PREDICTION OF INCIDENCE AND SEVERITY OF COFFEE LEAF RUST

A Project Report

Submitted in partial fulfilment of requirements for the degree of

Master of Management

By

Amitabh Gunjan

Under the guidance of

Prof. Chiranjit Mukhopadhyay

Department of Management Studies

Indian Institute of Science

Bangalore – 560012

June 2018

Acknowledgements

I would like to extend my deepest gratitude to Prof. Chiranjit Mukhopadhyay for his guidance and support throughout this project. It is an honour to work under him.

I take immense pleasure in thanking my external examiner Prof. P. Balachandra for his views on my work. His suggestions were very helpful in shaping the project.

Most of all, I would like to my thank my parents for their blessings that saw me through all the difficult times.

Amitabh Gunjan

IISc, Bangalore

26/06/2018

Abstract

Coffee leaf rust is a disease caused by the fungus, Hemileia Vastatrix. It leads to premature fall of leaves and leads to huge losses of yield. In recent years there have been outbreaks around the world especially Latin America. There is a dearth of studies focusing on prediction of coffee leaf rust for plantations in India. The same has been undertaken in this project.

In this project, coffee leaf rust disease variables and weather variables have been modelled using vector Autoregression (VAR) and forecasting of coffee leaf rust has been done. The motivation behind using VAR comes from the field of econometrics where it has been used widely for modelling multiple time series variables and to understand the relationship among the variables. The same has been investigated in this study, through structural analysis.

Later in the project, recurrent neural network and one of its most popular variant, LSTM has been used for modelling the rust incidence and severity. The forecast from VAR and LSTM has been compared. Finally, a model for forecasting leaf rust incidence has been proposed in the conclusion.

Table of Contents

List of Tables

List of figures

Chapter One

Introduction

Background

Coffee production in India is dominated in the hill tracts of South Indian states, with Karnataka accounting for 71%, followed by Kerala with 21% and Tamil Nadu (5% of overall production with 8,200 tonnes). Indian coffee is said to be the finest coffee grown in the shade rather than direct sunlight anywhere in the world. There are about 250,000 coffee growers in the country; 98% of them are small growers. As of 2009, Indian coffee made up just 4.5% of the global production. Almost 80% of Indian coffee is exported [1].

Coffee leaf rust (CLR) spread to India in 1870s and caused large scale damage to Arabica crops [2]. Coffee Leaf Rust (CLR) has many direct and indirect impacts on coffee production. Direct impacts include decreased quantity and quality of yield produced by the diseased plant. Indirect impacts include increased costs to combat and control the disease. Fungicide application and stumping diseased plants and replacing them with resistant breeds are some of the methods employed for combatting the disease. Both methods include huge labour costs and material costs [6].

Coffee rust epidemics, with intensities higher than previously observed, have affected several countries in recent years including: Colombia, from 2008 to 2011; Central America and Mexico, in 2012–13; and Peru and Ecuador in 2013 [4]. There has been an increasing number of research papers focused on prediction of coffee leaf rust incidence and severity using machine learning methods, presumably because of coffee leaf rust epidemics becoming more common and advances in machine learning and deep learning. The accuracy of models predicting disease has not been up to the mark [9].

Many of the literature on prediction of coffee leaf rust incidence and the severity of the disease suggest that machine learning techniques have been used extensively for the same [5]. Prediction of the disease incidence in advance can be very helpful in alerting farmers and can ultimately decrease the losses of yield and reduce cost of combatting the disease. There has been an increasing number of research papers focused on prediction of coffee leaf rust incidence and severity using machine learning methods. [3]

Problem statement

An experimental trial was initiated in one of the CCRI farms in Karnataka, India, to record the coffee leaf rust incidence on S.795, Sln.5B and CXR varieties for developing forewarning model on the outbreak of diseases. Observations on leaf rust disease were recorded on fortnightly basis from S.795 Sln.5B and CXR. In all three varieties, twenty-five plants have been marked and observations were recorded on total number of plant population infected, incidence of leaf rust and its severity. A forewarning model of leaf rust incidence and severity must be developed for the plantation so that the owners can take appropriate measures to prevent the outbreak of the disease.

Literature review

There have been many research papers focusing on predicting coffee leaf rust incidence and severity using machine learning algorithms. David Camillo Corrales et. al. in Two-Level Classifier Ensembles for Coffee Rust Estimation in Colombian Crops, proposed classifier ensembles of Neural networks, regression tree and support vector regression for coffee rust estimation. The dataset used was composed of three categories, namely, weather conditions (six attributes), physic crop properties (three attributes) and crop management (four attributes). Perez Ariza et. al. [3] presented an agricultural case study for prediction of coffee leaf rust using Bayesian networks. The dataset comprised of monthly accounts of coffee leaf rust incidence. In the paper Using nondeterministic learners to alert on coffee rust disease **-** Oscar Luaces et. al. discretized the continuous disease variable and predicted if the value will be greater than a given threshold. The dataset comprised of monthly accounts of coffee leaf rust incidence in an experimental farm in Brazil.

David Corrales presented an overview of all the machine learning algorithms being used for prediction of coffee leaf rust in the paper Towards Detecting Crop Diseases and Pest by Supervised Learning. He found that Decision trees, Bayesian Networks, artificial neural networks, support vector machines, K-nearest neighbour have been used by researchers for predicting coffee leaf rust.

In this work I have used Vector Autoregression (VAR) which is one of the most commonly used methods in time series forecasting for multivariate time series, especially in econometrics. Later in the project I have used long short-term memory (LSTM), a type of RNN. LSTM has been very successful on different sequence learning tasks such as natural language translation, speech recognition, time series forecasting. Both techniques have not been used for prediction of coffee leaf rust in previous works.

Flow of project report

I have described the data collection methodology and a brief discussion of the data set has been done in the later parts of chapter 1. In this work, Vector Autoregressive (VAR) models have been used to model incidence (percentage of coffee leaves infected out of the total observed) and severity (the average intensity of the rust) of coffee leaf rust along with five weather variables. Vector autoregressive models (VAR) are used to model multiple time series and the two main goals of VAR are forecasting and structural analysis [7]. The same has been discussed in chapter 2. It has been shown in previous research that there is a link between the weather conditions and the intensity of disease [8]. Structural analysis has been done to discover Granger causality among the weather and disease variables, instantaneous causality. Impulse response analysis has been done to explore the affects of giving a unit shock in one of the weather variables induces the disease variables to change and how.

In chapter 3 recurrent neural networks have been introduced. Chapter 4 deals with the comparison of the accuracy of VAR and LSTM models. Finally, conclusion and references are attached at the last.

Data collection, description

Data collection

Data was collected on an experimental farm having the below illustrated setup:

An experimental trial was initiated in the CCRI farm to record the leaf rust incidence on S.795, Sln.5B and CXR varieties for developing forewarning model on the outbreak of diseases. Observations on leaf rust disease were recorded at fortnightly interval from S.795 Sln.5B and CXR for a span of eight years.

In both Arabica and Robusta cultivars twenty-five plants have been marked and observations were recorded on total number of plant population infected, incidence and severity of the leaf rust.

Procedure followed for recording the observations: All the standard recommended practices applied to the experimental blocks. For leaf rust studies, use of systemic fungicides in Arabica plots was avoided. In each of the experimental plots, five spots were marked in the below manner.

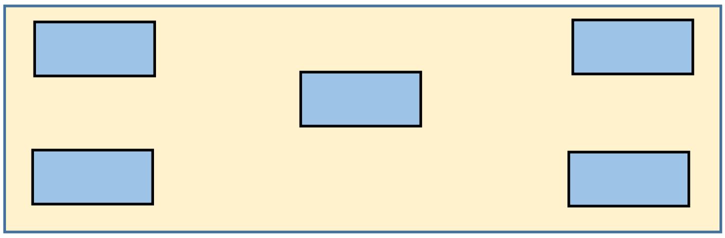


Fig- 2.1 – A plot showing the experimental farm

In each spot five normal plants were marked randomly, for recording of various observations by Agronomists, Entomologists and Plant Pathologists.

Data description

Two categories of data were collected for the plantation:

a) Weather variables
b) Disease variables

Five weather variables were collected monthly and they are:

a) Average maximum temperature (monthly)
b) Average minimum temperature(monthly)
c) Average Relative humidity(monthly)
d) Average Rainfall
e) Average Sunshine

Three measures for leaf rust were collected fortnightly:

a) Population infected
b) Rust incidence
c) Severity

a) Population infected: (Number of plants infected/Total number of plants observed) *100

b) Rust incidence: (Number of infected leaves/Total number of leaves observed) *100

c) Severity: (Sum of severity ratings of all observed leaves/ (5* Total number of leaves observed))*100

Population infected measures the percentage of plants that have been infected out of total observed. Rust incidence measures the percentage of leaves infected out of the total number of leaves observed. The two measures are only giving information about how much of the population is infected by the rust incidence. Severity measures the intensity of the rust. Every leaf is given a rating based on the infection and then the ratings are added up and then divided by five times the number of leaves observed. The whole quantity is multiplied by 100 so that the range of the severity is between 0 to 100.

Chapter Two

VAR

For this project I have a total of eight time series variables. Five of them are weather variables and three are disease variables. When there are a lot of variables then it is also often of interest to learn about the dynamic interrelationships between several variables. VAR is used to model multivariate time series and provides great interpretation tools that improve the understanding of the dynamic interrelationships among the time series variables. In particular, Granger causality leads to infer about the causal relationships among variables.

For instance, in a system consisting of three disease variables and five weather variables, one may want to know about the likely impact of a change in one of the weather variables, let's say average minimum temperature, on one of the disease variables, for instance rust incidence. The interesting questions that demand an answer could be: What will be the present and future implications of such an event for rust incidence? Under what conditions can the effect of an increase in average minimum temperature be isolated and traced through the system? These questions can be answered through structural analysis using VAR. But equally important objective of VAR is forecasting. So basically, the two main goals of using VAR are forecasting and structural analysis and I have focused on both throughout the project.

VAR Model

VAR models are the state-of-the-art when it comes to modelling multiple time series variables. A VAR model describes the evolution of a set of *k* variables (called *endogenous variables*) over time (sample period) as a linear function of only their past values. The variables are modelled as a linear combination of their own lagged values and of lagged values all other variables in the system.

For illustration a VAR(p) model:

VAR(p) model:

$$y_t = \nu + A_1 y_{t-1} + \cdots + A_p y_{t-p} + u_t$$

where, $y_t = (y_{1t} \ldots y_{kt})'$ is a $K * 1$ random vector, the A_i are fixed $K * K$ coefficient matrices, $\nu = (\nu_1, \ldots \nu_k)'$ is a fixed $K * 1$ vector of intercept terms allowing for the possibility of non-zero mean term, $E(y_t)$. $u = (u_{1t} \ldots u_{kt})'$ is a K-dimensional white noise or innovation process, that is, $E(u_t) = 0$, $E(u_t u_t')$ = $\sum_u$ and $E(u_t u_s') = 0$ for $s \neq t$. The covariance matrix is assumed to be non-singular.

One of the parameters of VAR model is the number of lags, which determines the number of past lags that will be used to model variables as linear combination of past lags of itself and all other variables. The order of VAR model is identified based on various criteria such as AIC, SC, HQ and FPE.

It is very difficult to interpret the coefficients of the variables. Another interpretation tool for VAR is the covariance matrix of the residuals. For a VAR system with eight variables the covariance matrix will be 8*8 or will have 64 terms out of which there will be (64 – 8)/2 = 28 covariance terms. If we want to only look at the covariances then also we will have to interpret 28 covariance terms which is a difficult task.

'Structural VAR' is a depiction of the underlying "structural" relationships between the variables under consideration. For correct interpretation from the structural relationship the residuals must be made uncorrelated. This is achieved by making the components of the autocovariance matrix of the residuals orthogonal. This makes the shocks in different variables independent. If the shocks are not independent one may argue that the error terms consist of all the influences and variables that are not included in the set of endogenous variables.

Modelling procedure

VAR order

To select the VAR order for the system I used Scwartz criteria. It gave the VAR order to be 1.

Model, plots and interpretation

Weather variables:

The dataset comprises of average monthly accounts of maximum temperature, average minimum temperature, Sunshine, rainfall and relative humidity. Let's have a look at the time series plots of the weather variables.

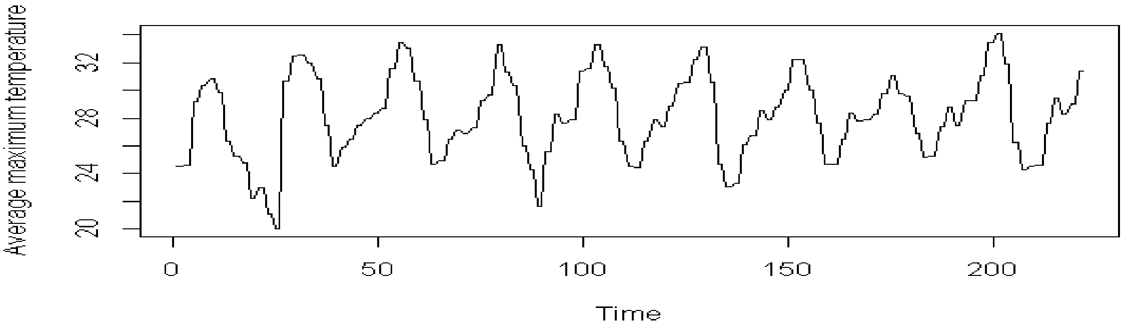


Fig- 3.1 – A time series plot showing the average max temperature

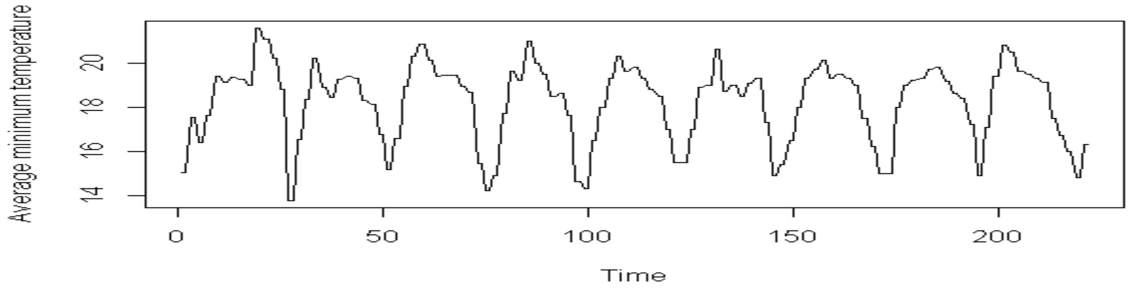


Fig- 3.2 – A time series plot showing the average min temperature

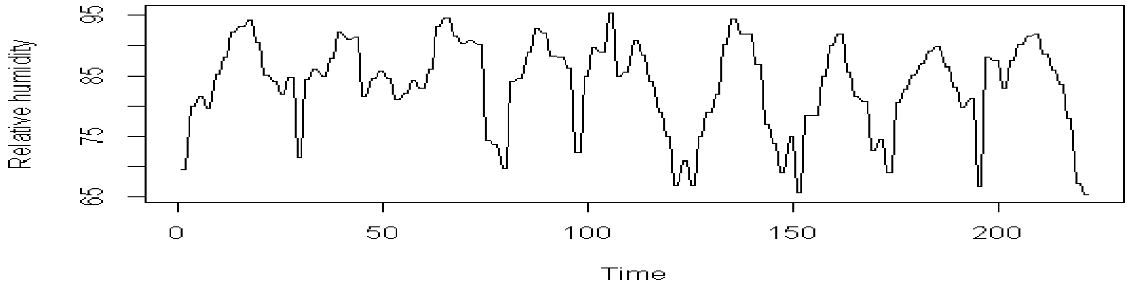


Fig- 3.3 – A time series plot showing the relative humidity

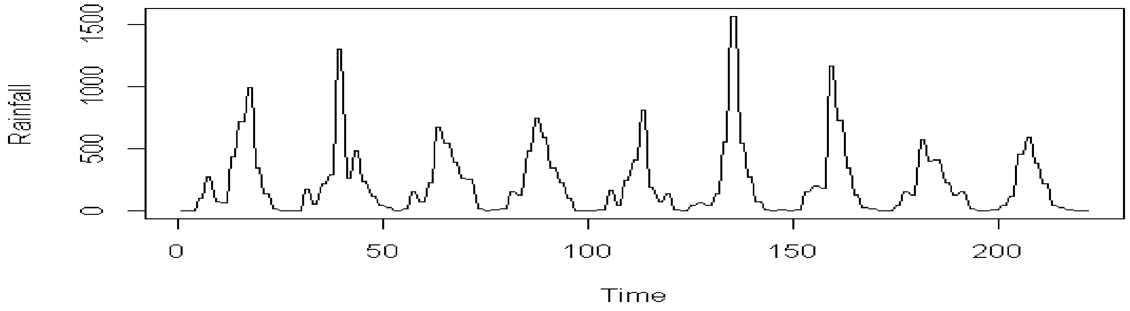


Fig- 3.4 – A time series plot showing rainfall

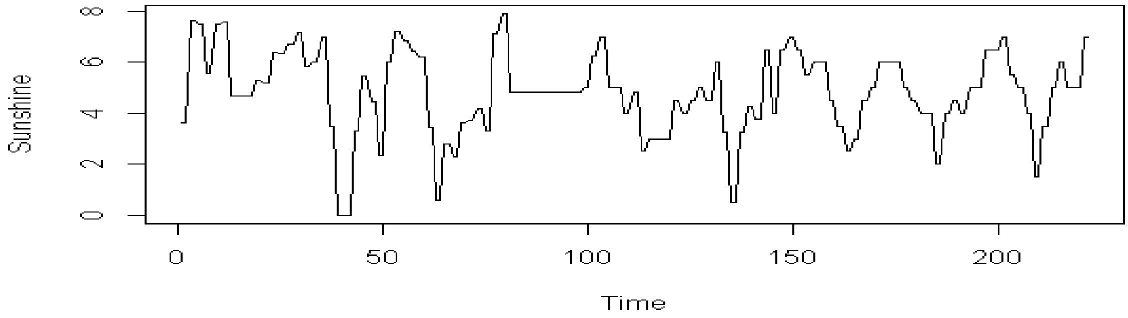


Fig- 3.5 – A time series plot showing sunshine

Coffee varieties:

A total of three varieties were modelled. They are:

a) S795
b) Sln5B
c) CXR

For each variety three disease variables were recorded fortnightly over a span of eight years on the experimental farm. Two of the three disease variables are the main variables of interest and they are:

a) Rust incidence: This depicts the percentage of leaves infected out of the total number of leaves observed.
b) Severity: This measure depicts the average intensity of the infection over the observed samples.

Variety S795

Rust incidence:

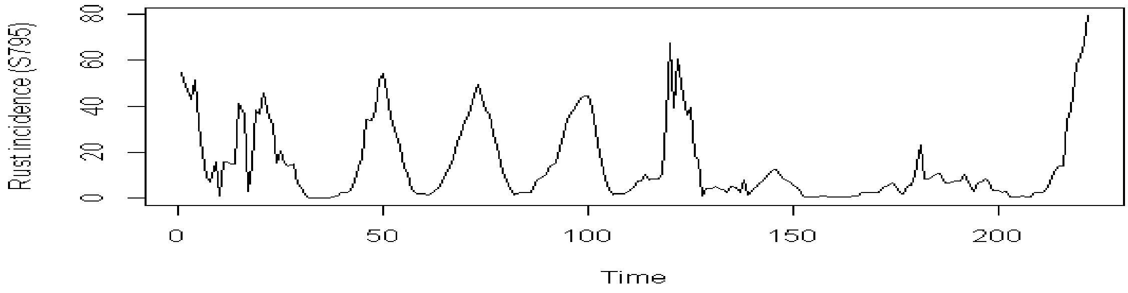


Fig- 3.6 – A time series plot of Rust incidence of S795

The optimal order of the model found to be was 1 using AIC, Scwartz criteria. The final model with the significant terms are:

```
Estimation results for equation RustIncidence:
==============================================
RustIncidence = Sunshine.l1 + RustIncidence.l1 + const

                 Estimate Std. Error t value Pr(>|t|)
Sunshine.l1      -0.88410    0.30628  -2.887  0.00433 **
RustIncidence.l1  0.86729    0.03272  26.503  < 2e-16 ***
const             5.86856    1.61280   3.639  0.00035 ***
```

The only significant terms are sunshine at lag 1 and rust incidence at lag 1.

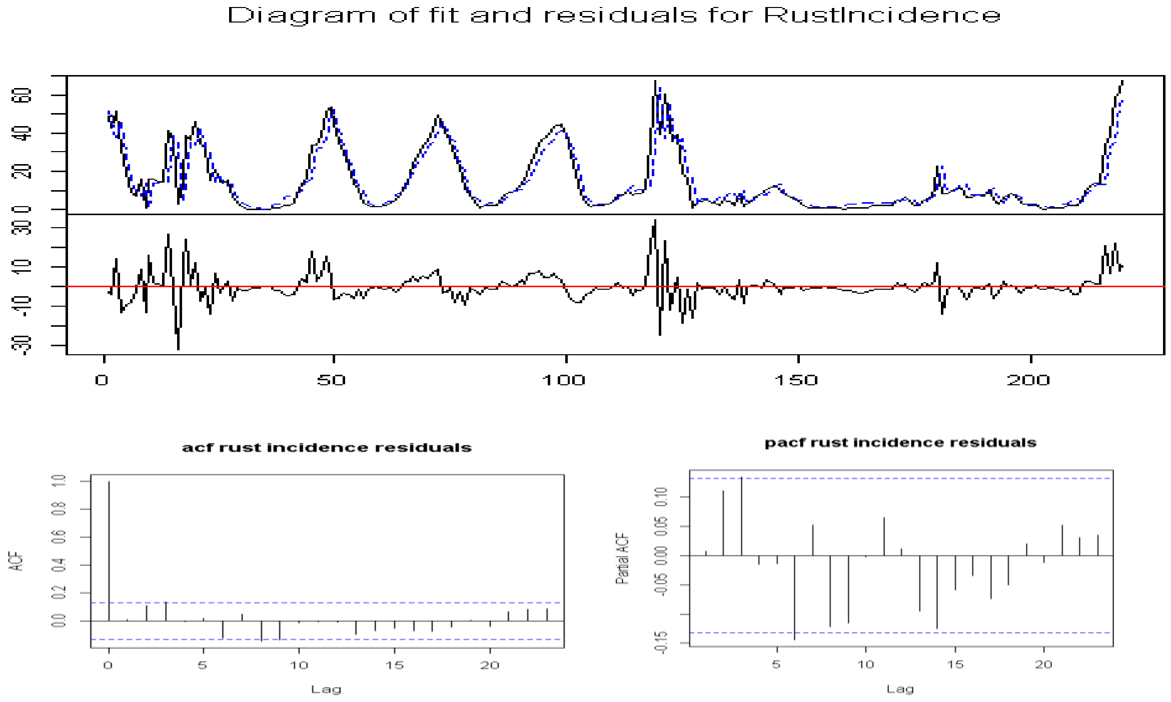


Fig- 3.7 – plot of fit vs residuals and ACF PACF plots of residuals Rust incidence of S795

The above plot shows the plot of fit versus actuals and the residuals. ACF and PACF of residuals has been plotted to show if they are stationary. The ACF and PACF plots of the residuals for rust incidence look fine. This is a sign of stationary residuals.

Severity:

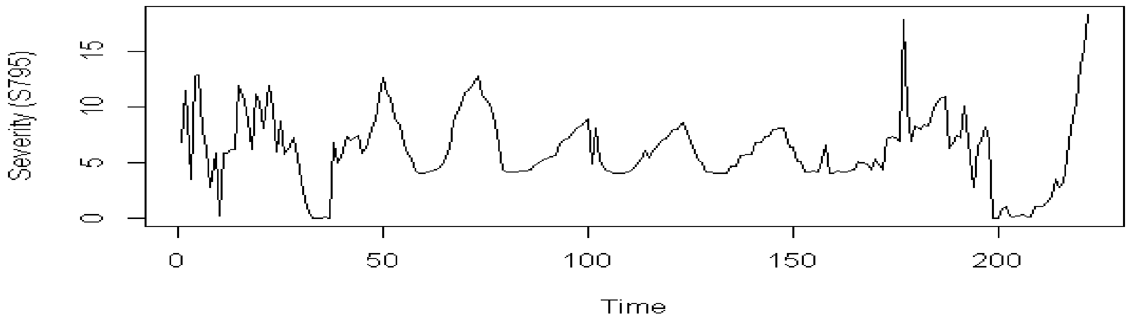


Fig- 3.8 – Time series plot of severity of S795

```
Estimation results for equation Severity:
Severity = TempMax.l1 + RustIncidence.l1 + Severity.l1 + const

                  Estimate Std. Error t value Pr(>|t|)
TempMax.l1        -0.16457    0.04599  -3.579 0.000436 ***
RustIncidence.l1   0.04419    0.01115   3.965 0.000103 ***
Severity.l1        0.51233    0.06355   8.062  7.4e-14 ***
const              7.09550    1.41527   5.014  1.2e-06 ***
```

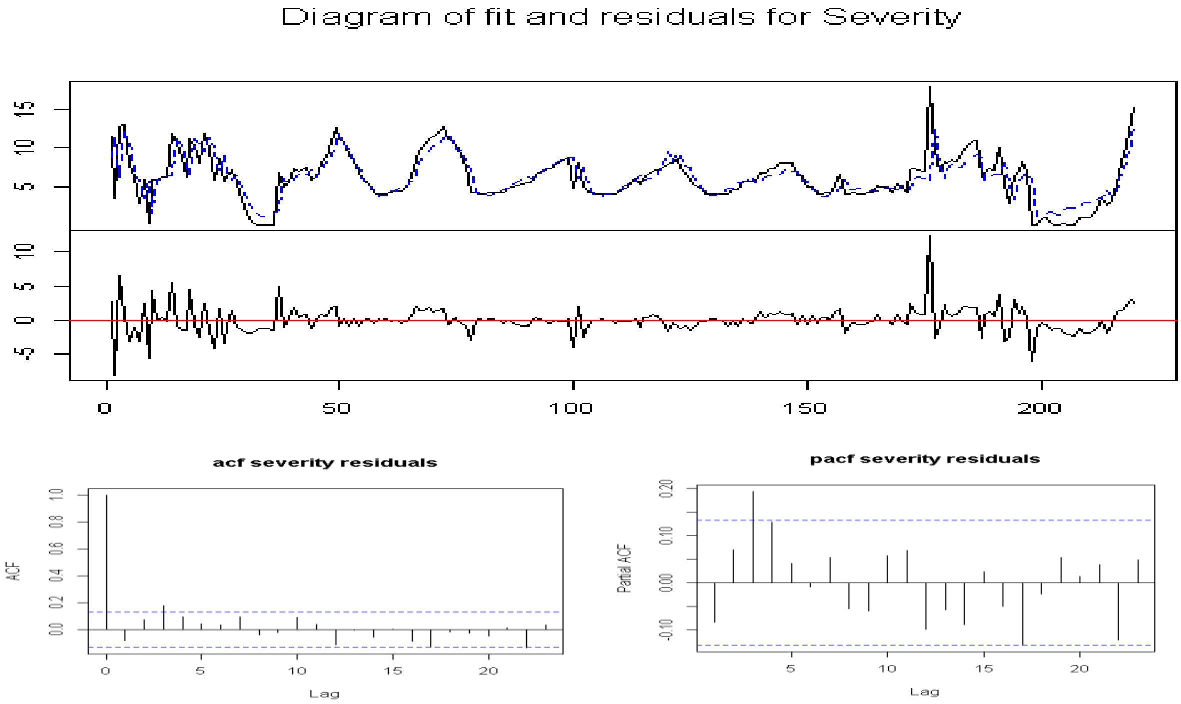


Fig- 3.9 – plot of fit vs residuals and ACF PACF plots of residuals Severity of S795

The above plot shows the plot of fit versus actuals and the residuals. ACF and PACF of residuals has been plotted to show if they are stationary. The ACF and PACF plots of the residuals for severity look fine. This is a sign of stationary residuals.

The multivariate JB test was carried out to check for normality of the residuals. The residuals were found to be non-normal with p value of 2.2e-16.

The portmanteau test was carried out to check for the residuals being white noise. The residuals were indeed found to be white noise with p value of 0.1702

Granger Causality

The set of weather variables containing average maximum temperature, average minimum temperature, relative humidity, sunshine and rainfall were found to be Granger causing rust incidence and severity. Th null hypothesis of the set of weather variables not Granger causing the disease variables can be rejected with p-value = 0.003674.

Impulse response analysis

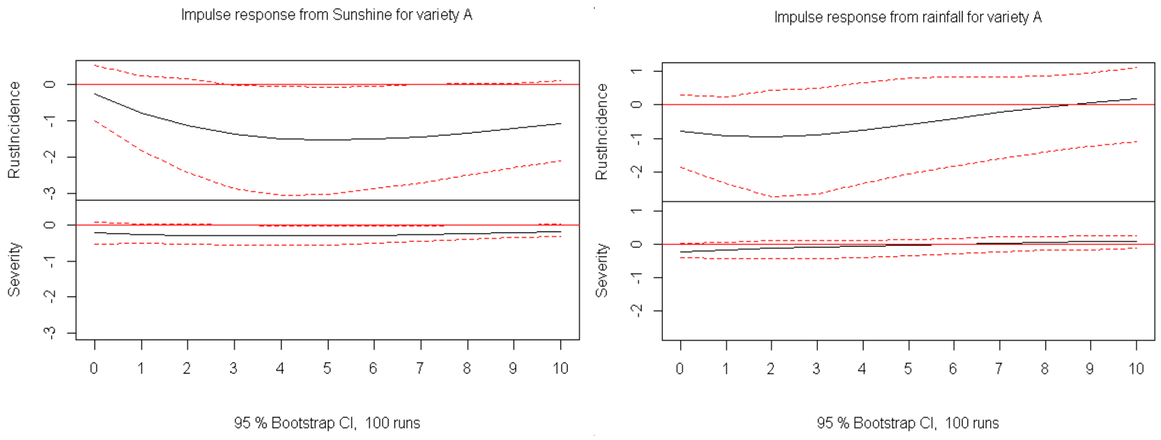


Fig- 3.10 – plots of impulse response from sunshine and rainfall for variety S795 on rust and severity

The IRF plots from the above figure show that a unit shock in sunshine at let's say time t = 0, has a significant impact on rust incidence and induces rust incidence to decrease in the coming periods. Rainfall also has similar effects on rust incidence. There is an instantaneous change in rust incidence at t = 0, in case of unit shock in rainfall which suggests instantaneous causality between rust incidence due to rainfall

Severity does not change much in the coming periods after a unit shock in sunshine or rainfall.

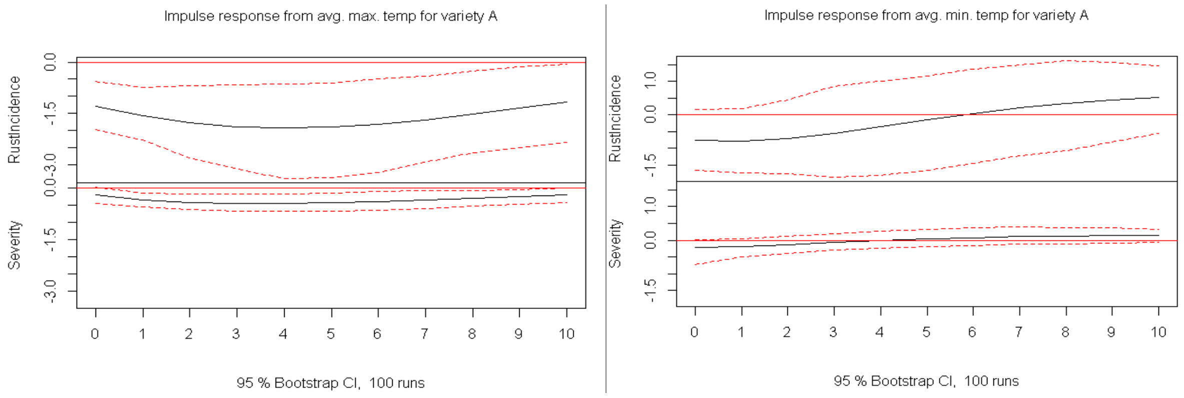


Fig- 3.11 – plots of impulse response from average max and min temperature for variety S795 on rust and severity

A unit shock at let's say t = 0 in average maximum temperature induces rust incidence to decrease in the coming periods. Similarly, for unit shock in average minimum temperature rust incidence decreases in the coming periods and starts increasing in periods after 5 lags or fortnights. There is an instantaneous change in rust incidence at t = 0, which suggests instantaneous causality among rust incidence and average maximum and average minimum temperature.

There is no significant impact on severity from a unit shock in either average maximum temperature or average minimum temperature.

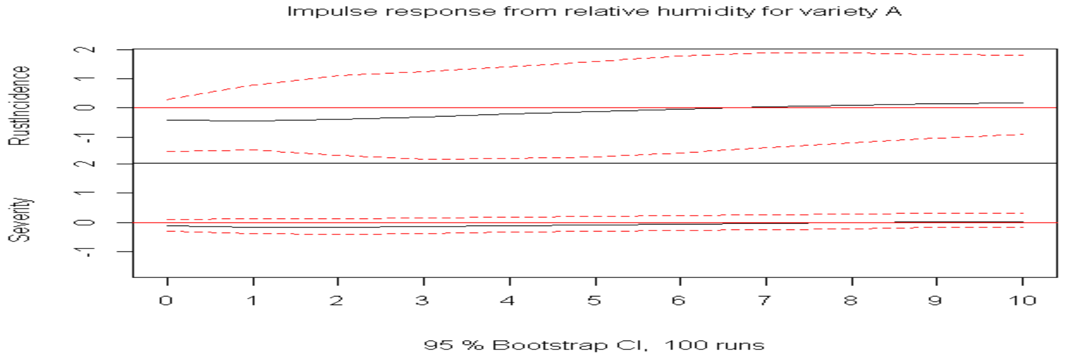


Fig- 3.12 – plots of impulse response from relative humidity for variety S795 on rust and severity

A unit shock in relative humidity at t =0 does not affect rust incidence or severity in the coming periods. Similarly, severity is not much affected by a unit shock in relative humidity.

We can conclude from the impulse response plots:

a) Severity does not get affected by unit shocks in weather variables.
b) Except relative humidity shocks in all other weather variables affect the rust incidence significantly.

c) There is instantaneous causality between rust incidence and the weather variables as we can see rust incidence changes to a unit shock right from the period of shock.

Prediction plot

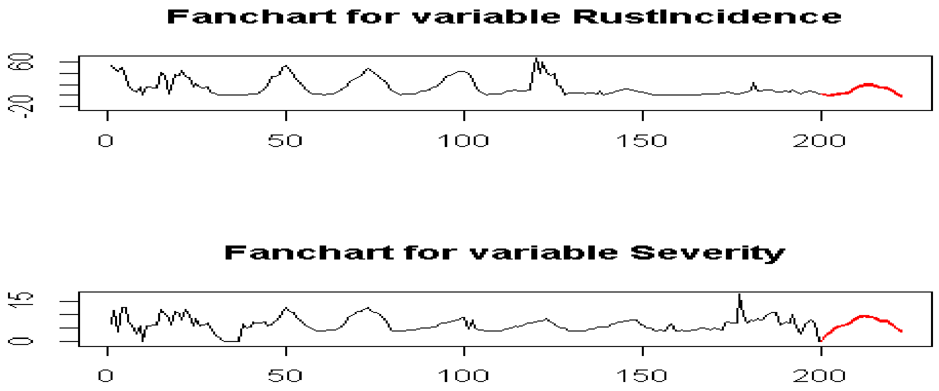


Fig- 3.13 – Fanchart for rust incidence and severity for variety S795

The fanchart for rust incidence and severity show that rust incidence and severity will increase for a few periods and then decrease in the future.

Variety Sln5B

Rust incidence:

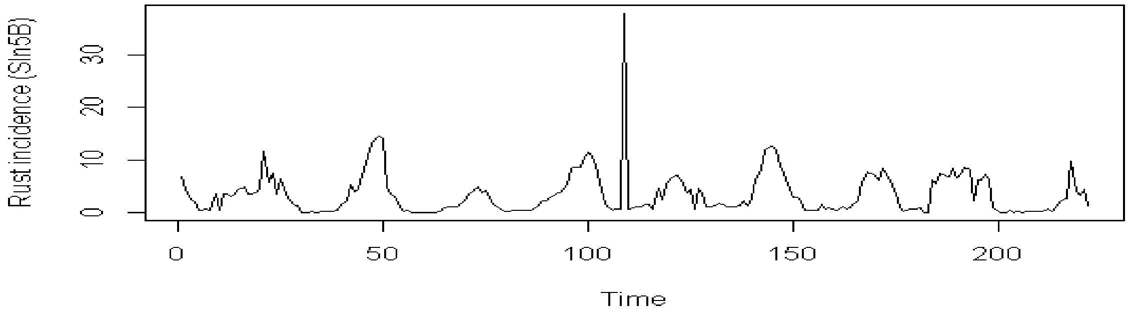


Fig- 3.14 – A time series plot of rust incidence for variety Sln5B

```
Estimation results for equation RustIncidence:
==============================================
RustIncidence = Sunshine.l1 + RainFall.l1 + RelativeHumidity.l1 + Populati
onInfected.l1 + RustIncidence.l1

                       Estimate Std. Error t value Pr(>|t|)
Sunshine.l1           -0.401346   0.175555  -2.286 0.023325 *
RainFall.l1           -0.002579   0.001211  -2.130 0.034410 *
RelativeHumidity.l1    0.039319   0.013692   2.872 0.004538 **
PopulationInfected.l1  0.042969   0.012609   3.408 0.000797 ***
RustIncidence.l1       0.347000   0.071913   4.825 2.82e-06 ***
```

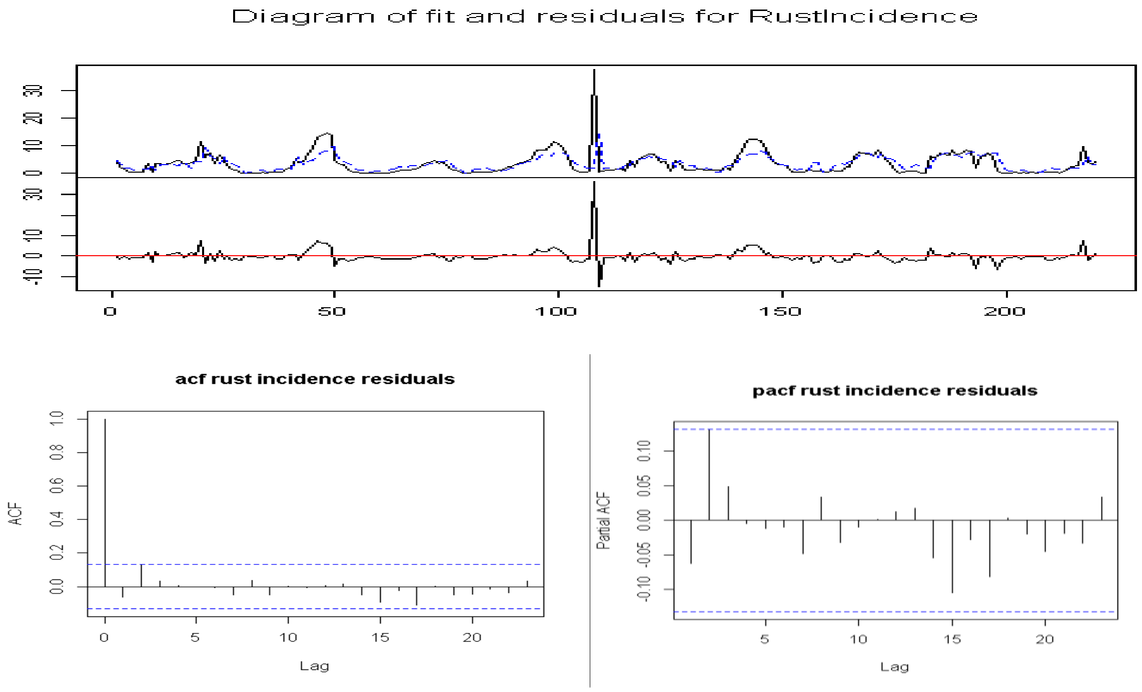


Fig- 3.15 – plot of fit vs residuals and ACF, PACF plots of residuals for rust incidence for variety Sln5

The above plot shows the plot of fit versus actuals and the residuals. ACF and PACF of residuals has been plotted and they look fine. There is no significant autocorrelation or partial autocorrelation which suggests that the residuals are stationary.

Severity:

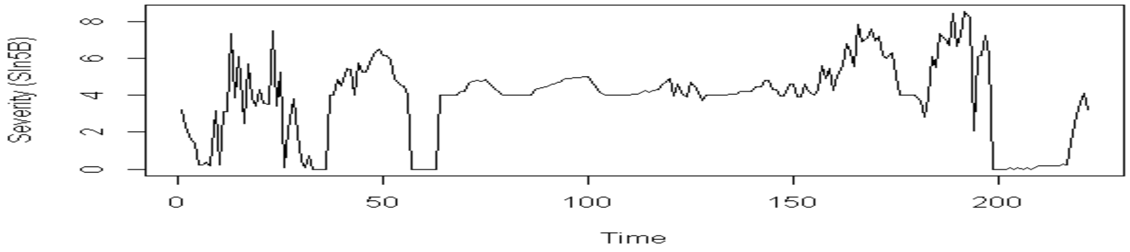


Fig- 3.16 – A time series plot of severity of variety Sln5B

```
Estimation results for equation Severity:
=========================================
Severity = Sunshine.l1 + TempMin.l1 + PopulationInfected.l1 + Severity.l1

                       Estimate Std. Error t value Pr(>|t|)
Sunshine.l1           -0.184664   0.051639  -3.576 0.000440 ***
TempMin.l1             0.106565   0.018094   5.890 1.67e-08 ***
PopulationInfected.l1  0.017341   0.005017   3.456 0.000672 ***
Severity.l1            0.592919   0.062730   9.452  < 2e-16 ***
```

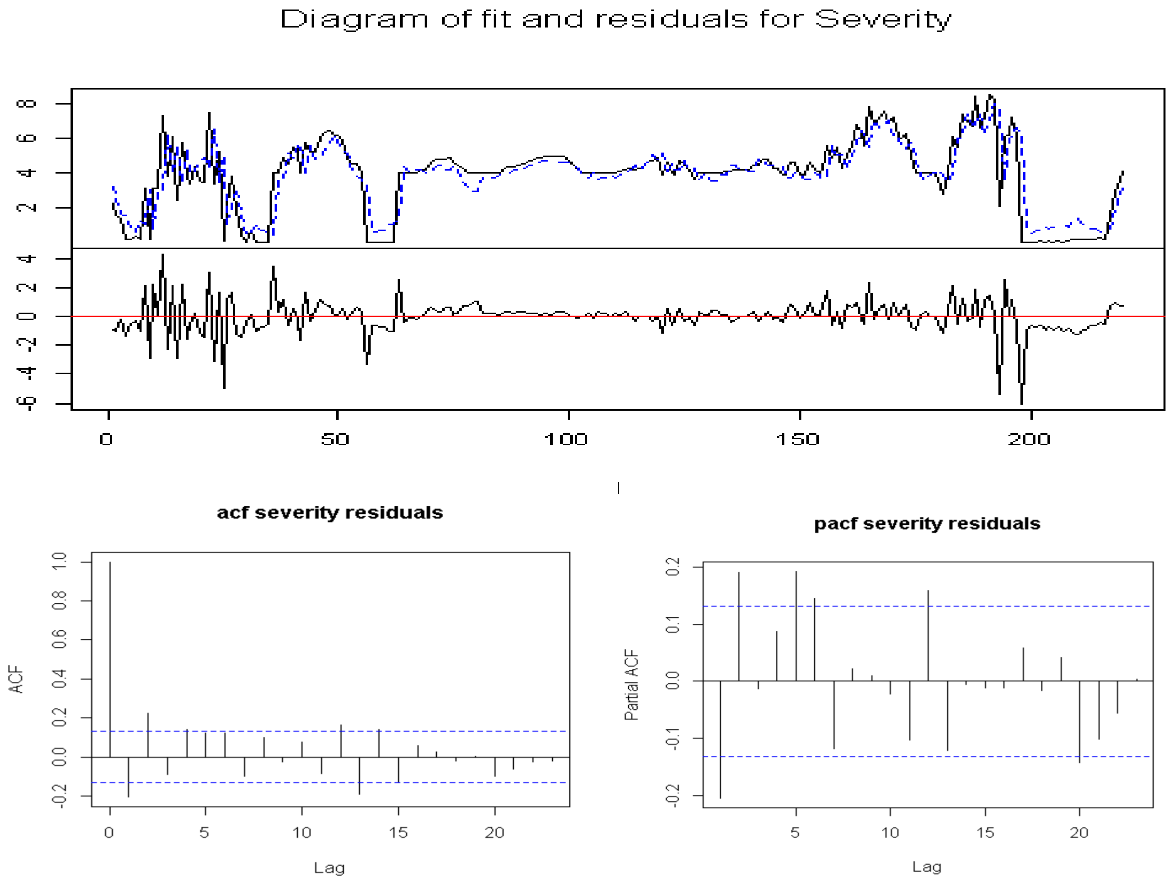


Fig- 3.17 – plot of fit vs residuals and ACF, PACF plots of residuals for severity for variety Sln5

The above plot shows the plot of fit versus actuals and the residuals. ACF and PACF of residuals has been plotted to show if they are stationary. There is no significant autocorrelation or partial autocorrelation which suggests that the residuals are stationary.

The multivariate JB test was carried out to check for normality of the residuals. The residuals were found to be non-normal with p value of 5.74e-14.

The portmanteau test was carried out to check for the residuals being white noise. The residuals were indeed found to be white noise with p value of 0.0823.

Granger Causality

The set of weather variables containing average maximum temperature, average minimum temperature, relative humidity, sunshine and rainfall were found to be Granger causing rust incidence and severity. Th null hypothesis of the set of weather variables not Granger causing the disease variables can be rejected with p-value = 0.003914.

Impulse response analysis

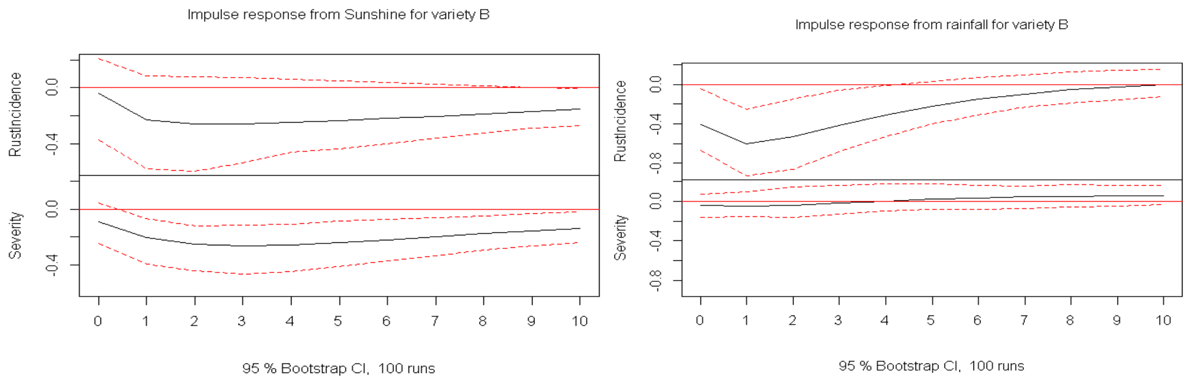


Fig- 3.18 – plots of impulse response from sunshine and rainfall for variety Sln5B on rust and severity

The IRF plots from the above figure shows that a unit shock in sunshine at let's say time t = 0, has a significant impact on rust incidence and induces it to decrease in the coming periods. Rainfall also has similar effects on rust incidence. There is an instantaneous change in rust incidence at t = 0, in case of unit shock in rainfall.

Severity is induced to decrease in the coming periods after a unit shock in sunshine. Severity does not change much in the coming periods after a unit shock in rainfall.

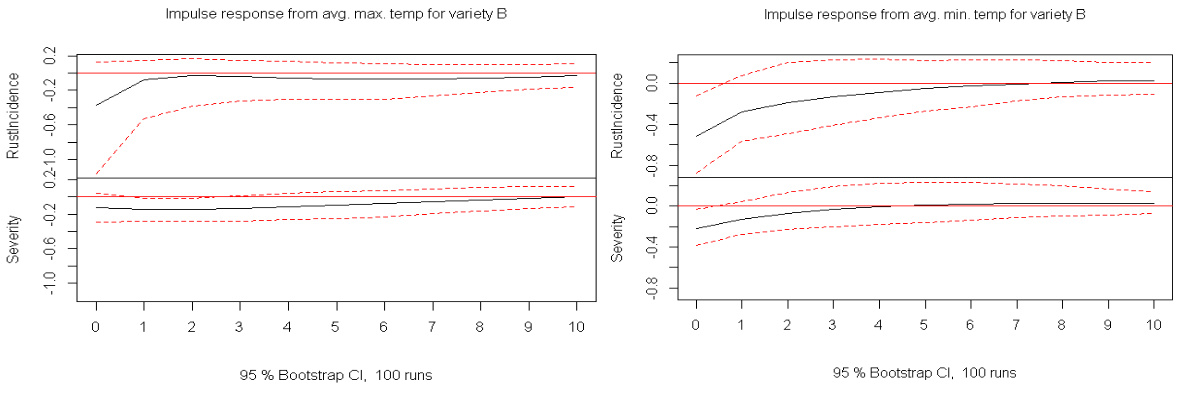


Fig- 3.19 – plots of impulse response from avg max and min temperature for variety Sln5B on rust and severity

A unit shock at let's say t = 0 in average maximum temperature induces rust incidence to decrease in the coming periods. Similarly, for average minimum temperature rust incidence decreases in the coming periods. There is an instantaneous change in rust incidence at t = 0, which suggests instantaneous causality among rust incidence and average maximum and average minimum temperature.

Unit shock in average maximum temperature induces severity to decrease in the coming periods.

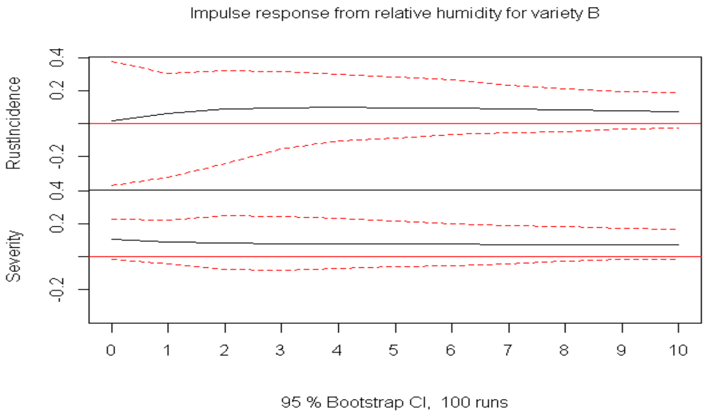


Fig- 3.20 – plots of impulse response from relative humidity for variety Sln5B on rust and severity

A unit shock in relative humidity at t =0 induces the rust incidence to increase in the coming periods. Severity is also affected in a similar fashion by a unit shock in relative humidity.

Prediction plot

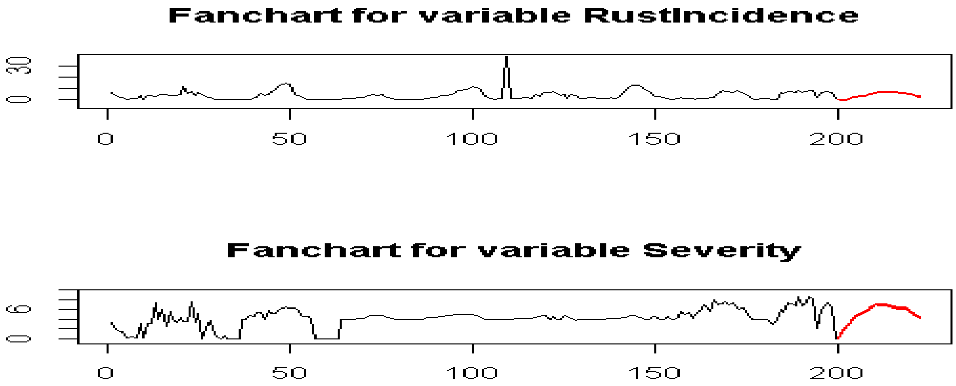


Fig- 3.21 – Fanchart for rust incidence and severity for variety Sln5B

The fanchart for rust incidence and severity show that the two variables will increase and then decrease in the future.

Forecasting

To forecast the disease variables based on the forecasted weather variables, the dataset containing all 8 variables was split into two categories and then fed into VAR. Weather variables were used as exogenous regressors in the VAR model, while disease variables were used as endogenous variables in the model. I had to forecast the disease variables based on the forecasted weather variables.

Out-of-sample forecast

To check the accuracy of the VAR model out of sample forecasting with a rolling origin was done and accuracy of the same was noted. The forecasts and forecasting accuracy has been shown in the latter part of the report when comparison with LSTM has been done.

Chapter Three

Recurrent Neural Network Modelling

Introduction

Recurrent Neural Networks (RNNs) have been very successful on tasks that require sequence learning. RNNs are non-linear models which are capable of learning non-linearities in sequential data. These models have been very successful at tasks which seemed very difficult for feed forward neural networks, such as natural language translation, text classification, text generation etc. They have been successfully applied to tasks such as sequence labelling, where a sequence is the input independent and a label is learned for the input sequence. Much more difficult task is of segment classification where the target sequences consist of multiple labels. RNNs have been successfully used for prediction of time series and there are research papers that have used both traditional time series techniques such as ARIMA and RNN. For some of time series tasks VAR and RNN both have been used and their accuracies have been compared.

The success of RNNs over sequence learning tasks such as time series prediction and the fact that there are studies that compare traditional time series models to RNN are the prime motivation to venture into the field of RNNs for this work.

Literature review

One of the most influential literature that I went through was the book Supervised Sequence Labelling with Recurrent Neural Networks by Alex Graves. It provides a thorough explanation of the basic concepts for understanding RNNs and provides the state-of-the-art problems that it is being applied to. Lipton et. al. (2015) in their paper, A critical review of RNN, provide the motivation for learning RNN by describing how it has been successful for sequence learning tasks. Their review provides both, historical and state-of-the-art perspective for understanding RNNs.

LSTM: A Search Space Odyssey is the paper that proved to be very useful from the viewpoint of tuning RNN architectures. They have demonstrated how the hyperparameter tuning has been carried out and if there are interaction among the hyperparameters. They did conclude that the hyperparameters can be tuned independently for all practical purposes as they could not find any conclusive interaction among them.

In the paper R2N2: Residual Recurrent Neural Networks for Multivariate Time Series Forecasting, Hardik Goel et. al. build three different types of models, namely, VAR, RNN and R2N2. They have compared the accuracy of these models and found that R2N2 was performing either the best or the second best on all the tasks. R2N2 is a hybrid model in which the original time series is modelled using VAR and the residuals from that model are modelled using RNN.

Recurrent Neural Networks

Artificial Neural Networks is a network of small processing units (nodes) joined to each other by weighted connections. The network gets activated by an input vector and that activation spreads throughout the network along the weighted connections. There are two types of artificial neural networks and they are networks with cycles and networks without cycles. ANNs without cycles are Feed forward neural networks for example Multilayer perceptron (MLP). ANNs with cycles are Recurrent neural networks (RNNs) [9].

RNNs are more suitable for sequential learning problems than MLPs, since the output of an MLP depends only on the current input and not on any past inputs, while output of an RNN depends on the current and previous inputs too. These architectures are the natural ones for sequence learning tasks. RNNs have the ability to selectively pass information across sequence steps, while processing sequential data one element at a time. RNNs retain information over long sequences and are perfectly capable of retaining information from very long sequences. But this has not been observed in practice and recurrent neural networks struggle to retain information over very long sequences. This occurs because they suffer from vanishing and exploding gradients problem.

One of the papers [8] I went through had done a review of recurrent neural network and I am using few of those images to illustrate the functioning of a recurrent neural network.

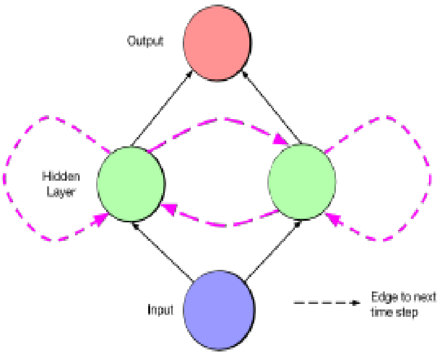


Fig:4.1 A simple RNN[8]

Input to the network through the input layer is just called input. Units in the hidden layer calculate a weighted sum of the input from the input layer. This weighted sum is known as network input to the input, h, and denote it a_h [9].

The above figure shows an input layer, a hidden layer and an output layer of a recurrent neural network. Input at time 't', to the network goes through the input layer to the nodes in the hidden layer. The nodes in the hidden layer apply an activation function on the network input i.e. weighted input from the input layer. Then the activations move to the output layer, exactly as in a feed forward neural network. The activations from time step 't' go back to the node as network input at time step 't+1'. That is the activations arrive at the hidden layer from both the current external input and hidden layer activations from the previous time steps.

Loss function:

It is a function of the difference between predicted and true values of the variable to be predicted. This function should be differentiable and that will enable training of the network through gradient descent. Choice of loss function must be based on the task at hand. The task here is of regression so I have used mean squared error as the loss function which is differentiable.

Back propagation through time:

Back Propagation is a repeated application of chain rule of partial derivatives. The loss function, which must be minimized, is differentiated with respect to the output units.

Training of an RNN is done by Back Propagation Through Time (BPTT). BPTT is like Back propagation and consist of repeated application of chain rule. It is important to note that the loss function depends on the hidden layer through [9]:

a) The influence of the hidden layer on output layer.
b) Influence of the hidden layer on the next time step.

Unfolding:

For ease of understanding RNNs can be visualized by unfolding the network along the input sequence. An unfolded RNN is shown below.

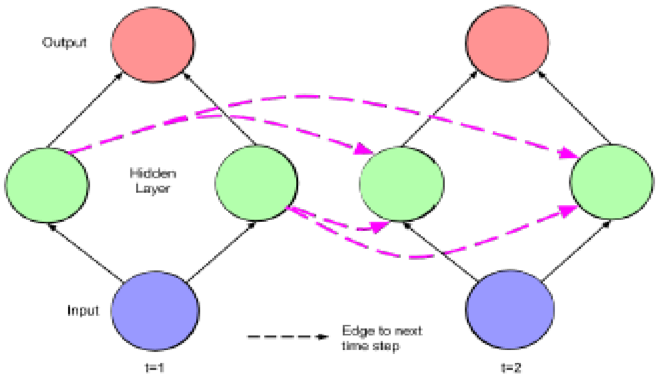


Fig- 4.2 Unfolded RNN [8]

The network can be interpreted as not cyclic but like a feed forward network that has one layer for each time step. The network also consists shared weights across time steps. This picture makes it clear how back propagation can be applied across many timesteps so that the network can be trained [8].

Vanishing and exploding gradients

Due to the requirement of learning long range dependencies, learning with recurrent neural networks can be especially challenging [8]. When back propagating gradients across many timesteps the problems of vanishing and exploding gradients occur. The influence of a given input on the hidden layer and therefore on the network output, either decays or blows up exponentially as it cycles around the recurrent connections of the network.

Vanishing gradients problem refers to the exponential decay of gradients in terms of magnitude as they are propagated back through time. This causes the memory to ignore long-term dependencies and not learn any association between two events separated by long lag [10].

Next major problem is of exploding gradients. When the network has to learn over long range dependencies the gradients explode when the weights are big. This problem refers to the fact that the large increase in the norm of the gradient during training sequences with long-term dependencies leads to the explosion of long-term components [10].

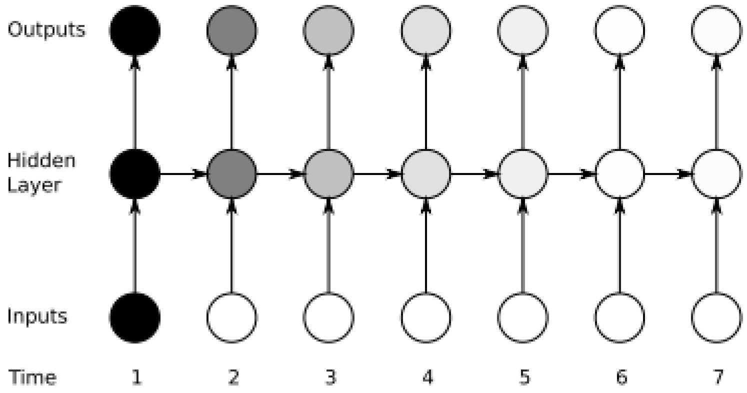


Fig 4.3: Vanishing gradient problem for RNN [11]

The shading of the nodes in the unfolded network indicate the sensitivity to the inputs at time 1. The sensitivity decreases over time as new inputs overwrite the activations of the hidden layer and the network forgets its first inputs.

Long Short-Term Memory (LSTM)

To address the issue of vanishing and exploding gradients, the nodes in the hidden layers of the network are replaced by so called memory blocks. The architectural variant of recurrent neural networks with memory blocks are referred to as long short-term memory (LSTM) [8]. In practice, the LSTM has shown a superior ability to learn long range dependencies as compared to simple RNNs.

The long short-term memory (LSTM) architecture was proposed in 1997, primarily to deal with the problem of exploding and vanishing gradients by Hochreiter and Schmidhuber [8], [10]. The LSTM architecture replaces nodes in the hidden layer of an RNN by an LSTM memory block. The block contains three multiplicative units, namely, input, output and forget gates. These gates are continuous analogues of write, read and reset operations for the cells. The block also contains a memory cell which is self-connected and is the most important part of the block. The cell retains information over long sequences and is updated when new inputs arrive at the network.

These multiplicative gates allow the memory cells to store and access information over long periods and so they mitigate the problems of vanishing gradients. This is accomplished as follows:

Suppose the input gate remains closed which means an activation of zero, the activation of the cell will not be overwritten by the new inputs coming to the network. Thus, the activation of the cell will be available for arbitrarily long time and will be available to the network later [11].

The gates control the activation of the cell via multiplications. The gates have sigmoid activations and hence are non-linear. Their output is between 0 to 1, where 0 means the gate is closed and 1 means gate is open. The forget gate multiplies the cell's previous state and the input and output gates multiply the input and output of the cell [11]. The figure below illustrates an LSTM memory block.

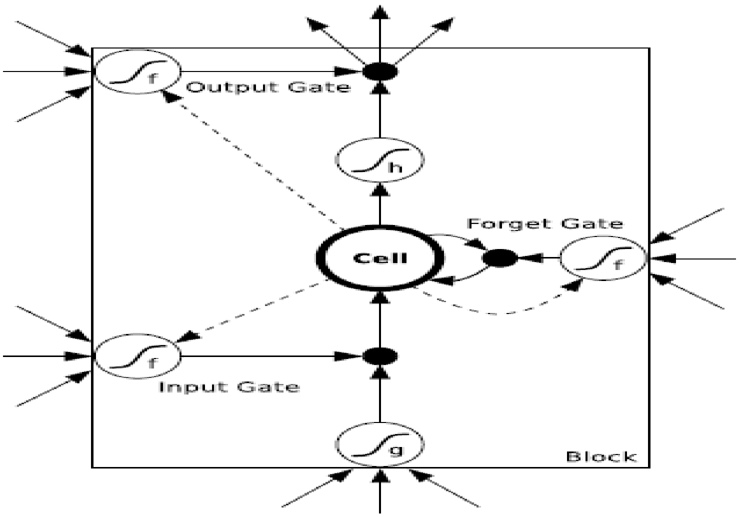


Fig 4.4 . An LSTM memory block. Source: [11]

As we can see in the figure the cell is connected to it's previous state through the forget gate multiplication which basically controls what information to learn from the previous timestep.

The output from the block passes through the output gate multiplication and goes to the rest of the network.

The supervised learning tasks on which LSTM has been successfully applied to are Natural language translation, image captioning, time series forecasting etc. [10]

Hyper parameters tuning

Hyper parameter tuning is one of the most important aspect of machine learning. To find the optimal combination of the hyper parameter is an iterative process of searching among the infinite possibilities of hyperparameters. One of the most important hyperparameters that needs to be tuned is the learning rate. There have been previous research efforts over explaining if any dependencies among the hyperparameters [13]. The most critical components of the LSTM architecture found was the forget gate and the output activation function. The hyperparameters appeared virtually independent [13]. It was found that for practical purposes hyperparameters can be treated as approximately independent and can be optimized separately [13]. For this project the hyperparameters were tuned separately using grid search, implemented in python.

Notes on tuning

- There is no apparent hyperparameter interactions structure [13].
- For all practical purposes the hyperparameters can be treated approximately independent [13].

Approach towards tuning

- Optimizer was tuned first through grid search.
- Then network size, activation function, weight initializer was tuned. (not in order)
- Regularizers were tuned then.
- Number of epochs and batch size was tuned first using grid search.
- Final selection of optimal hyperparameters was done after final tuning by hand.

Tuned Hyperparameters

A total of six LSTM models were tuned for the six variables that had to be predicted. All of them were optimized using nadam optimizer. Initializer, activation function, optimizer, batch size and number of layers was common across all the six models. L2 regularization was used in all the six models.

Dropout has been very successful in the feed forward neural networks, but it does not have much empirical evidence suggesting successfull application of the same in recurrent neural networks [14]. They show that using dropout can greatly reduce overfitting in LSTM models. I have used dropout and recurrent dropout both in few of the models. Dropout indeed reduces overfitting for this problem.

Below is a table showing the final hyperparameters:

Hyper parameters	S795 rust	S795 severity	Sln5B rust	Sln5B severity	CXR rust	CXR severity
Layers	1	1	1	1	1	1
Neurons	3	3	5	10	5	8
Initializer	Glorot normal	Glorot Normal	Glorot Normal	Glorot Normal	Glorot Normal	Glorot Normal
activation	elu	elu	elu	elu	elu	elu
Optimizer	Nadam	Nadam	Nadam	Nadam	Nadam	Nadam
Regularizer	L2	L2	L2	L2	L2	L2
Dropout	0.005	None	None	0.03		0.005
Recurrent dropout	None	0.40	0.01		0.5	0.5
Batch size	1	1	1	1	1	1
Epochs	100	60	100	100	100	100

Table 3.1 Hyperparameter table

Loss Curves

This section illustrates the loss curves for the given hyperparameters in the above table. Blue line shows the loss in training and orange ones show the loss during validation.

For the variety CXR the loss curves show good generalisation as compared to the other two varieties.

Some of the models are generalizing well and in some cases the validation loss is low compared to the training loss. This could be because the data for validation might have been ‘easy’ to predict as compared to the data used for learning during training. Different train/test splits could not be tried in this case as this is a time series problem, so splitting randomly is out of question. And the number of data points are less, i.e. around 20 for validation.

Variety S795

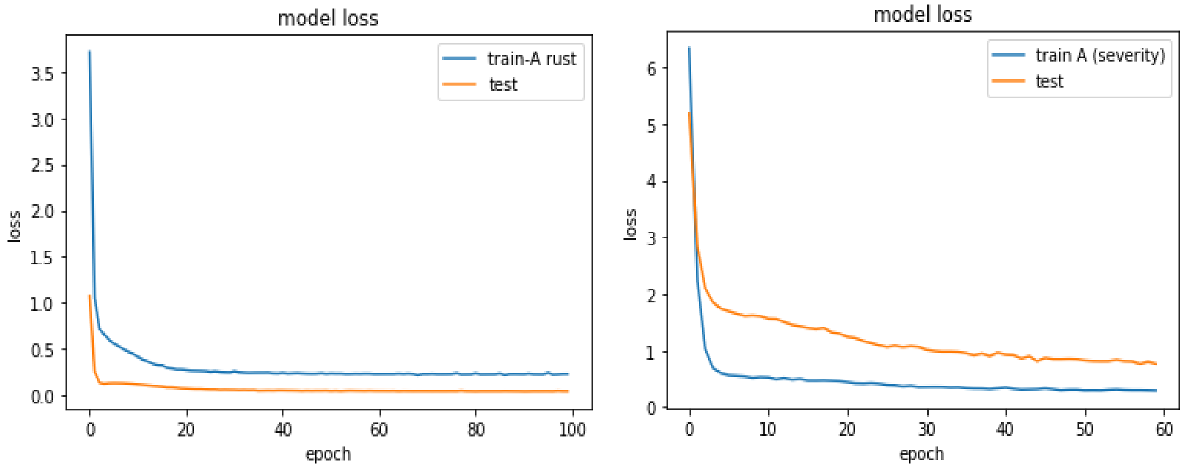


Fig- 4.5 Model loss for variety S795

Variety Sln5B

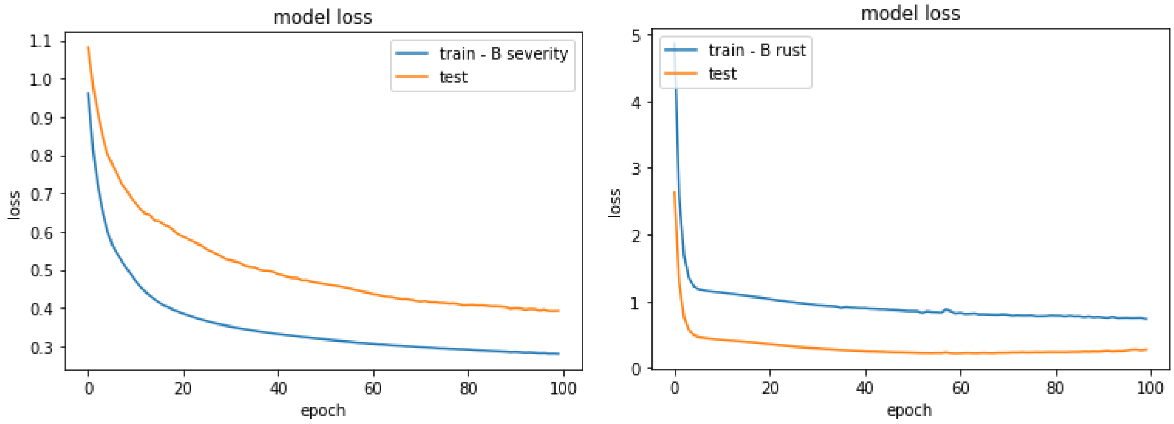


Fig- 4.6 Model loss for variety Sln5B

Variety CXR

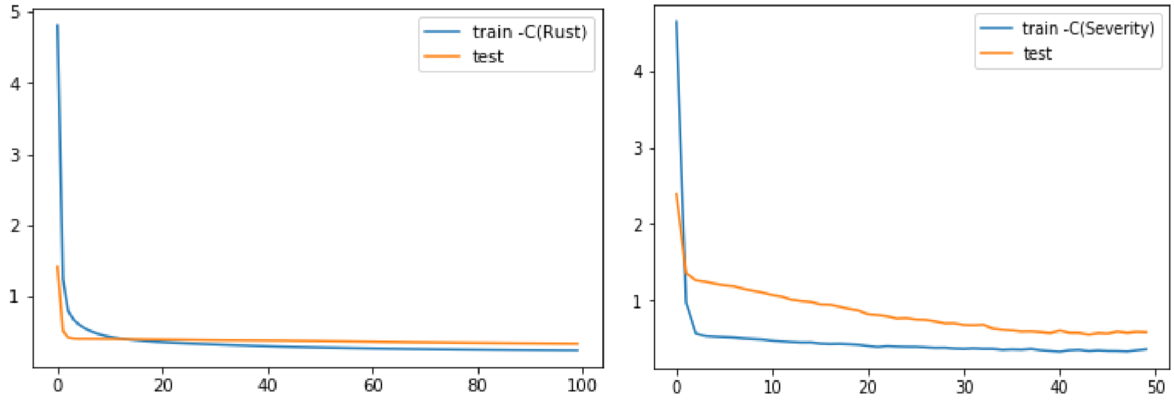


Fig- 4.7 Model loss for variety CXR

Chapter Four

Comparison of forecasts from VAR and LSTM:

Rolling forecasts were generated from both VAR and LSTM and were evaluated on two below mentioned error metrics.

a) Root mean squared error (RMSE)
b) Mean relative squared error (MRSE)

RMSE and MRSE both should be small for good forecasts. MRSE measures the goodness of a forecast as compared to predicting the mean of the variable every time. RMSE is an absolute measure of error and is affected by scale of the variables. Smaller values of both of these measures is desirable.

Variety S795

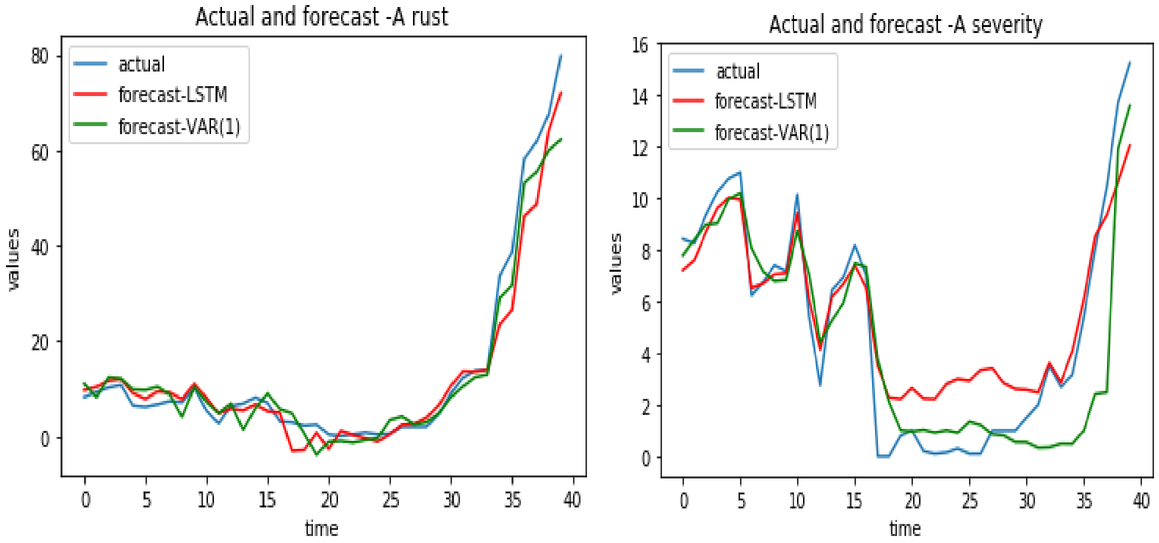


Fig- 5.1 prediction plots for variety S795 VAR and LSTM

	RMSE	MRSE
LSTM	4.41	0.22
VAR	4.16	0.21

	RMSE	MRSE
LSTM	1.64	0.38
VAR	2.17	0.52

Table 4.1 Accuracy measures for variety S795

For variety S795 rust incidence forecasts LSTM is outperformed by VAR while VAR is outperformed by LSTM for forecast of severity.

Variety Sln5B

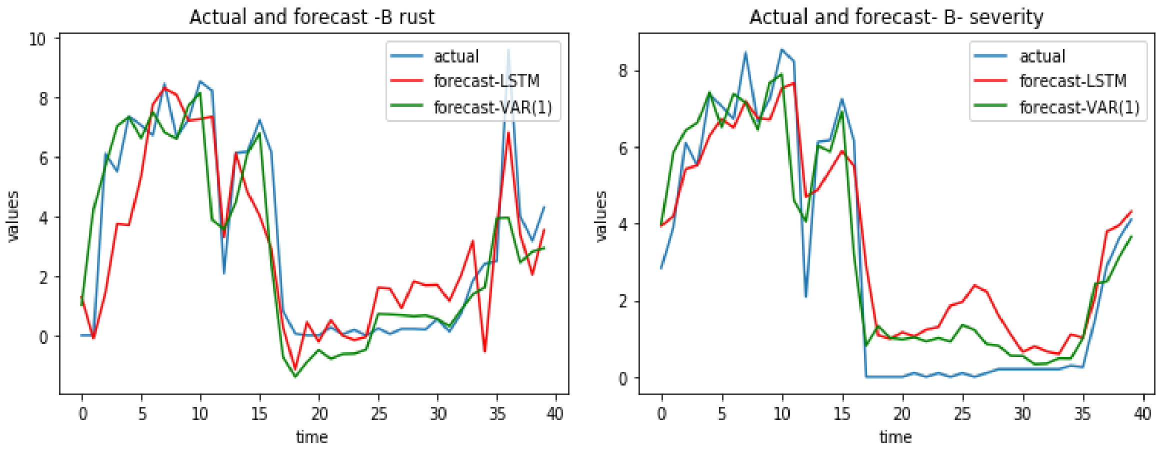


Fig- 5.2 prediction plots for variety Sln5B VAR and LSTM

	RMSE	MRSE		RMSE	MRSE
LSTM	1.65	0.511	LSTM	1.18	0.37
VAR	1.67	0.516	VAR	1.11	0.35

Table 4.2 Accuracy measures for variety Sln5B

LSTM is better at forecasting rust for variety Sln5B while VAR is better at forecasting severity.

Variety CXR

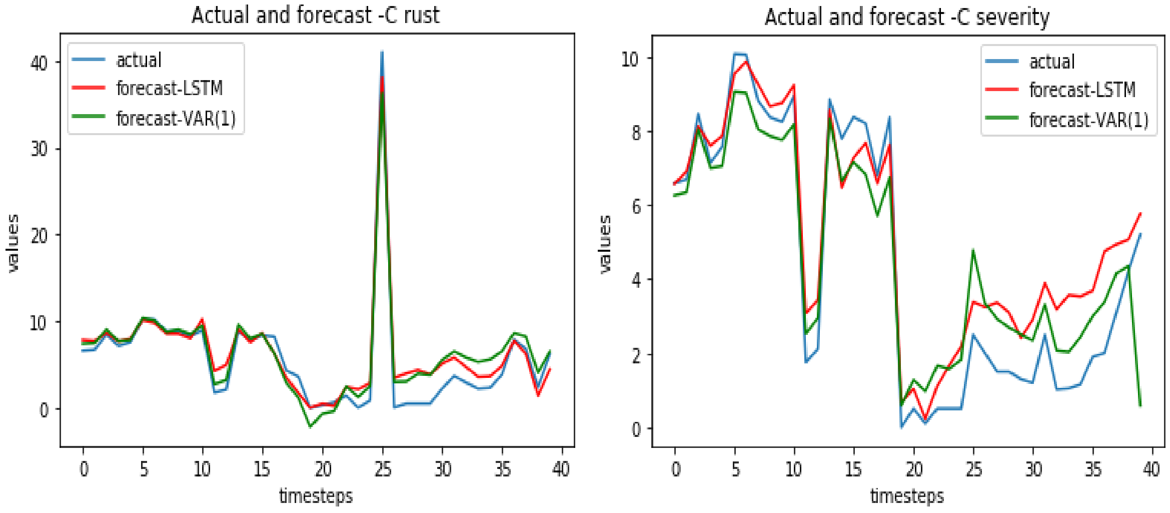
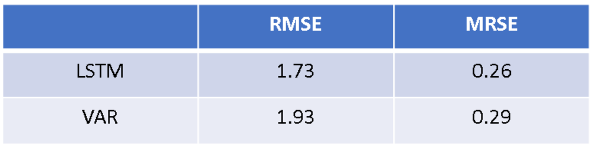
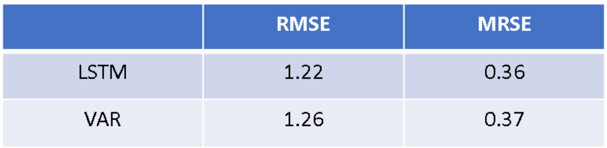


Fig- 5.3 prediction plots for variety CXR VAR and LSTM



	RMSE	MRSE
LSTM	1.73	0.26
VAR	1.93	0.29



	RMSE	MRSE
LSTM	1.22	0.36
VAR	1.26	0.37

Table 4.3 Accuracy measures for variety CXR

LSTM is performing better than VAR for both rust incidence and severity in case of the variety CXR. So, LSTM has outperformed VAR on four out of the six tasks of prediction.

Conclusion

The major objective of this work was prediction of coffee leaf rust incidence and severity. The same has been accomplished using Vector Autoregression (VAR) and Long Short-Term Memory (LSTM). Three VAR(1) model were developed for forecasting rust incidence and severity for all the three varieties and a total of six LSTM models were developed for forecasting the same.

Inferences using impulse response analysis was drawn which showed how a unit shock in weather variables induce disease variables to change in the coming period.

VAR, along with forecasts, provides tools for interpretation of the relationships among the variables in a system. The tools such as Granger causality, Impulse response analysis have been used to interpret the causal relations among the weather and the disease variables. LSTM is a complex machine learning model. It lacks tools for interpretation but at the same time it has outperformed VAR in four out of six predictions, i.e. rust incidence and severity for each of the three varieties.

Based on the data and the approach taken in this work, LSTM is recommended for forecasting the rust incidence and severity while VAR must be used for understanding the structural relationships among the weather and disease variables.

Future work

With more data machine learning model's performance increases and so with new data one can expect to get better forecasting accuracy for LSTM than that is achieved now.

Hybrid models can be used for improving the forecasts. VAR can be used to model the multivariate time series and the residuals from VAR can be modelled using RNN.

References

1) https://en.wikipedia.org/wiki/Coffee_production_in_India
2) http://www.ico.org/documents/cy2013-14/presentations/icc-india-march2014.pdf
3) Prediction of Coffee Rust Disease Using Bayesian Networksc - Cora B. P´erez-Ariza
4) The coffee rust crises in Colombia and Central America (2008–2013): impacts, plausible causes and proposed solutions, Jacques Avelino et. al.
5) Towards Detecting Crop Diseases and Pest by Supervised Learning - David Camilo Corrales et. al
6) https://en.wikipedia.org/wiki/Hemileia_vastatrix
7) Helmut Lütkepohl-New Introduction to Multiple Time Series Analysis-Springer (2006)
8) The intensity of a coffee rust epidemic is dependent on the production situation – J. Avelino et. al
9) A Guideline for Building Large Coffee Rust Samples Applying Machine Learning Methods, David Camilo Corrales et. al
10) A Critical Review of Recurrent Neural Networks for Sequence Learning, Lipton et. al.
11) Supervised sequence labelling with Recurrent Neural Networks, Alex Graves
12) Learning over long time lags, Hojjat Salehinejad
13) LSTM: A search space odyssey, Greff et. al.
14) Recurrent neural networks regularization, Zaremba et. al.
15) R2N2: Residual Recurrent Neural Networks for Multivariate Time Series Forecasting, Hardik Goel et. al.
16) A theoretically Grounded Application of Dropout in Recurrent Neural Networks, Yarin Gal et. al.

Appendix

Modelling Variety CXR

Rust incidence:

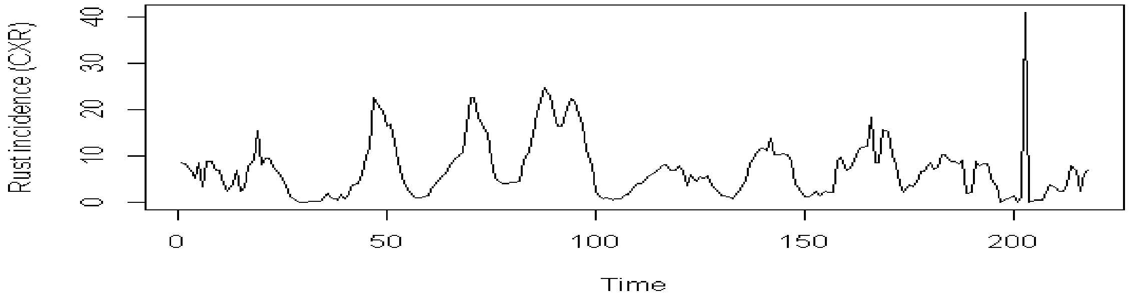
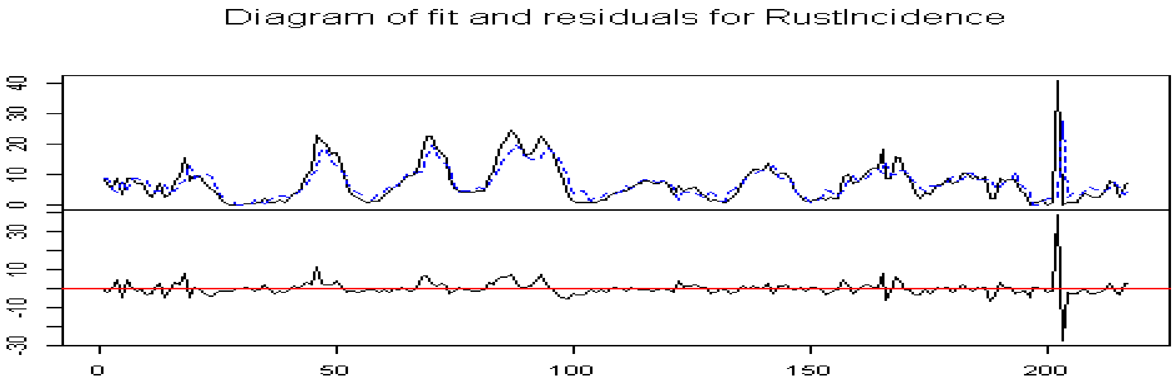


Fig- 3.22 – Time series plot of rust incidence for variety CXR

```
Estimation results for equation RustIncidence:
==============================================
RustIncidence = RainFall.l1 + TempMax.l1 + TempMin.l1 + RelativeHumidity.l
1 + PopulationInfected.l1 + RustIncidence.l1

                       Estimate Std. Error t value Pr(>|t|)
RainFall.l1           -0.003243   0.001440  -2.253 0.025317 *
TempMax.l1            -0.189367   0.086906  -2.179 0.030439 *
TempMin.l1            -0.497049   0.217853  -2.282 0.023512 *
RelativeHumidity.l1    0.185665   0.050201   3.698 0.000277 ***
PopulationInfected.l1  0.059147   0.019653   3.010 0.002935 **
RustIncidence.l1       0.518575   0.071922   7.210 9.82e-12 ***
```

Out of the weather variables, average maximum temperature, average minimum temperature and relative humidity are significant at lag 1.



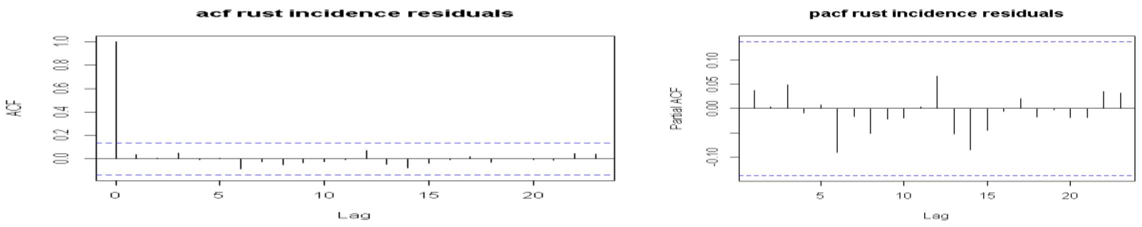
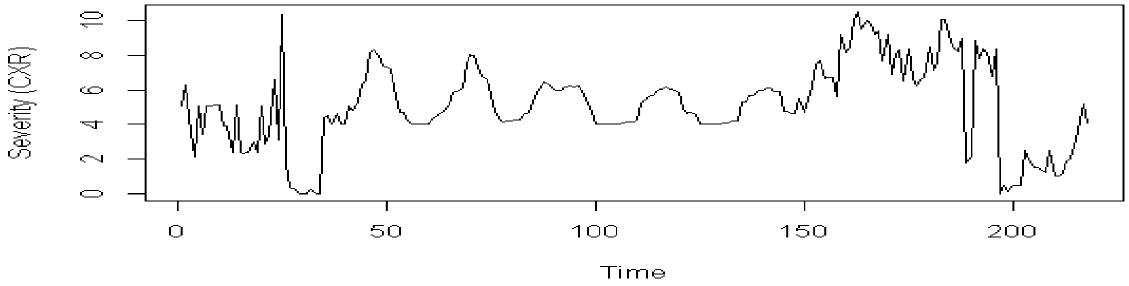


Fig- 3.23 – plot of fit vs residuals and ACF, PACF plots of residuals for rust incidence for variety CXR

The above plot shows the plot of fit versus actuals and the residuals. ACF and PACF of residuals has been plotted to see if they are stationary. There are no significant ACF or PACF which suggests that the residuals are stationary.

Severity



```
Estimation results for equation Severity:
=========================================
Severity = Sunshine.l1 + RelativeHumidity.l1 + PopulationInfected.l1 + Sev
erity.l1

                       Estimate Std. Error t value Pr(>|t|)    
Sunshine.l1           -0.116938   0.054151  -2.159 0.031931 *  
RelativeHumidity.l1    0.013809   0.003765   3.668 0.000309 ***
PopulationInfected.l1  0.018188   0.006559   2.773 0.006047 ** 
Severity.l1            0.704173   0.055770  12.626  < 2e-16 ***
```

Sunshine, relative humidity, population infected and severity at lag 1 are significant.

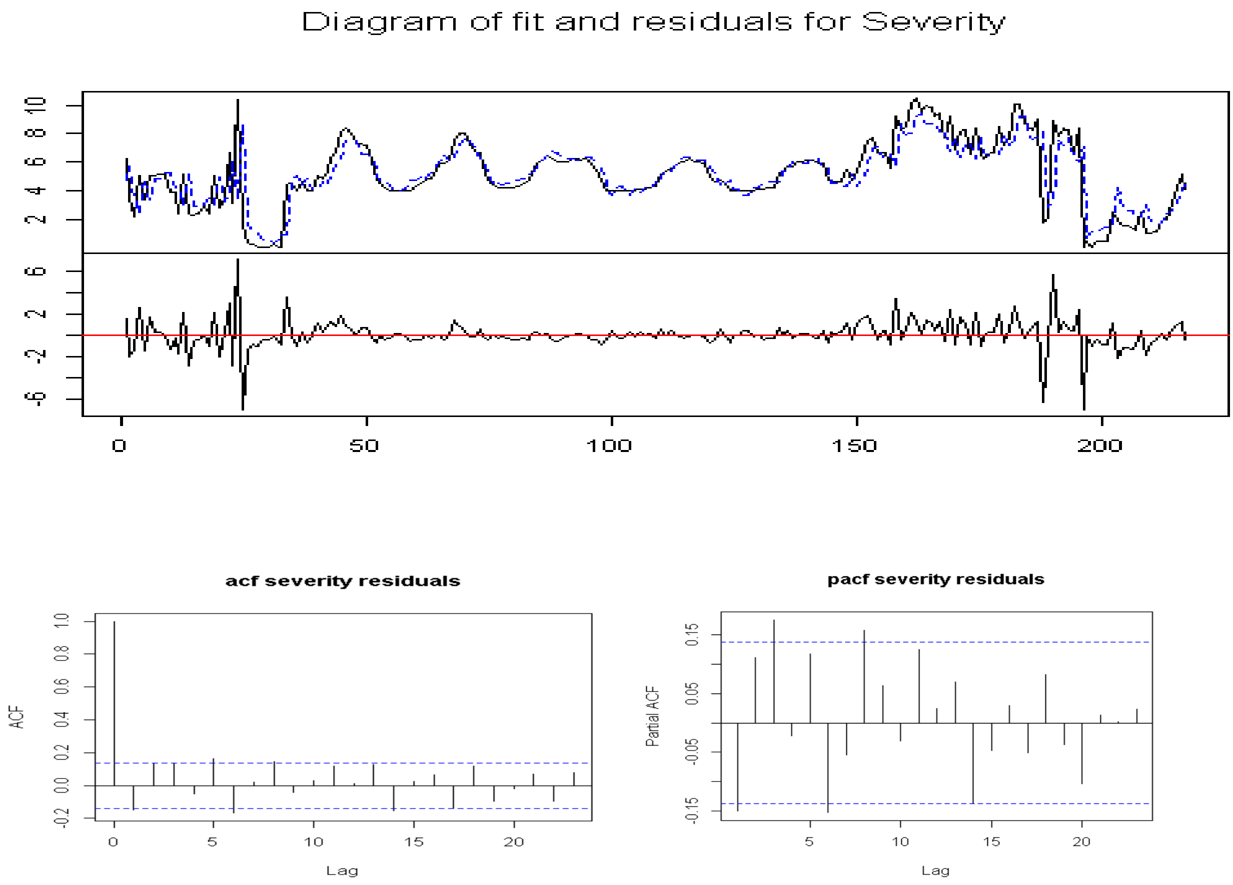


Fig- 3.24 – plot of fit vs residuals and ACF, PACF plots of residuals for severity for variety CXR

The above plot shows the plot of fit versus actuals and the residuals. ACF and PACF of residuals has been plotted to show if they are stationary. The ACF and PACF plots of the residuals for severity look fine. This is a sign of stationary residuals.

The multivariate JB test was carried out to check for normality of the residuals. The residuals were found to be non-normal with p value of 2.2e-16.

The portmanteau test was carried out to check for the residuals being white noise. The residuals were indeed found to be white noise with p value of 0.2104

Granger Causality

The set of weather variables containing average maximum temperature, average minimum temperature, relative humidity, sunshine and rainfall were found to be Granger causing rust incidence and severity. Th null hypothesis of the set of weather variables not Granger causing the disease variables can be rejected with p-value = 1.365e-05.

Impulse response analysis

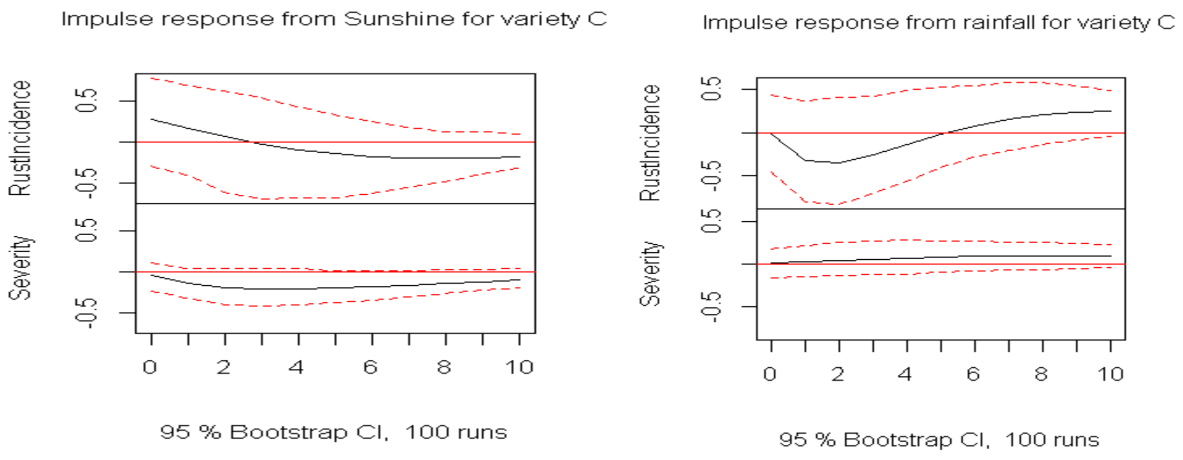


Fig- 3.25 – Impulse response plots from Sunshine and rainfall for variety CXR

A unit shock in sunshine at time t =0, induces the rust incidence to increase in the coming period and then it starts decreasing after second period. Severity is induced to decrease due to a unit shock in sunshine.

A unit shock in rainfall at time t =0, induces the rust incidence to decrease in the coming period and then it increases after 5th period. Severity does change much due to a unit shock in rainfall.

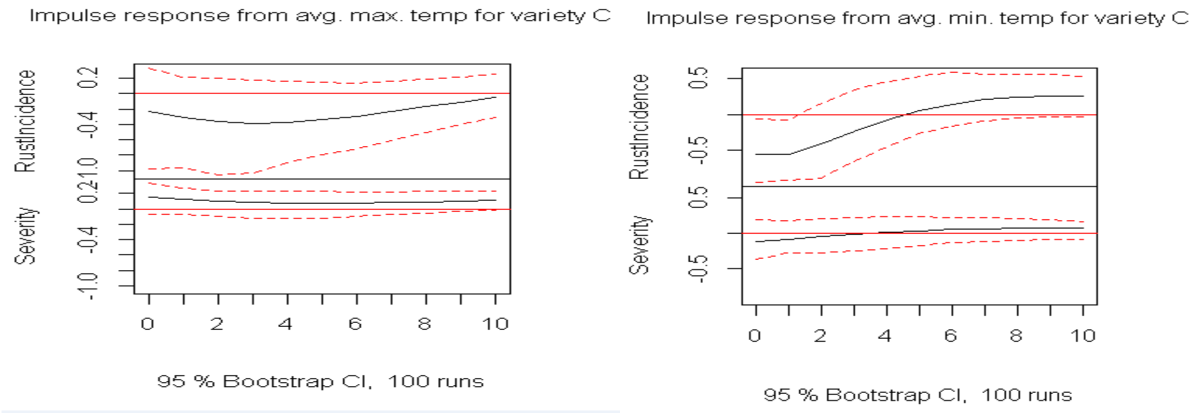


Fig- 3.26 – Impulse response plots from average max and min temperature for variety CXR

A unit shock in average maximum temperature at time t = 0, induces rust incidence to decrease in the coming period and induces severity to increase in the coming periods.

A unit shock in average minimum temperature at time t = 0, induces rust incidence to decrease in the coming periods and then increase after 4th period. Severity is not much affected due to a unit shock in average minimum temperature.

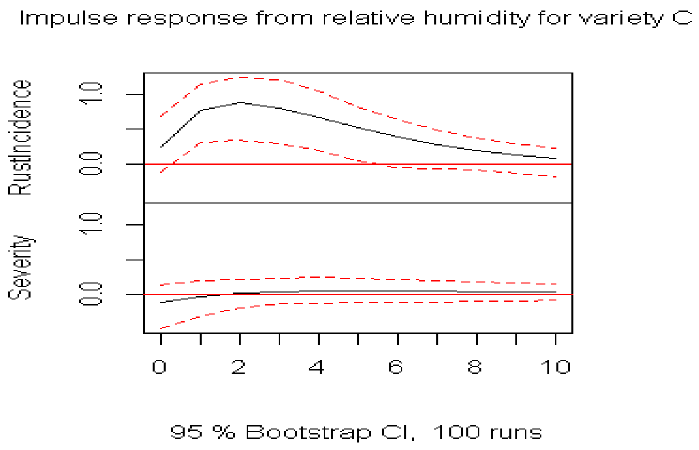


Fig- 3.27 – Impulse response plots from relative humidity for variety CXR

A unit shock in relative humidity at time t = 0, induces rust incidence to increase in the coming period but the same shock does not affect severity much.

Prediction plot

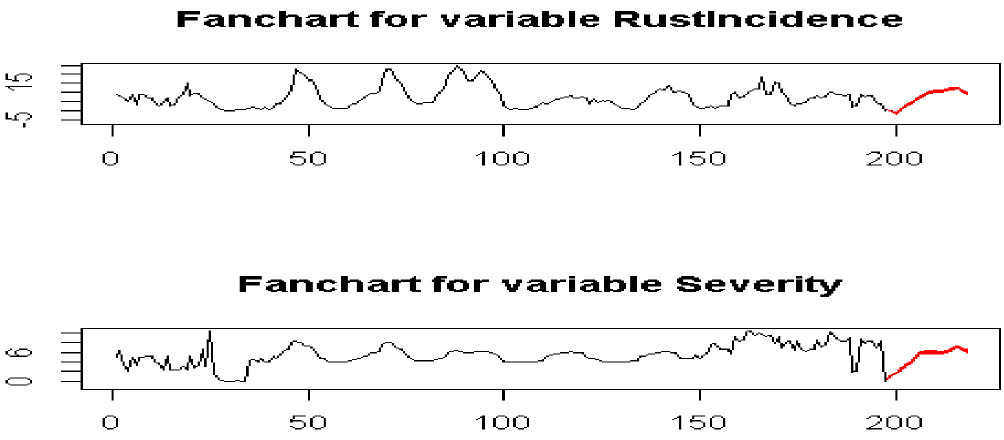


Fig- 3.28 – Fanchart for variety CXR

The fanchart shows that both rust incidence and severity will increase in the future.