640. 132. 771. ## 10 1960 317. 103.
229. 649. 118 767. ## # … with 62 more rows
```

# Remove Columns by Index j <- j[,-c(2,4,5,6,7)] j

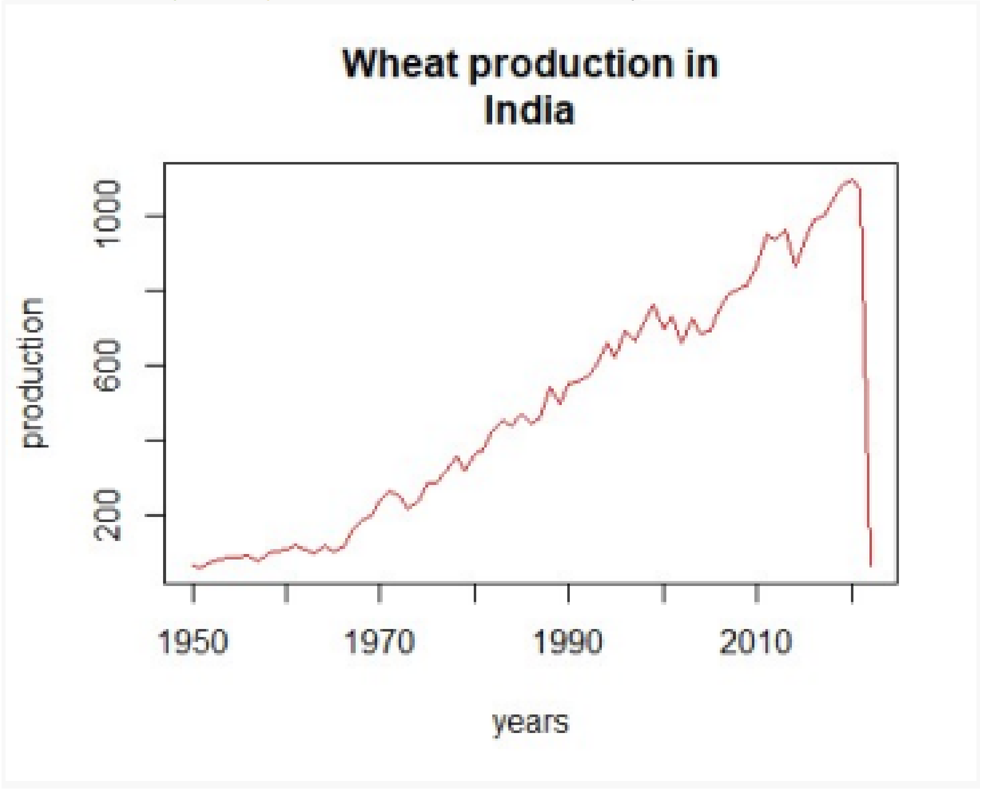
A tibble: 72 × 2 ## YEAR WHEAT ## <dbl> <dbl> ## 1 1951 64.6 ## 2 1952 61.8 ## 3 1953 75 ## 4 1954 80.2 ## 5 1955 90.4 ## 6 1956 87.6 ## 7 1957 94 ## 8 1958 79.9 ## 9 1959 99.6 ## 10 1960 103. ## # … with 62 more rows

library(tseries)

## Registered S3 method overwritten by 'quantmod': ## method from ## as.zoo.data.frame zoo

wheat_production=ts(j$WHEAT,start=1950,end=2022,frequency = 1)

ts.plot(wheat_production,main='Wheat production in India',xlab='years',ylab='production',col='red')



## Checking whether data is nonseasonal or not:

adf.test(wheat_production)

```
##
## Augmented Dickey-Fuller Test ##
## data: wheat_production ## Dickey-Fuller = -2.1116, Lag order = 4, p-value = 0.5299
## alternative hypothesis: stationary
```

Since p values is 0.5 which greater than 0.05.Then we can say that it is non stationary

```
diff_wheat_production=diff(wheat_production)
adf.test(diff_wheat_production)
```

```
##
## Augmented Dickey-Fuller Test  ##
## data: diff_wheat_production  ## Dickey-Fuller = -0.75779, Lag order = 4, p-value =
0.9609  ## alternative hypothesis: stationary
```

```
length(wheat_production)
```

```
## [1] 73
wheat_production_1=wheat_production[1:70]
wheat_production_1=ts(wheat_production, start=1950,end=2020,frequency = 1)
```

The non-stationary data is changed into stationary data and we have performed the differencing.

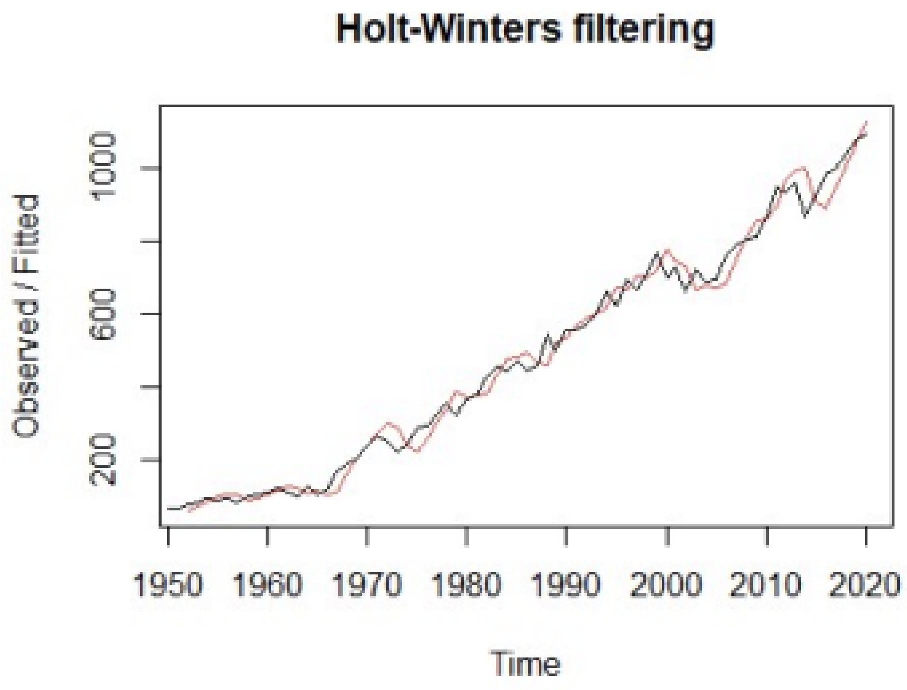
. . .

Now we are modeling our data using Holts method.

HOLTS EXPONENTIAL SMOOTHING

```
#install.packages("astsa")
```

```
library(astsa)
wheat_prod=HoltWinters(wheat_production_1,beta= TRUE,gamma = FALSE)
plot(wheat_prod)
```



```
library(forecast)
```

```
##
## Attaching package: 'forecast'
```

```
## The following object is masked from 'package:astsa':  ##
## gas
forecast_data=forecast(wheat_prod,h=2)
forecast_data
```

```
##  Point  Forecast  Lo  80  Hi  80  Lo  95  Hi  95   ##  2021  1156.443  1104.079  1208.807
1076.360 1236.527  ## 2022 1198.264 1129.595 1266.933 1093.244 1303.284
```

Checking accuracy measures

```
#install.packages("Metrics")
```

```
library(Metrics)
```

```
##
## Attaching package: 'Metrics'
```

```
## The following object is masked from 'package:forecast':  ##
## accuracy
```

```
actual_wheat_production=c(1095.86,1068.40)
predict_wheat_production=c(1156.443,1198.264)
```

```
result_1=rmsle(actual_wheat_production,predict_wheat_production)
result_1
```

```
## [1] 0.08951492
```

```
result_2=mape(actual_wheat_production,predict_wheat_production)
result_2
```

```
## [1] 0.08841675
```

```
result_3=mase(actual_wheat_production,predict_wheat_production)
result_3
```

```
## [1] 3.467717
```

RESIDUAL ANALYSIS : HOLTS EXPONENTIAL SMOOTHING

- Residulas are uncorrelated
- Residuals are normally distributed

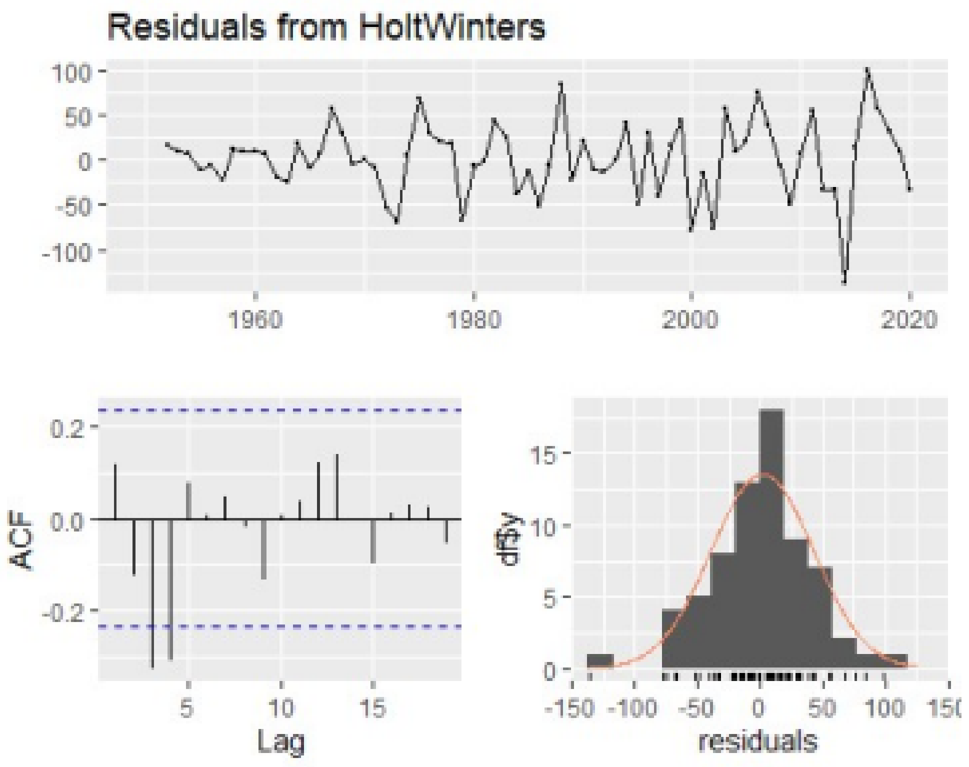
So we have done the Arima forecasting and we obtained the in-sample forecasted values, and we noticed that the RMSE values are high, but all other values are good, and hence we could say that the model is good and let us check for residuals.

RESIDUAL ANALYSIS: ARIMA

```
#res<-resid(fit)
#Checing whether the residuals are uncorrelated #Box.test(res, type = "Ljung-Box")
```

```
forecast::checkresiduals(forecast_data)
```

```
## Warning in modeldf.default(object): Could not find appropriate degrees of ## freedom
for this model.
```



normality test
#install.packages("nortest")

library(nortest)
#ad.test(res)

Here, we have done the Holt exponential smoothing's in the sample forecast and the values for 2019 and 2020 are predicted and compared with the original values and obtained the rmse, mape, and mase. Here we can see that mase and mape is significantly less and rmse is not very high. Hence it is a good model, and when we check the residuals, we obtain that the residual assumptions are followed. Since both the p-values are greater than .05 they are uncorrelated and are normally distributed. Since we are performing a comparative study, let's fit the ARIMA model with the same dataset and observe the result.
==
== ==

ARIMA MODEL
library(forecast)
#Model is fitted fit=auto.arima(wheat_production_1)
fit

```
## Series: wheat_production_1  ## ARIMA(1,1,0) with drift  ##
## Coefficients:  ## ar1 drift  ## -0.3416 14.7873  ## s.e. 0.1114 2.9638  ##
## sigma^2 = 1131: log likelihood = -344.45  ## AIC=694.9 AICc=695.26 BIC=701.65
```

Basically, auto. Arima gives the best-fitted model among all classes of ARIMA models using AIC value. Here, it provides ARIMA(1,1,0) as a best-fitted model for our series. The general representation of Arima is ARIMA(p,d,q) where p is the lag order of AR process, d is integrated term which gives the number of differencing to get the stationary series and q is the lag order of MA process. The suggested model is ARIMA(1,1,0) which indicates it's an AR(1) process with order of integration as 1. Since its the AR() process, there are two parameters. The estimated parameters are -0.3416 with drift 14.7873 for AR(1) .

```
forecast_wheat_production<-forecast(fit,h=2)
forecast_wheat_production
```

```
## Point Forecast Lo 80 Hi 80 Lo 95 Hi 95 ## 2021 1109.803 1066.706 1152.900
1043.891 1175.714 ## 2022 1124.878 1073.278 1176.478 1045.963 1203.794
```

```
library(Metrics)
```

```
actual_wheat_production2=c(1095.86,1068.40)
predict_wheat_production2=c(1156.443,1198.264)
```

```
result_1=rmsle(actual_wheat_production2,predict_wheat_production2)
result_1
```

```
## [1] 0.08951492
```

```
result_2=mape(actual_wheat_production2,predict_wheat_production2)
result_2
```

```
## [1] 0.08841675
```

```
result_3=mase(actual_wheat_production2,predict_wheat_production2)
result_3
## [1] 3.467717
```

So we have done the Arima forecasting and we obtained the in-sample forecasted values, and we noticed that the RMSE values are high, but all other values are good, and hence we could say that the model is good and let us check for residuals

RESIDUAL ANALYSIS: ARIMA

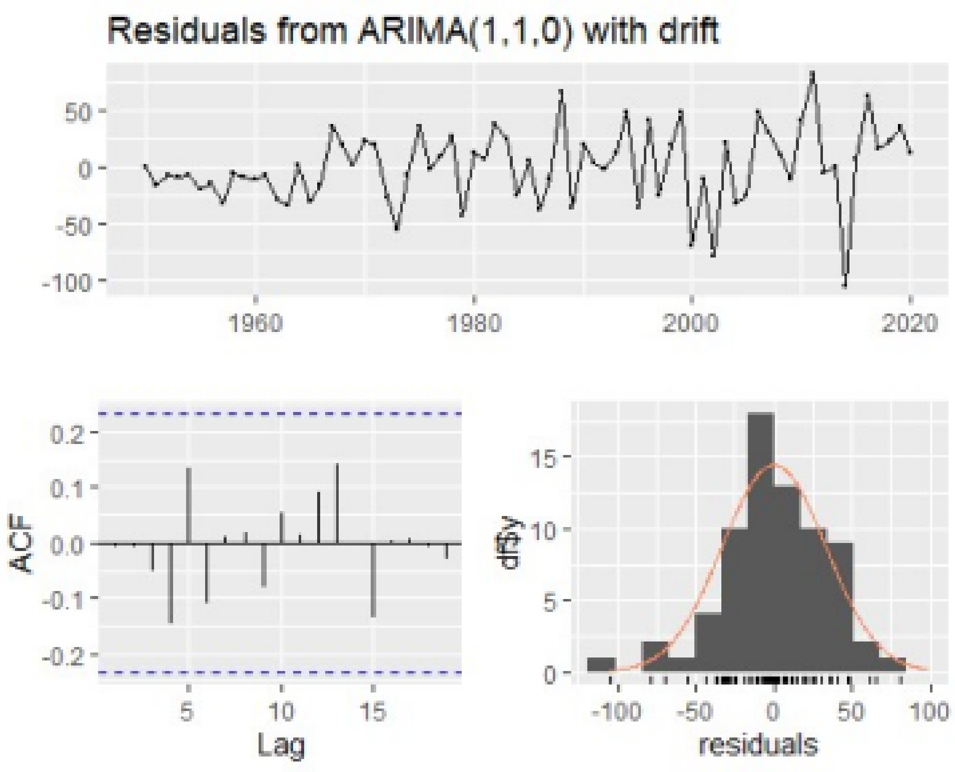
```
res <- resid(fit)
```

Checing whether the residuals are uncorrelated

```
Box.test(res, type = "Ljung-Box")
```

```
##
## Box-Ljung test ##
## data: res ## X-squared = 0.0063778, df = 1, p-value = 0.9363
```

```
forecast::checkresiduals(fit)
```



```
##
## Ljung-Box test ##
## data: Residuals from ARIMA(1,1,0) with drift ## Q* = 5.0536, df = 9, p-value = 0.8296
##
## Model df: 1. Total lags used: 10
```

```
# normality test shapiro.test(res)
```

```
##
## Shapiro-Wilk normality test ##
## data: res ## W = 0.98475, p-value = 0.5451
```

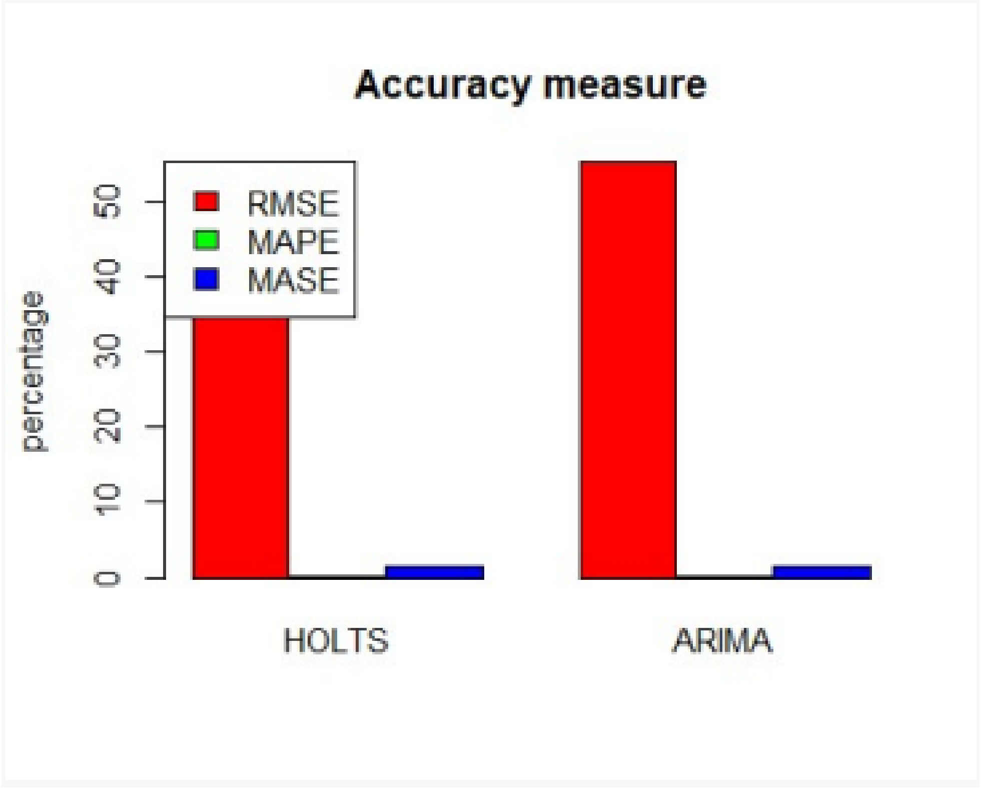
Since all lags in acf and pacf lies inside the threshold line, the correlations in the all lags are negligible. Therefore, the residuals are uncorrelated. In residuals we observe that they are uncorrelated and normally distributed. Hence ARIMA also follows the residual assumptions. Now let's compare the two models to find the best model and lets forecast for the future years

COMPARING THESE TWO MODELS.

```
# Create a data frame col1 <- c(48.04075,0.03903546,1.388059) col2 <- c(
55.22117,0.04489276,1.596279)

data <- data.frame(col1,col2)  names(data) <- c("HOLTS","ARIMA")

# barplot with colors. Make sure that the plot and legends have same colors for items.
barplot(height=as.matrix(data), main="Accuracy measure", ylab="percentage",
beside=TRUE,co l=rainbow(3))
#Add legends legend("topleft", c("RMSE","MAPE","MASE"),fill=rainbow(3))
```



So from the graph, it’s evident that the Holt is a better model with less RMSE, MAPE,MASE values than ARIMA. Let’s forecast the net 5 year value using Holts

FORECASTING USING THE BEST MODEL.

```
#fitted the model library(astsa)
wheat_prod_for1=HoltWinters(wheat_production, beta= TRUE,gamma = FALSE)  ##
prediction for next 5 years library(forecast)
forecast_data=forecast(wheat_prod_for1,h=5)
forecast_data

## Point Forecast Lo 80 Hi 80 Lo 95 Hi 95  ## 2023 -513.484 -675.8879 -351.0801
-761.8593 -265.1087  ## 2024 -1310.483 -1611.4980 -1009.4680 -1770.8457 -850.1203
## 2025 -2107.482 -2592.3932 -1622.5709 -2849.0896 -1365.8745  ## 2026 -2904.481
-3605.9644  -2202.9977  -3977.3072  -1831.6549   ##  2027  -3701.480  -4646.7566
-2756.2036 -5147.1557 -2255.8044
```

==

===
==

Time Series Forecasting with KNN in R

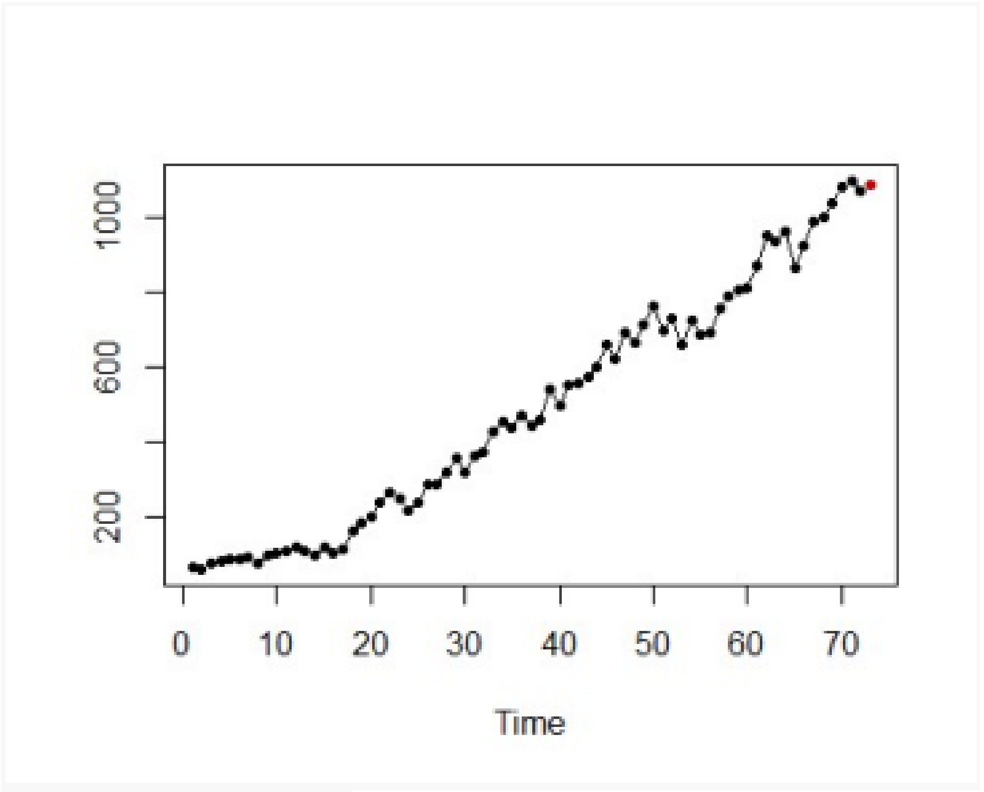
```
#install.packages("tsfknn")
library(tsfknn)
pred <- knn_forecasting(ts(j$WHEAT),h = 1, lags = 1:2, k = 2, transform = "none")
knn_examples(pred)
```

```
## Lag2 Lag1 H1  ## [1,] 64.6 61.80 75.00  ## [2,] 61.8 75.00 80.20  ## [3,] 75.0 80.20 90.40  ## [4,] 80.2 90.40 87.60  ## [5,] 90.4 87.60 94.00  ## [6,] 87.6 94.00 79.90  ## [7,] 94.0 79.90 99.60  ## [8,] 79.9 99.60 103.20  ## [9,] 99.6 103.20 110.00  ## [10,] 103.2 110.00 120.70  ## [11,] 110.0 120.70 107.80  ## [12,] 120.7 107.80 98.50  ## [13,] 107.8 98.50 122.60  ## [14,] 98.5 122.60 104.00  ## [15,] 122.6 104.00 113.90  ## [16,] 104.0 113.90 165.40  ## [17,] 113.9 165.40 186.50  ## [18,] 165.4 186.50 200.90  ## [19,] 186.5 200.90 238.30  ## [20,] 200.9 238.30 264.10  ## [21,] 238.3 264.10 247.40  ## [22,] 264.1 247.40 217.80  ## [23,] 247.4 217.80 241.00  ## [24,] 217.8 241.00 288.40 ## [25,] 241.0 288.40 290.10  ## [26,] 288.4 290.10 317.50  ## [27,] 290.1 317.50 355.10  ## [28,] 317.5 355.10 318.30  ## [29,] 355.1 318.30 363.10  ## [30,] 318.3 363.10 374.50  ## [31,] 363.1 374.50 427.90  ## [32,] 374.5 427.90 454.80  ## [33,] 427.9 454.80 440.70  ## [34,] 454.8 440.70 470.50  ## [35,] 440.7 470.50 443.20  ## [36,] 470.5 443.20 461.70  ## [37,] 443.2 461.70 541.10  ## [38,] 461.7 541.10 498.50 ## [39,] 541.1 498.50 551.40  ## [40,] 498.5 551.40 556.90  ## [41,] 551.4 556.90 572.10  ## [42,] 556.9 572.10 598.40
## [43,] 572.1 598.40 657.70  ## [44,] 598.4 657.70 621.00  ## [45,] 657.7 621.00 693.50  ## [46,] 621.0 693.50 663.50  ## [47,] 693.5 663.50 712.90  ## [48,] 663.5 712.90 763.70  ## [49,] 712.9 763.70 696.80  ## [50,] 763.7 696.80 727.70  ## [51,] 696.8 727.70 657.60  ## [52,] 727.7 657.60 721.60  ## [53,] 657.6 721.60 686.40  ## [54,] 721.6 686.40 693.50  ## [55,] 686.4 693.50 758.10  ## [56,] 693.5 758.10 785.70 ## [57,] 758.1 785.70 806.80  ## [58,] 785.7 806.80 808.00  ## [59,] 806.8 808.00 868.70  ## [60,] 808.0 868.70 948.80  ## [61,] 868.7 948.80 935.10  ## [62,] 948.8 935.10 958.50  ## [63,] 935.1 958.50 865.30  ## [64,] 958.5 865.30 922.90  ## [65,] 865.3 922.90 985.10  ## [66,] 922.9 985.10 998.70  ## [67,] 985.1 998.70 1036.00  ## [68,] 998.7 1036.00 1078.60  ## [69,] 1036.0 1078.60 1095.86  ## [70,] 1078.6 1095.86 1068.40
```

```
pred$prediction
```

```
## Time Series:  ## Start = 73  ## End = 73  ## Frequency = 1  ## [1] 1082.13
```

```
plot(pred)
```

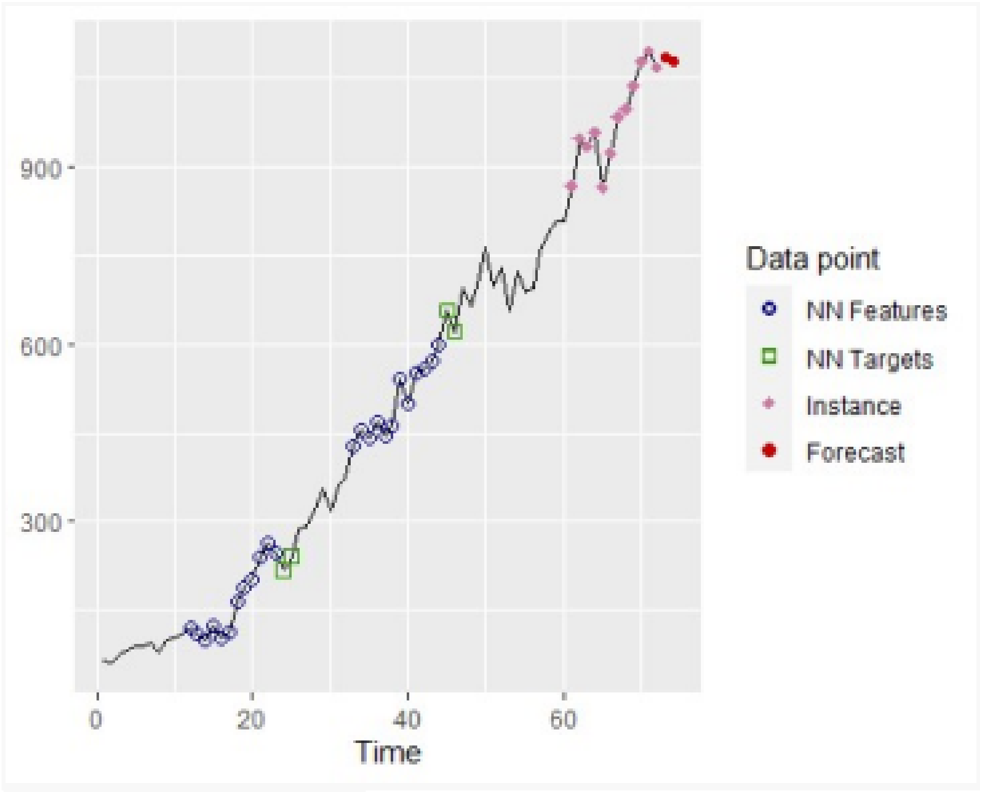


```
nearest_neighbors(pred)
```

```
## [[1]]  ## [[1]]$instance  ## Lag 2 Lag 1  ## 1095.86 1068.40  ##
## [[1]]$nneighbors  ## Lag 2 Lag 1 H1  ## 1 1078.6 1095.86 1068.40  ## 2 1036.0
1078.60 1095.86
```

```
#library(ggplot2)
#autoplot(pred, highlight = "neighbors")
```

```
pred <- knn_forecasting(j$WHEAT, h = 2, lags = 1:12, k = 2, msas = "MIMO")
autoplot(pred, highlight = "neighbors", faceting = FALSE)
```

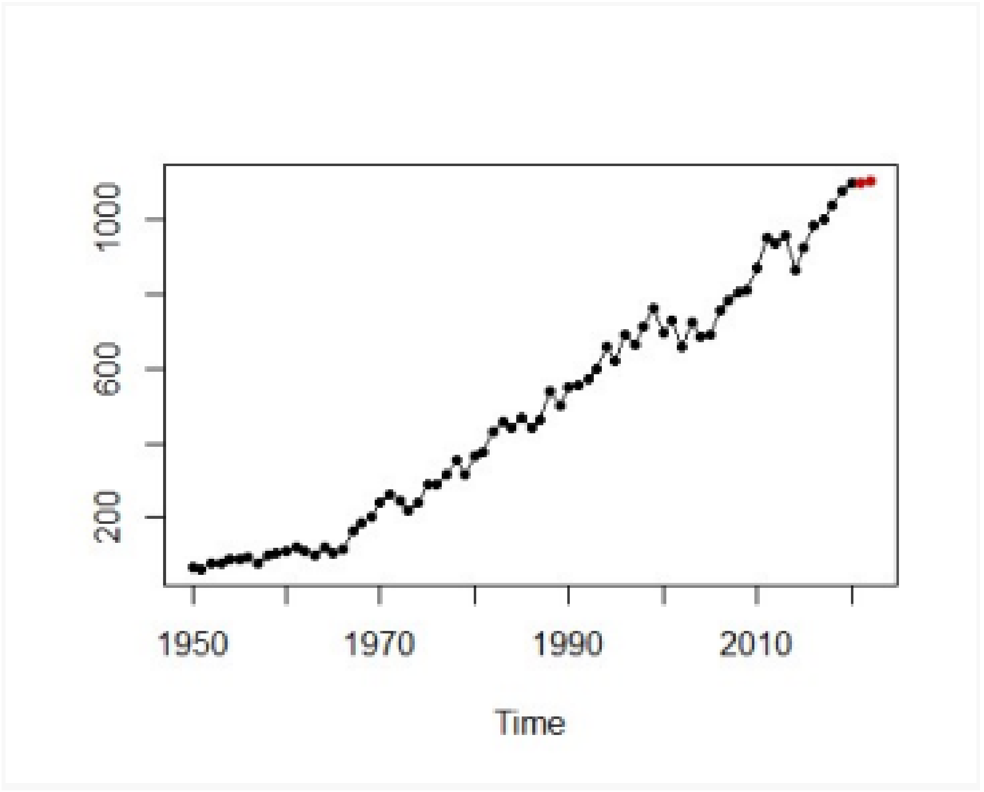


```
#install.packages("class")
pred <- knn_forecasting(wheat_production_1, h =2)  pred$prediction
```

```
## Time Series:  ## Start = 2021  ## End = 2022  ## Frequency = 1  ## [1] 1099.855
1103.850
```

```
r=pred$model$examples
```

```
plot(pred)
```



## Forecasting time series with a trend

KNN is not suitable for forecasting a time series with a trend. The reason is simple, KNN predicts an average of historical values of the time series, so it cannot predict correctly values out of the range of the time series. In your time series has a trend we recommend using the parameter transform to transform the training samples. Use the value "additive"if the trend is additive or "multiplicative" for exponential time series

```
actual_wheat_production=c(1095.86,1068.40)
predict_wheat_production=c(1099.855,1103.850)

result_3=rmsle(actual_wheat_production,predict_wheat_production)
result_3
```

```
## [1] 0.0232029
```

```
actual_wheat_production=c(1095.86,1068.40)
predict_wheat_production=c(1099.855,1103.850)

result_3=mape(actual_wheat_production,predict_wheat_production)
result_3
```

```
## [1] 0.018413
```

```
actual_wheat_production=c(1095.86,1068.40)
predict_wheat_production=c(1099.855,1103.850)
```

```
result_3=mase(actual_wheat_production,predict_wheat_production)
result_3
```

```
## [1] 0.7182265
```

==
== ==

#### PEDICTION FOR THE NEXT FIVE YEARS

```
pred <- knn_forecasting(wheat_production_1, h =5) pred$prediction
```

```
## Time Series: ## Start = 2021 ## End = 2025 ## Frequency = 1 ## [1] 1099.855
1103.850 1107.846 1111.841 1115.836
```

THE VALUES FOR THE NEXT FIVE YEAR ARE :

1099.855,

1103.850,

1107.846,

1111.841,

1115.836

CONCLUSION

Here I tried to forecast the wheat production in India for the next five years. Here I use three Models to predict the time series data: Holt Exponential, ARIMA, and KNN. For Holt exponential, I got the RMSLE of the model is almost 0.8, and for the ARIMA model, it's 0.08, and for the KNN, I got 0.02. So from my study, I learned that RMSLE with low value has better prediction ability, so here, my KNN model has low RMSLE, which is the best model I used in my time series prediction.