Modelos NN

Jimena Murillo

2022-05-05

Paquetes

```
library(keras) # for deep learning
library(tidyverse) # general utility functions
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.6     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(caret) # machine learning utility functions
```

```
## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(tibble)
library(readr)
library(ggplot2)
library(tensorflow)
```

```
##
## Attaching package: 'tensorflow'

## The following object is masked from 'package:caret':
##
##     train
```

Construir una base con el cantón de Alajuela y partirla en train y test

```
load("C:/Users/usuario1/Desktop/CIMPA/Github_CIMPA/PRACTICA_CIMPA/base_cantones.RData")




Alajuela1 <- basecanton %>% filter(Canton == "Alajuela")

Alajuela1 <- Alajuela1%>%
  dplyr::select(Year,Month,Nino12SSTA,Nino3SSTA, Nino4SSTA, Nino34SSTA,TNA,EVI,NDVI,NDWI,LSD,LSN,Precip_

  arrange(Year,Month) %>% ungroup() %>% mutate(Month=as.numeric(Month))




if(anyNA(Alajuela1)){
  Alajuela1 <- na.omit(Alajuela1)
}

#Escala


normalize <- function(x) {
  return ((x - min(x)) / (max(x) - min(x)))
}

max <- apply(Alajuela1,2,max)
min <- apply(Alajuela1,2,min)

Alajuela1.2 <- apply(Alajuela1, 2, normalize)


#Train y test

data_train1 = as.data.frame(Alajuela1.2) %>% filter(Year < 0.85)#PARA ENTRENAR HASTA 2018
data_test1 = as.data.frame(Alajuela1.2) %>% filter(Year >= 0.85)

X_train1 = as.matrix(data_train1[,-ncol(data_train1)])
y_train1 = as.matrix(data_train1[,ncol(data_train1)])

X_test1 = as.matrix(data_test1[,-ncol(data_test1)])
y_test1 = as.matrix(data_test1[,ncol(data_test1)])
```

Base de datos con lag

```r
Alajuela <- basecanton %>% filter(Canton == "Alajuela") %>%

  dplyr::select(Year,Month,Nino12SSTA, Nino3SSTA, Nino4SSTA,Nino34SSTA,Nino34SSTA1, Nino34SSTA2, Nino34S

  arrange(Year,Month) %>% ungroup() %>% mutate(Month=as.numeric(Month))


if(anyNA(Alajuela)){
  Alajuela <- na.omit(Alajuela)
}

#Escala


normalize <- function(x) {
  return ((x - min(x)) / (max(x) - min(x)))
}

max <- apply(Alajuela,2,max)
min <- apply(Alajuela,2,min)

Alajuela2 <- apply(Alajuela, 2, normalize)


#Train y test

data_train = as.data.frame(Alajuela2) %>% filter(Year < 0.85)#PARA ENTRENAR HASTA 2018
data_test = as.data.frame(Alajuela2) %>% filter(Year >= 0.85)

X_train = as.matrix(data_train[,-ncol(data_train)])
y_train = as.matrix(data_train[,ncol(data_train)])

X_test = as.matrix(data_test[,-ncol(data_test)])
y_test = as.matrix(data_test[,ncol(data_test)])



#Almacen de eval de los modelos

mse = NULL
MSE = NULL
MSE_results = NULL


metricas <- function(tabla){
  NRMSE <- mean((tabla$fit-tabla$RR)^2)/mean(tabla$RR)
  return(data.frame(NRMSE))
}
```

Planteamiento de modelos:

Modelos con datos simples (sin lag)

MODELO 1

```
set.seed(123)
model <- keras_model_sequential()
```

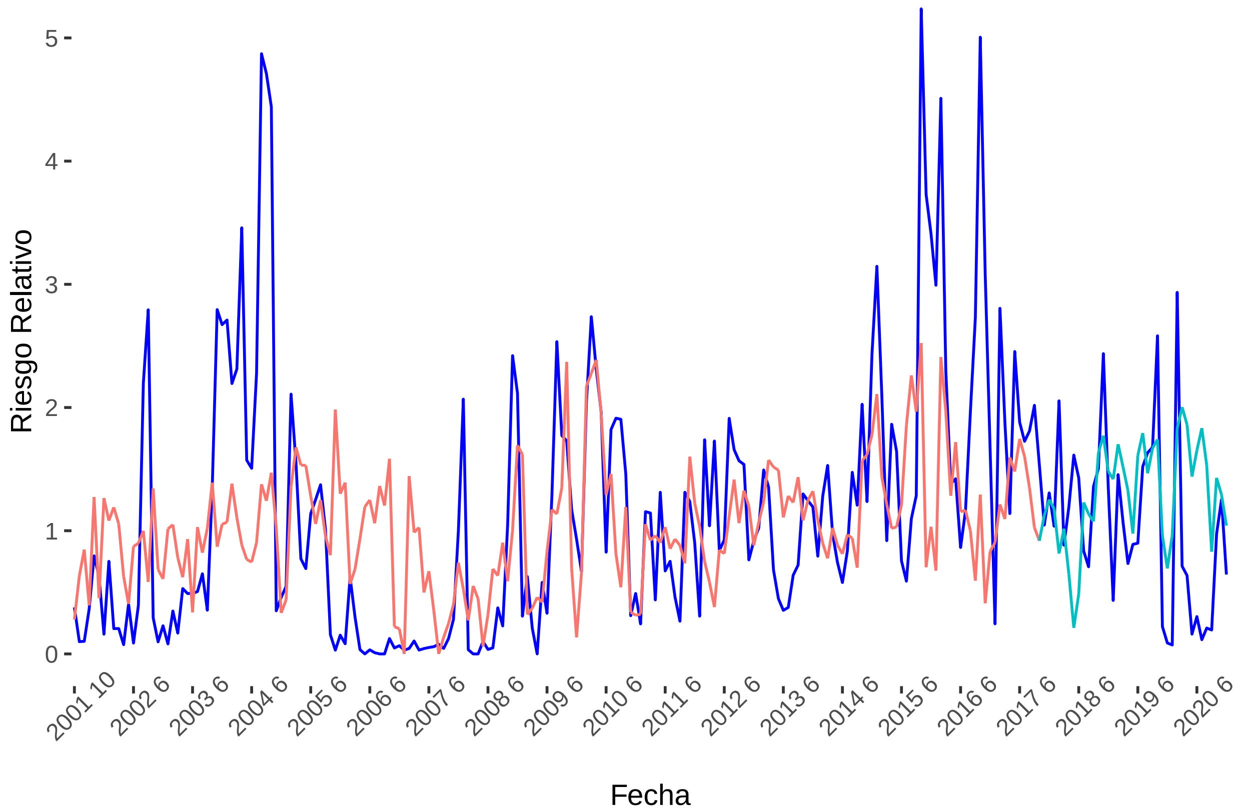
```
## Loaded Tensorflow version 2.8.0
```

```
# our input layer
model %>%
  layer_dense(input_shape = ncol(X_train1), units = 13) %>%
  layer_dense(units = 1, activation = "relu")


# look at our model architecture
summary(model)
```

```
## Model: "sequential"
## ________________________________________________________________________________
##  Layer (type)                       Output Shape                    Param #
## ================================================================================
##  dense_1 (Dense)                    (None, 13)                      182
##
##  dense (Dense)                      (None, 1)                       14
##
## ================================================================================
## Total params: 196
## Trainable params: 196
## Non-trainable params: 0
## ________________________________________________________________________________
```

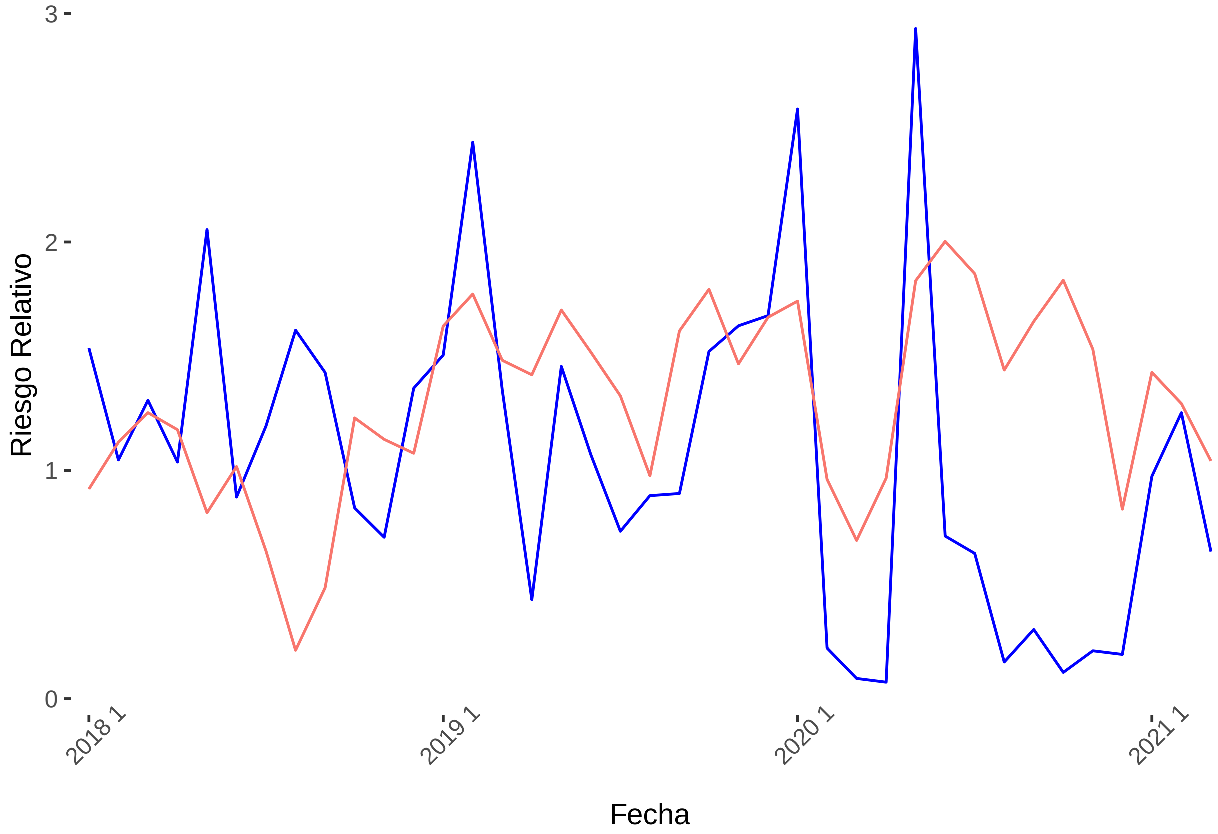
```
model %>% compile(loss = "mse",
                  optimizer = "adam",
                  metric = "mae")

trained_model <- model %>% fit(
  x = X_train1, # sequence we're using for prediction
  y = y_train1, # sequence we're predicting
  batch_size = 18, # how many samples to pass to our model at a time
  epochs = 50, # how many times we'll look @ the whole dataset
  validation_split = 0.2) # how much data to hold out for testing as we go along

mse[1] = (model %>% evaluate(X_test1, y_test1))[1]

#Escala

denorm <- function(x, max, min) {
  return (x*(max - min)+min)
}
```

```r
max <- apply(Alajuela1,2,max)
min <- apply(Alajuela1,2,min)


#Gráfico

pred = denorm(model %>% predict(X_train1),  max[length(Alajuela1)], min[length(Alajuela1)])
results = denorm(model %>% predict(X_test1), max[length(Alajuela1)], min[length(Alajuela1)])


var2 = c(rep(0,length(pred)), rep(1, length(results)))

pred = rbind(pred, results)


data1 = as.data.frame(cbind(pred, Alajuela1$RR))
names(data1) = c("fit", "RR")

MSE[1] = metricas (data1)


data2 = as.data.frame(cbind(results, Alajuela1$RR[197:235]))
names(data2) = c("fit", "RR")

MSE_results[1] = metricas(data2)

Fecha = paste(Alajuela1$Year, Alajuela1$Month)

everyother1 <- function(x) x[(seq_along(Fecha) + 5)%%12 == 6]
everyother2 <- function(x) x[(seq_along(Fecha))%%12 == 1]

p1 <- ggplot(data1, aes(x = Fecha, y = RR, group = 1)) + geom_line(colour = "blue") +
  geom_line( aes(x = Fecha, y = fit, colour = (var2>0)))+
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
  panel.background = element_blank(), axis.text.x = element_text(angle = 45), legend.position = "none" )
  scale_x_discrete(breaks = everyother1) + labs (x = "Fecha", y = "Riesgo Relativo")

p2 <- ggplot(data2, aes(x = Fecha[197:235], y = RR, group = 1)) + geom_line(colour = "blue") +
  geom_line( aes(x = Fecha[197:235], y = fit, colour = "red"))+
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
  panel.background = element_blank(), axis.text.x = element_text(angle = 45), legend.position = "none" )
  scale_x_discrete(breaks = everyother2) + labs (x = "Fecha", y = "Riesgo Relativo")


print(p1)
```



Riesgo Relativo
5
4
3
2
1
0
2001 10
2002 6
2003 6
2004 6
2005 6
2006 6
2007 6
2008 6
2009 6
2010 6
2011 6
2012 6
2013 6
2014 6
2015 6
2016 6
2017 6
2018 6
2019 6
2020 6
Fecha


```
print(p2)
```



MODELO 2

Se agrega una capa

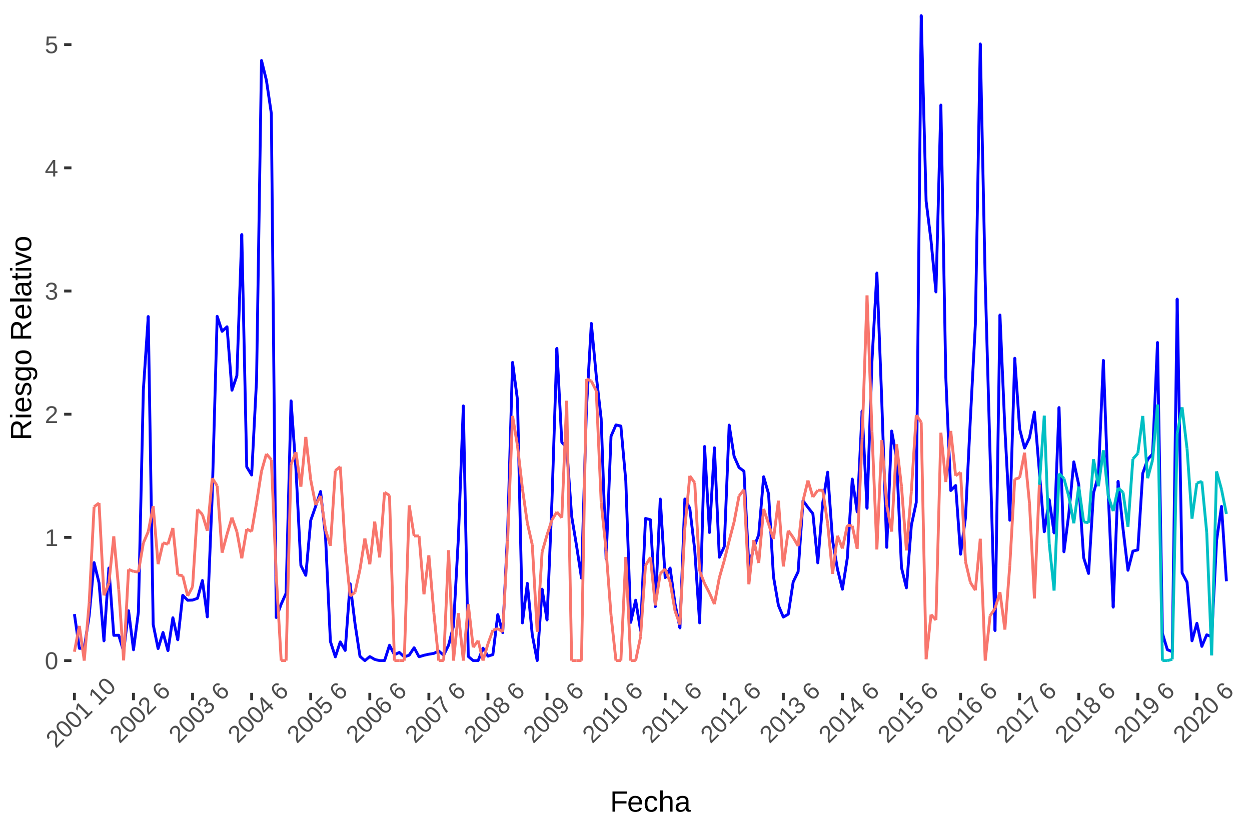
```
set.seed(123)
model2 <- keras_model_sequential()
# our input layer
model2 %>%
  layer_dense(input_shape = ncol(X_train1), units = 13) %>%
  layer_dense(units = 8, activation = "relu")%>%
  layer_dense(units = 1, activation = "relu")


# look at our model architecture
summary(model2)
```

```
## Model: "sequential_1"
## ________________________________________________________________________________
##  Layer (type)                       Output Shape                      Param #
## ================================================================================
##  dense_4 (Dense)                    (None, 13)                        182
##
##  dense_3 (Dense)                    (None, 8)                         112
##
##  dense_2 (Dense)                    (None, 1)                         9
##
```

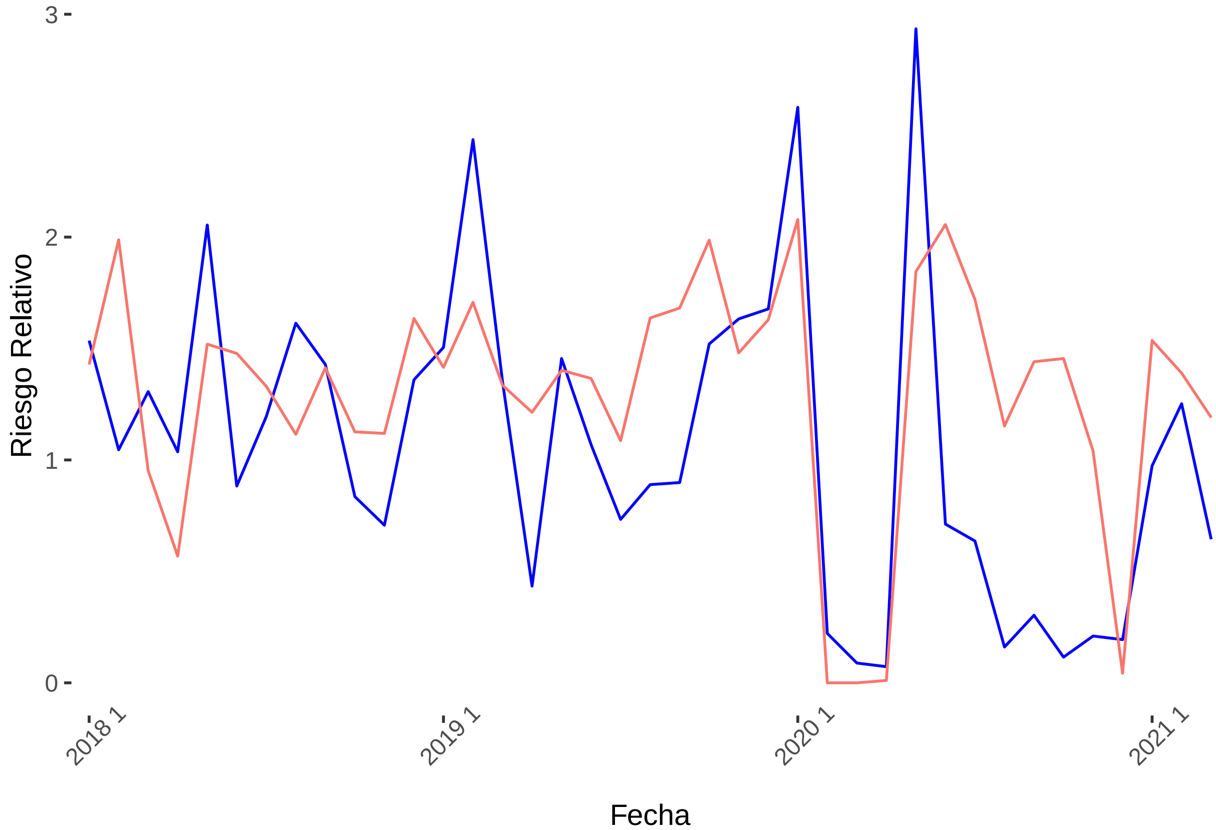
```
## ================================================================================
## Total params: 303
## Trainable params: 303
## Non-trainable params: 0
## ________________________________________________________________________________
```

```
model2 %>% compile(loss = "mean_squared_error",
                   optimizer = "adam",
                   metric = "mean_absolute_error")

trained_model2 <- model2 %>% fit(
  x = X_train1, # sequence we're using for prediction
  y = y_train1, # sequence we're predicting
  batch_size = 18, # how many samples to pass to our model at a time
  epochs = 60, # how many times we'll look @ the whole dataset
  validation_split = 0.2) # how much data to hold out for testing as we go along

mse[2] = (model2 %>% evaluate(X_test1, y_test1))[1]

#Escala

denorm <- function(x, max, min) {
  return (x*(max - min)+min)
}

max <- apply(Alajuela1,2,max)
min <- apply(Alajuela1,2,min)

#Gráfico

pred = denorm(model2 %>% predict(X_train1),  max[length(Alajuela1)], min[length(Alajuela1)])
results = denorm(model2 %>% predict(X_test1), max[length(Alajuela1)], min[length(Alajuela1)])

var2 = c(rep(0,length(pred)), rep(1, length(results)))

pred = rbind(pred, results)


data1 = as.data.frame(cbind(pred, Alajuela1$RR))
names(data1) = c("fit", "RR")

MSE[2] = metricas (data1)


data2 = as.data.frame(cbind(results, Alajuela1$RR[197:235]))
names(data2) = c("fit", "RR")

MSE_results[2] = metricas(data2)

Fecha = paste(Alajuela1$Year, Alajuela1$Month)

everyother1 <- function(x) x[(seq_along(Fecha) + 5)%%12 == 6]
everyother2 <- function(x) x[(seq_along(Fecha))%%12 == 1]
```

```r
p1 <- ggplot(data1, aes(x = Fecha, y = RR, group = 1)) + geom_line(colour = "blue") +
  geom_line( aes(x = Fecha, y = fit, colour = (var2>0)))+
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
  panel.background = element_blank(), axis.text.x = element_text(angle = 45), legend.position = "none" )
  scale_x_discrete(breaks = everyother1) + labs (x = "Fecha", y = "Riesgo Relativo")

p2 <- ggplot(data2, aes(x = Fecha[197:235], y = RR, group = 1)) + geom_line(colour = "blue") +
  geom_line( aes(x = Fecha[197:235], y = fit, colour = "red"))+
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
  panel.background = element_blank(), axis.text.x = element_text(angle = 45), legend.position = "none" )
  scale_x_discrete(breaks = everyother2) + labs (x = "Fecha", y = "Riesgo Relativo")


print(p1)
```




```r
print(p2)
```



En este modelo se observa una reducción del error cuadrado medio.

Modelo con lag:

MODELO 3

NN creada con las nuevas variables lag, se ajusta el dropout, y unidades a lo que generó mejores resultados.

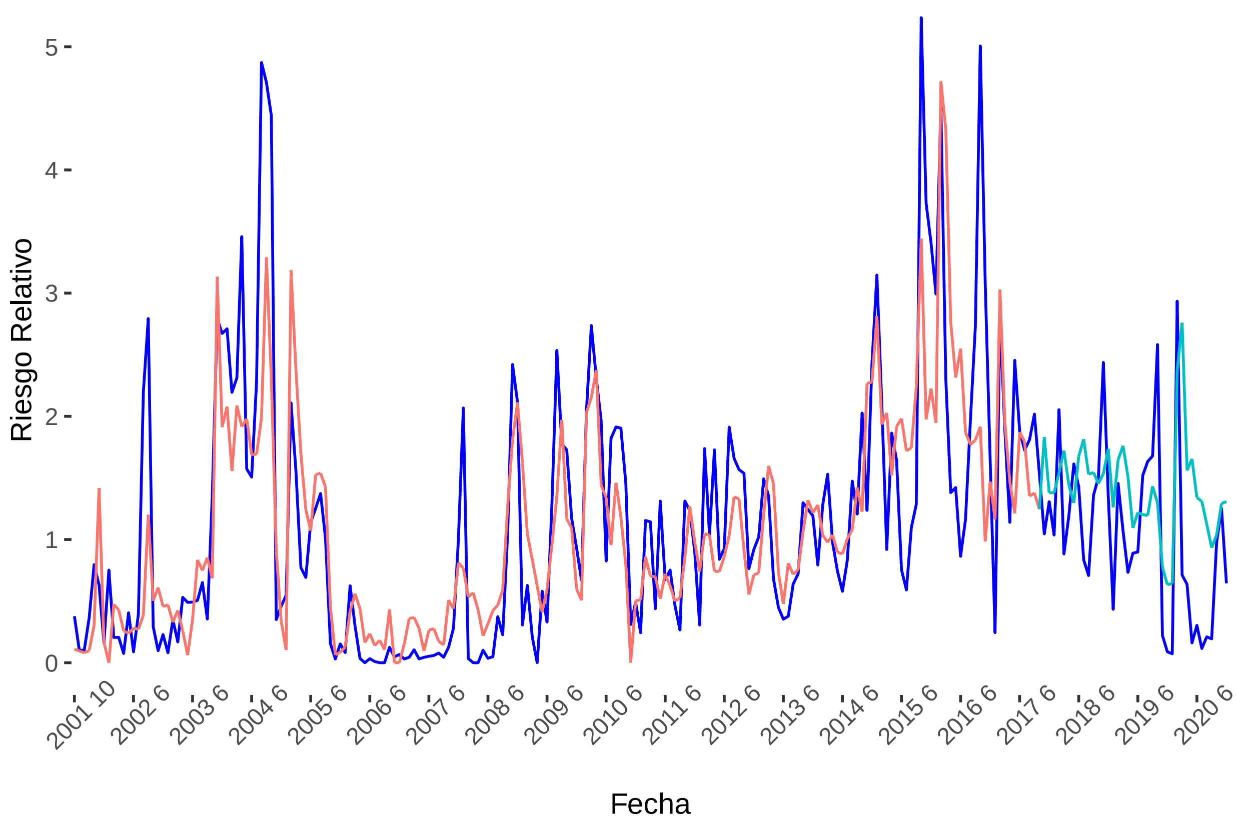
```
set.seed(123)
model3 <- keras_model_sequential()
# our input layer
model3 %>%
  layer_dense(input_shape = ncol(X_train), units = 32) %>%
  layer_dropout(rate = 0.2)%>%
  layer_dense(units = 16, activation = "relu")%>%
  layer_dense(units = 1, activation = "relu")


# look at our model architecture
summary(model3)
```

```
## Model: "sequential_2"
## ________________________________________________________________________________
##  Layer (type)                       Output Shape                    Param #
## ================================================================================
```

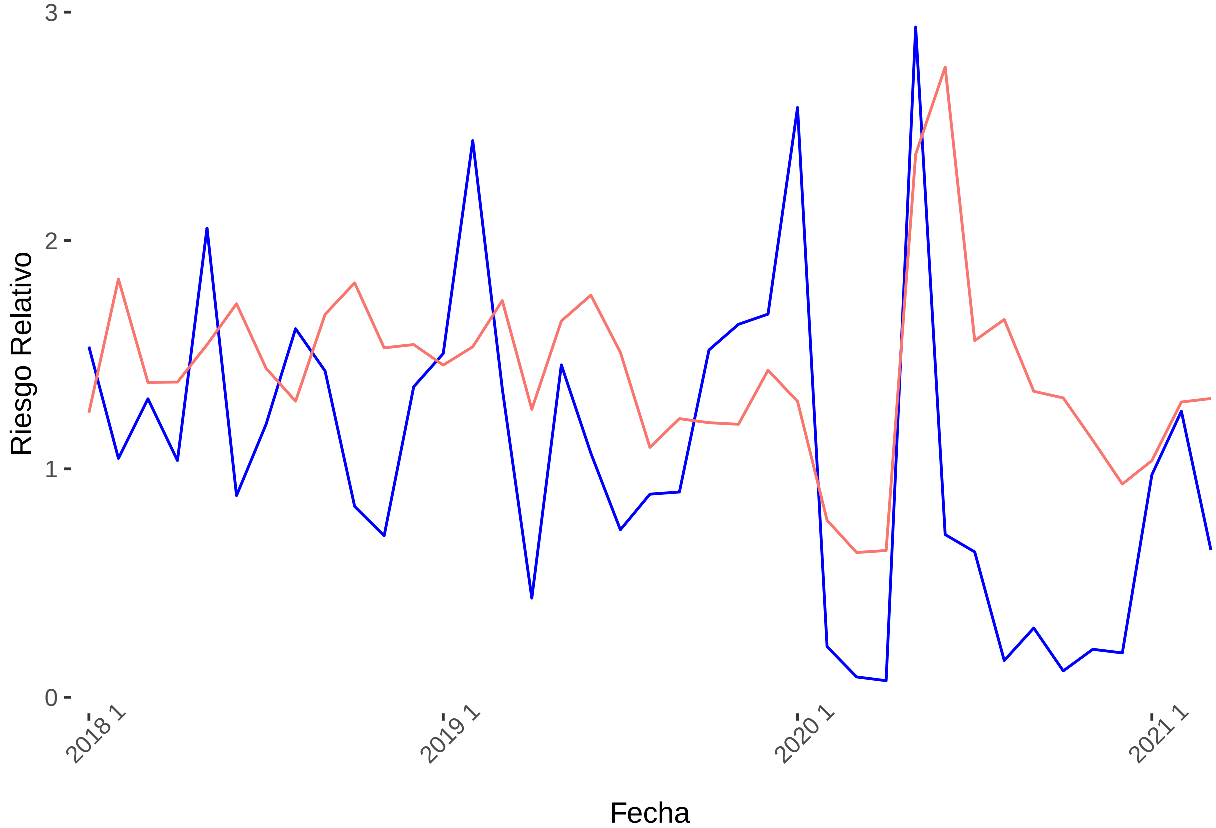
```
##  dense_7 (Dense)                    (None, 32)                    1056
##
##  dropout (Dropout)                  (None, 32)                    0
##
##  dense_6 (Dense)                    (None, 16)                    528
##
##  dense_5 (Dense)                    (None, 1)                     17
##
## ================================================================================
## Total params: 1,601
## Trainable params: 1,601
## Non-trainable params: 0
## ________________________________________________________________________________
```

```
model3 %>% compile(loss = "mean_squared_error",
                  optimizer = "adam",
                  metric = "mean_absolute_error")

trained_model3 <- model3 %>% fit(
  x = X_train, # sequence we're using for prediction
  y = y_train, # sequence we're predicting
  batch_size = 18, # how many samples to pass to our model at a time
  epochs = 50, # how many times we'll look @ the whole dataset
  validation_split = 0.2) # how much data to hold out for testing as we go along

mse[3] = (model3 %>% evaluate(X_test, y_test))[1]

#Escala

denorm <- function(x, max, min) {
  return (x*(max - min)+min)
}

max <- apply(Alajuela,2,max)
min <- apply(Alajuela,2,min)

#Gráfico

pred = denorm(model3 %>% predict(X_train),  max[length(Alajuela)], min[length(Alajuela)])
results = denorm(model3 %>% predict(X_test), max[length(Alajuela)], min[length(Alajuela)])

var2 = c(rep(0,length(pred)), rep(1, length(results)))

pred = rbind(pred, results)


data1 = as.data.frame(cbind(pred, Alajuela1$RR))
names(data1) = c("fit", "RR")

MSE[3] = metricas (data1)


data2 = as.data.frame(cbind(results, Alajuela1$RR[197:235]))
names(data2) = c("fit", "RR")
```

```r
MSE_results[3] = metricas(data2)

Fecha = paste(Alajuela$Year, Alajuela$Month)

everyother1 <- function(x) x[(seq_along(Fecha) + 5)%%12 == 6]
everyother2 <- function(x) x[(seq_along(Fecha))%%12 == 1]

p1 <- ggplot(data1, aes(x = Fecha, y = RR, group = 1)) + geom_line(colour = "blue") +
  geom_line( aes(x = Fecha, y = fit, colour = (var2>0)))+
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
  panel.background = element_blank(), axis.text.x = element_text(angle = 45), legend.position = "none" )
  scale_x_discrete(breaks = everyother1) + labs (x = "Fecha", y = "Riesgo Relativo")

p2 <- ggplot(data2, aes(x = Fecha[197:235], y = RR, group = 1)) + geom_line(colour = "blue") +
  geom_line( aes(x = Fecha[197:235], y = fit, colour = "red"))+
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
  panel.background = element_blank(), axis.text.x = element_text(angle = 45), legend.position = "none" )
  scale_x_discrete(breaks = everyother2) + labs (x = "Fecha", y = "Riesgo Relativo")


print(p1)
```




```r
print(p2)
```



Se construye un modelo con rnn:

MODELO 4

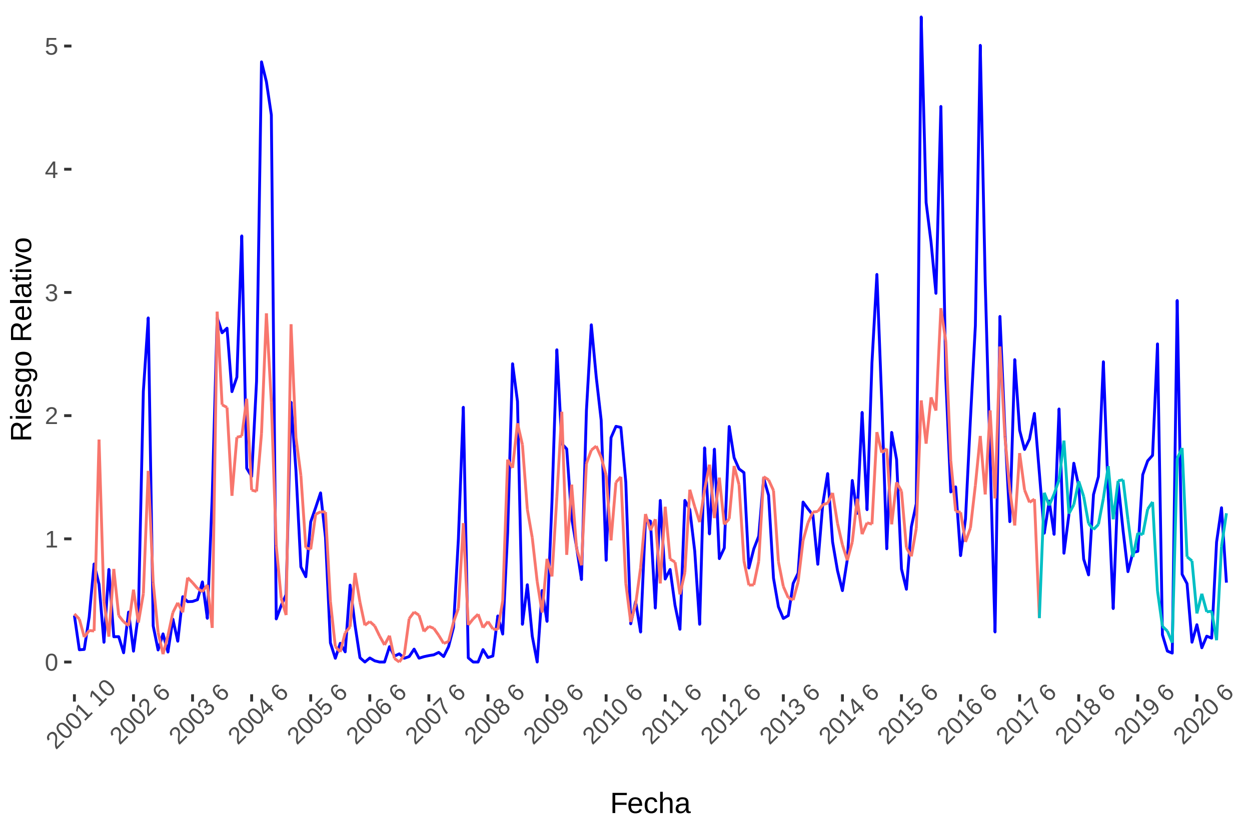
```
set.seed(123)
model4 <- keras_model_sequential()
# our input layer
model4 %>%
  layer_simple_rnn(units = 24, input_shape = c(ncol(X_train),1), activation='relu') %>%
  layer_dropout(rate = 0.4)%>%
  layer_dense(units = 12, activation = "relu")%>%
  layer_dense(units = 1, activation = "relu")


# look at our model architecture
summary(model4)
```

```
## Model: "sequential_3"
## ________________________________________________________________________________
##  Layer (type)                       Output Shape                      Param #
## ================================================================================
##  simple_rnn (SimpleRNN)             (None, 24)                        624
##
##  dropout_1 (Dropout)                (None, 24)                        0
##
```

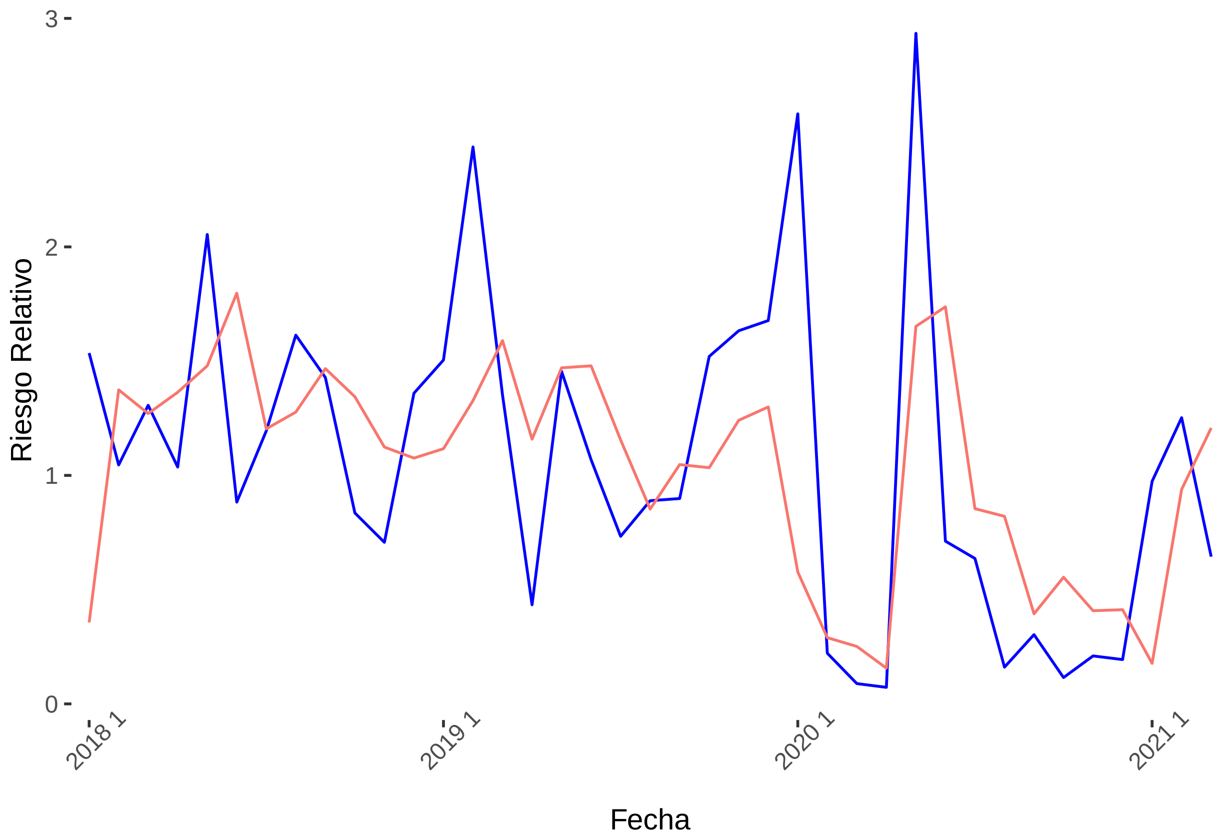
```
##  dense_9 (Dense)                    (None, 12)                    300
##
##  dense_8 (Dense)                    (None, 1)                     13
##
## ================================================================================
## Total params: 937
## Trainable params: 937
## Non-trainable params: 0
## ________________________________________________________________________________
```

```
model4 %>% compile(loss = "mean_squared_error",
                  optimizer = "adam",
                  metric = "mean_absolute_error")

trained_model4 <- model4 %>% fit(
  x = X_train, # sequence we're using for prediction
  y = y_train, # sequence we're predicting
  batch_size = 18, # how many samples to pass to our model at a time
  epochs = 50, # how many times we'll look @ the whole dataset
  validation_split = 0.2,
  shuffle = F) # how much data to hold out for testing as we go along

mse[4] = (model4 %>% evaluate(X_test, y_test))[1]

#Escala

denorm <- function(x, max, min) {
  return (x*(max - min)+min)
}

max <- apply(Alajuela,2,max)
min <- apply(Alajuela,2,min)

#Gráfico

pred = denorm(model4 %>% predict(X_train),  max[length(Alajuela)], min[length(Alajuela)])
results = denorm(model4 %>% predict(X_test), max[length(Alajuela)], min[length(Alajuela)])

var2 = c(rep(0,length(pred)), rep(1, length(results)))

pred = rbind(pred, results)


data1 = as.data.frame(cbind(pred, Alajuela1$RR))
names(data1) = c("fit", "RR")

MSE[4] = metricas (data1)


data2 = as.data.frame(cbind(results, Alajuela1$RR[197:235]))
names(data2) = c("fit", "RR")

MSE_results[4] = metricas(data2)
```

```
Fecha = paste(Alajuela$Year, Alajuela$Month)

everyother1 <- function(x) x[(seq_along(Fecha) + 5)%%12 == 6]
everyother2 <- function(x) x[(seq_along(Fecha))%%12 == 1]

p1 <- ggplot(data1, aes(x = Fecha, y = RR, group = 1)) + geom_line(colour = "blue") +
  geom_line( aes(x = Fecha, y = fit, colour = (var2>0)))+
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
  panel.background = element_blank(), axis.text.x = element_text(angle = 45), legend.position = "none" )
  scale_x_discrete(breaks = everyother1) + labs (x = "Fecha", y = "Riesgo Relativo")

p2 <- ggplot(data2, aes(x = Fecha[197:235], y = RR, group = 1)) + geom_line(colour = "blue") +
  geom_line( aes(x = Fecha[197:235], y = fit, colour = "red"))+
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
  panel.background = element_blank(), axis.text.x = element_text(angle = 45), legend.position = "none" )
  scale_x_discrete(breaks = everyother2) + labs (x = "Fecha", y = "Riesgo Relativo")


print(p1)
```




```
print(p2)
```



El modelo anterior ajusta muy bien; sin embargo, está basándose casi completamente en RR lag, es autoregresivo.

Modelos sin variable RRl1

MODELO 5

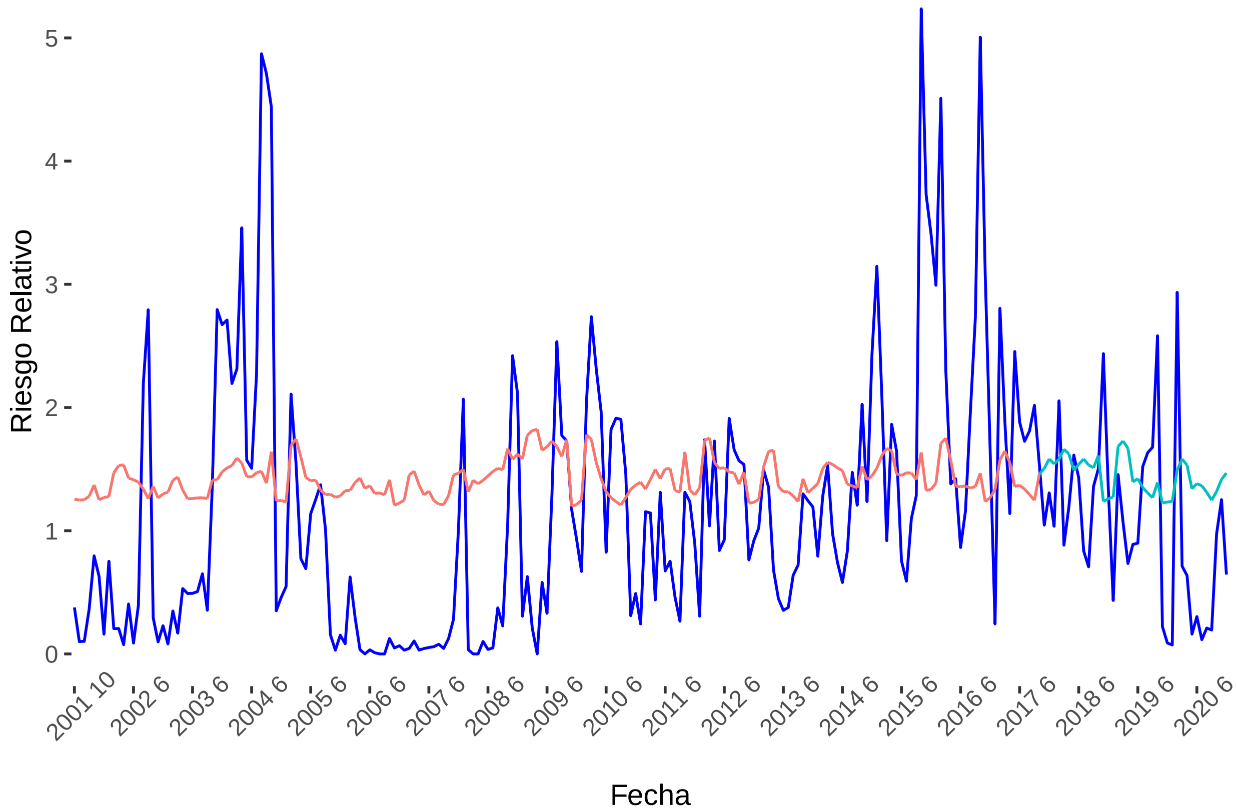
```
set.seed(123)
model5 <- keras_model_sequential()
# our input layer
model5 %>%
  layer_simple_rnn(units = 24, input_shape = c(ncol(X_train)-1,1), activation='tanh') %>%
  layer_dropout(rate = 0.4)%>%
  layer_dense(units = 12, activation = "relu")%>%
  layer_dense(units = 8, activation = "relu")%>%
  layer_dropout(rate = 0.4)%>%
  layer_dense(units = 1, activation = "sigmoid")


# look at our model architecture
summary(model5)
```

```
## Model: "sequential_4"
## ________________________________________________________________________________
##  Layer (type)                       Output Shape                      Param #
```

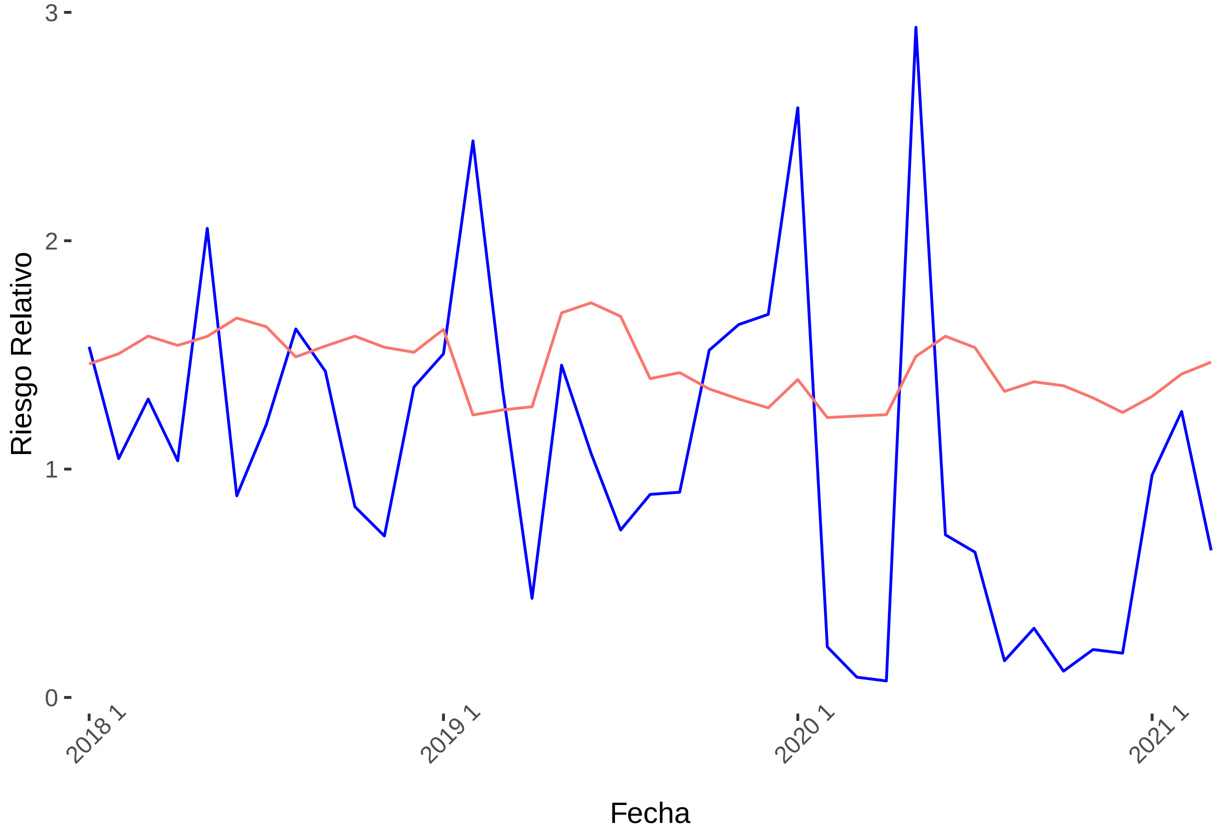
```
## ================================================================================
##  simple_rnn_1 (SimpleRNN)          (None, 24)                      624
##
##  dropout_3 (Dropout)               (None, 24)                      0
##
##  dense_12 (Dense)                  (None, 12)                      300
##
##  dense_11 (Dense)                  (None, 8)                       104
##
##  dropout_2 (Dropout)               (None, 8)                       0
##
##  dense_10 (Dense)                  (None, 1)                       9
##
## ================================================================================
## Total params: 1,037
## Trainable params: 1,037
## Non-trainable params: 0
## ________________________________________________________________________________
```

```
model5 %>% compile(loss = "mean_squared_error",
                  optimizer = "adam",
                  metric = "mean_absolute_error")

trained_model5 <- model5 %>% fit(
  x = X_train[,-32], # sequence we're using for prediction
  y = y_train, # sequence we're predicting
  batch_size = 18, # how many samples to pass to our model at a time
  epochs = 50, # how many times we'll look @ the whole dataset
  validation_split = 0.1,
  shuffle = F) # how much data to hold out for testing as we go along

mse[5] = (model5 %>% evaluate(X_test[,-32], y_test))[1]

#Escala

denorm <- function(x, max, min) {
  return (x*(max - min)+min)
}

max <- apply(Alajuela,2,max)
min <- apply(Alajuela,2,min)

#Gráfico

pred = denorm(model5 %>% predict(X_train[,-32]),  max[length(Alajuela)], min[length(Alajuela)])
results = denorm(model5 %>% predict(X_test[,-32]), max[length(Alajuela)], min[length(Alajuela)])

var2 = c(rep(0,length(pred)), rep(1, length(results)))

pred = rbind(pred, results)


data1 = as.data.frame(cbind(pred, Alajuela1$RR))
names(data1) = c("fit", "RR")
```

```
MSE[5] = metricas (data1)


data2 = as.data.frame(cbind(results, Alajuela1$RR[197:235]))
names(data2) = c("fit", "RR")

MSE_results[5] = metricas(data2)

Fecha = paste(Alajuela$Year, Alajuela$Month)

everyother1 <- function(x) x[(seq_along(Fecha) + 5)%%12 == 6]
everyother2 <- function(x) x[(seq_along(Fecha))%%12 == 1]

p1 <- ggplot(data1, aes(x = Fecha, y = RR, group = 1)) + geom_line(colour = "blue") +
  geom_line( aes(x = Fecha, y = fit, colour = (var2>0)))+
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
  panel.background = element_blank(), axis.text.x = element_text(angle = 45), legend.position = "none" )
  scale_x_discrete(breaks = everyother1) + labs (x = "Fecha", y = "Riesgo Relativo")

p2 <- ggplot(data2, aes(x = Fecha[197:235], y = RR, group = 1)) + geom_line(colour = "blue") +
  geom_line( aes(x = Fecha[197:235], y = fit, colour = "red"))+
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
  panel.background = element_blank(), axis.text.x = element_text(angle = 45), legend.position = "none" )
  scale_x_discrete(breaks = everyother2) + labs (x = "Fecha", y = "Riesgo Relativo")


print(p1)
```


Riesgo Relativo
5
4
3
2
1
0
2001 10
2002 6
2003 6
2004 6
2005 6
2006 6
2007 6
2008 6
2009 6
2010 6
2011 6
2012 6
2013 6
2014 6
2015 6
2016 6
2017 6
2018 6
2019 6
2020 6
Fecha

```
print(p2)
```



El ajuste no parece ser tan bueno como el modelo autoregresivo, pero no es un mal ajuste y no tiene autoregresión.

También se crea un modelo NN sin la variable de RRl1:

MODELO 6

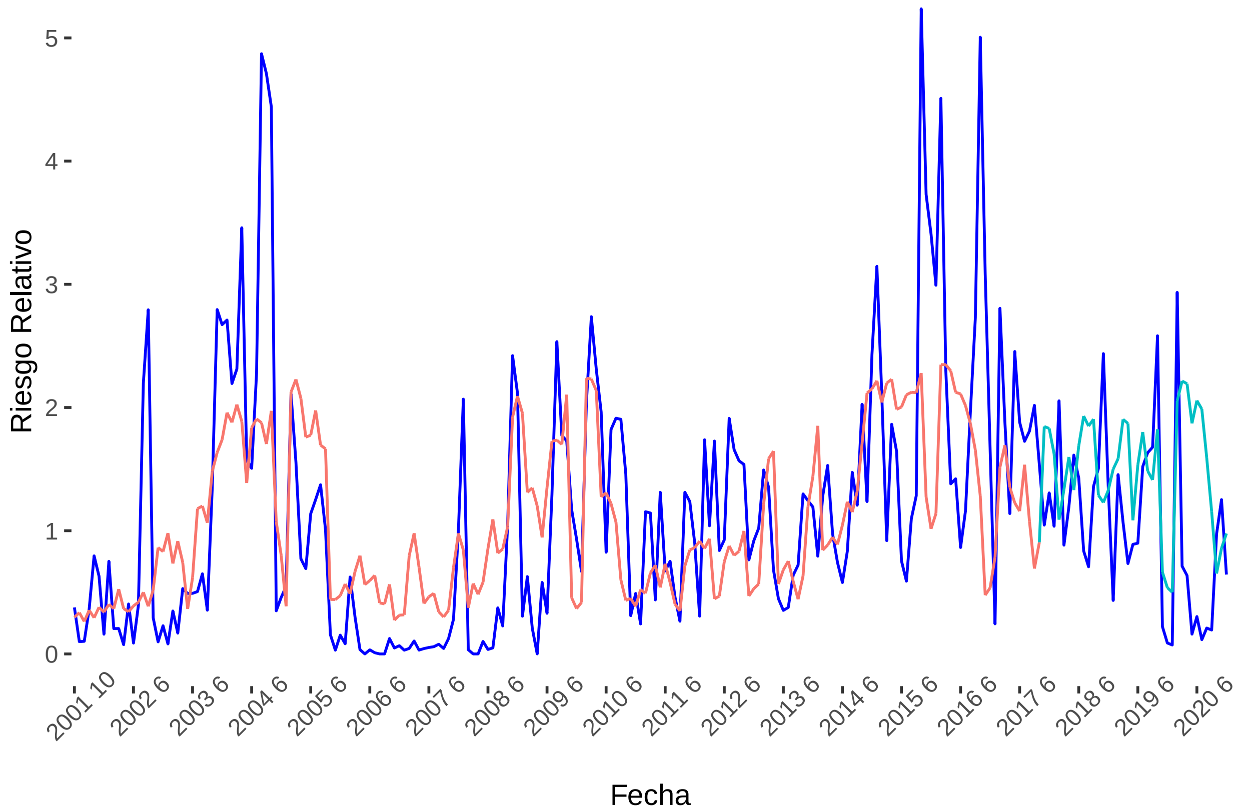
```
set.seed(123)
model6 <- keras_model_sequential()
# our input layer
model6 %>%
  layer_dense(input_shape = ncol(X_train)-1, units = 32) %>%
  layer_dense(units = 16, activation = "tanh")%>%
  layer_dense(units = 8, activation = "relu")%>%
  layer_dense(units = 4, activation = "relu")%>%
  layer_dropout(rate = 0.15)%>%
  layer_dense(units = 1, activation = "sigmoid")


# look at our model architecture
summary(model6)
```

```
## Model: "sequential_5"
## ________________________________________________________________________________
##  Layer (type)                       Output Shape                      Param #
```

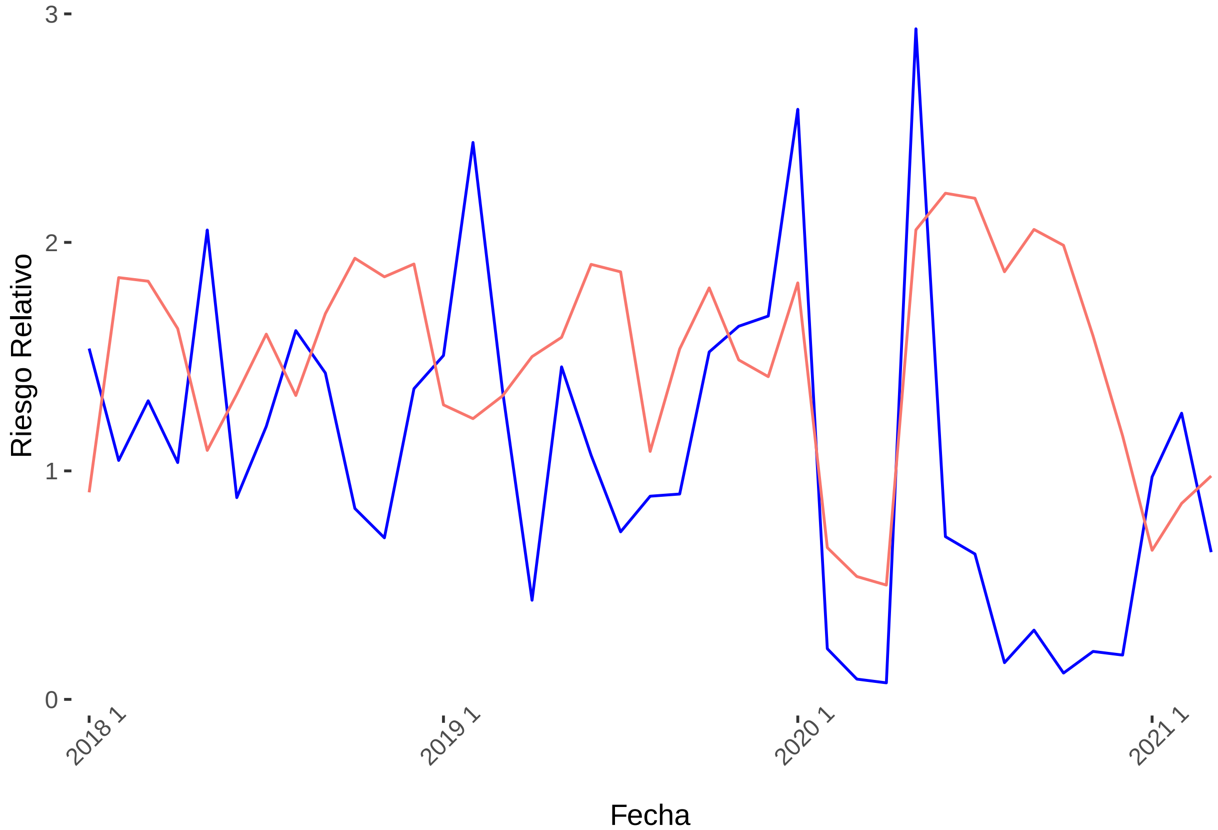
```
## ================================================================================
##  dense_17 (Dense)                  (None, 32)                     1024
##
##  dense_16 (Dense)                  (None, 16)                     528
##
##  dense_15 (Dense)                  (None, 8)                      136
##
##  dense_14 (Dense)                  (None, 4)                      36
##
##  dropout_4 (Dropout)               (None, 4)                      0
##
##  dense_13 (Dense)                  (None, 1)                      5
##
## ================================================================================
## Total params: 1,729
## Trainable params: 1,729
## Non-trainable params: 0
## ________________________________________________________________________________
```

```r
model6 %>% compile(loss = "mean_squared_error",
                  optimizer = "adam",
                  metric = "mean_absolute_error")

trained_model6 <- model6 %>% fit(
  x = X_train[,-32], # sequence we're using for prediction
  y = y_train, # sequence we're predicting
  batch_size = 18, # how many samples to pass to our model at a time
  epochs = 80, # how many times we'll look @ the whole dataset
  validation_split = 0.2) # how much data to hold out for testing as we go along

mse[6] = (model6 %>% evaluate(X_test[,-32], y_test))[1]

#Escala

denorm <- function(x, max, min) {
  return (x*(max - min)+min)
}

max <- apply(Alajuela,2,max)
min <- apply(Alajuela,2,min)

#Gráfico

pred = denorm(model6 %>% predict(X_train[,-32]),  max[length(Alajuela)], min[length(Alajuela)])
results = denorm(model6 %>% predict(X_test[,-32]), max[length(Alajuela)], min[length(Alajuela)])

var2 = c(rep(0,length(pred)), rep(1, length(results)))

pred = rbind(pred, results)


data1 = as.data.frame(cbind(pred, Alajuela1$RR))
names(data1) = c("fit", "RR")
```

```
MSE[6] = metricas (data1)


data2 = as.data.frame(cbind(results, Alajuela1$RR[197:235]))
names(data2) = c("fit", "RR")

MSE_results[6] = metricas(data2)

Fecha = paste(Alajuela$Year, Alajuela$Month)

everyother1 <- function(x) x[(seq_along(Fecha) + 5)%%12 == 6]
everyother2 <- function(x) x[(seq_along(Fecha))%%12 == 1]

p1 <- ggplot(data1, aes(x = Fecha, y = RR, group = 1)) + geom_line(colour = "blue") +
  geom_line( aes(x = Fecha, y = fit, colour = (var2>0)))+
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
  panel.background = element_blank(), axis.text.x = element_text(angle = 45), legend.position = "none" )
  scale_x_discrete(breaks = everyother1) + labs (x = "Fecha", y = "Riesgo Relativo")

p2 <- ggplot(data2, aes(x = Fecha[197:235], y = RR, group = 1)) + geom_line(colour = "blue") +
  geom_line( aes(x = Fecha[197:235], y = fit, colour = "red"))+
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
  panel.background = element_blank(), axis.text.x = element_text(angle = 45), legend.position = "none" )
  scale_x_discrete(breaks = everyother2) + labs (x = "Fecha", y = "Riesgo Relativo")


print(p1)
```


5
4
3
2
1
0
Riesgo Relativo
2001 10
2002 6
2003 6
2004 6
2005 6
2006 6
2007 6
2008 6
2009 6
2010 6
2011 6
2012 6
2013 6
2014 6
2015 6
2016 6
2017 6
2018 6
2019 6
2020 6
Fecha

```
print(p2)
```



Ahora se prueba utilizar la estructura de los modelos anteriores, pero con la variable RRl1

Nuevos modelos con RRl1

MODELO 7

Tiene la estructura del modelo 5

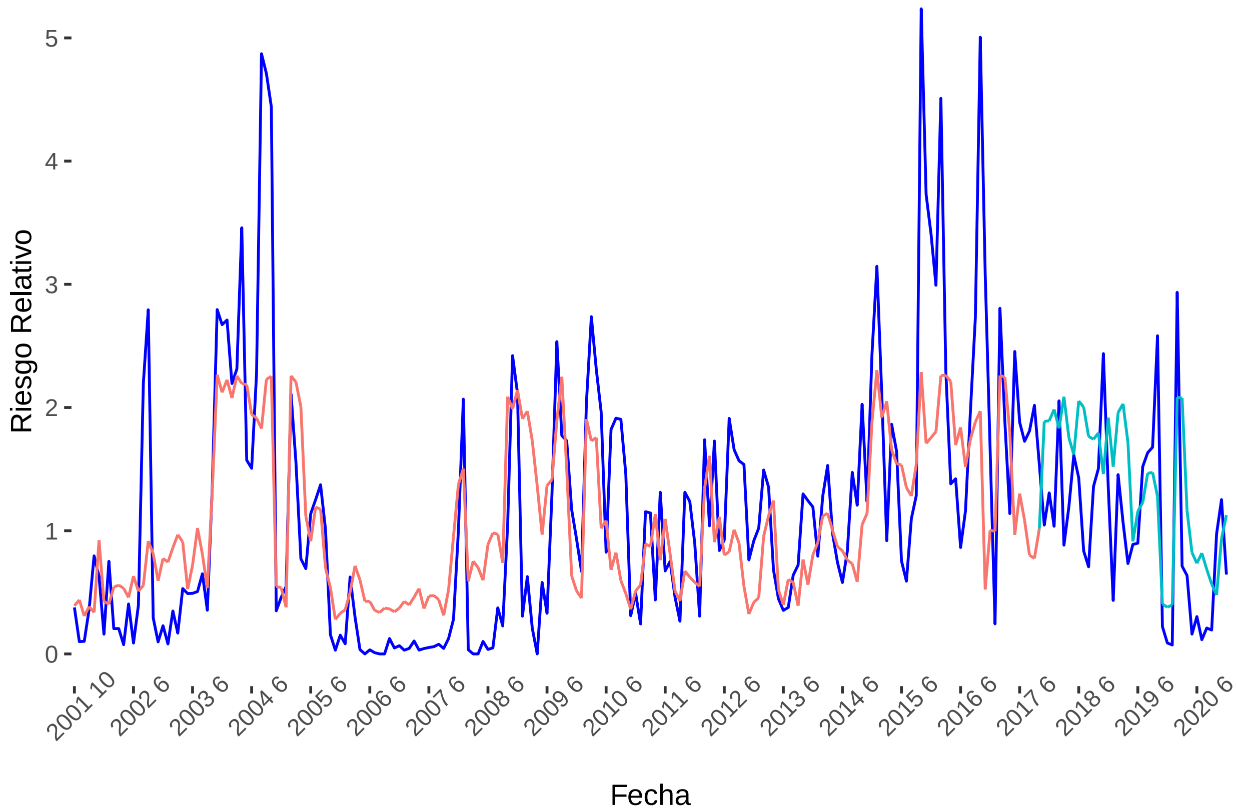
```
set.seed(123)
model7 <- keras_model_sequential()
# our input layer
model7 %>%
  layer_simple_rnn(units = 24, input_shape = c(ncol(X_train),1), activation='tanh') %>%
  layer_dropout(rate = 0.4)%>%
  layer_dense(units = 12, activation = "relu")%>%
  layer_dense(units = 8, activation = "relu")%>%
  layer_dropout(rate = 0.4)%>%
  layer_dense(units = 1, activation = "sigmoid")


# look at our model architecture
summary(model7)
```

```
## Model: "sequential_6"
## ________________________________________________________________________________
```

```
##  Layer (type)                       Output Shape                   Param #
## ================================================================================
##  simple_rnn_2 (SimpleRNN)           (None, 24)                     624
##
##  dropout_6 (Dropout)                (None, 24)                     0
##
##  dense_20 (Dense)                   (None, 12)                     300
##
##  dense_19 (Dense)                   (None, 8)                      104
##
##  dropout_5 (Dropout)                (None, 8)                      0
##
##  dense_18 (Dense)                   (None, 1)                      9
##
## ================================================================================
## Total params: 1,037
## Trainable params: 1,037
## Non-trainable params: 0
## ________________________________________________________________________________
```

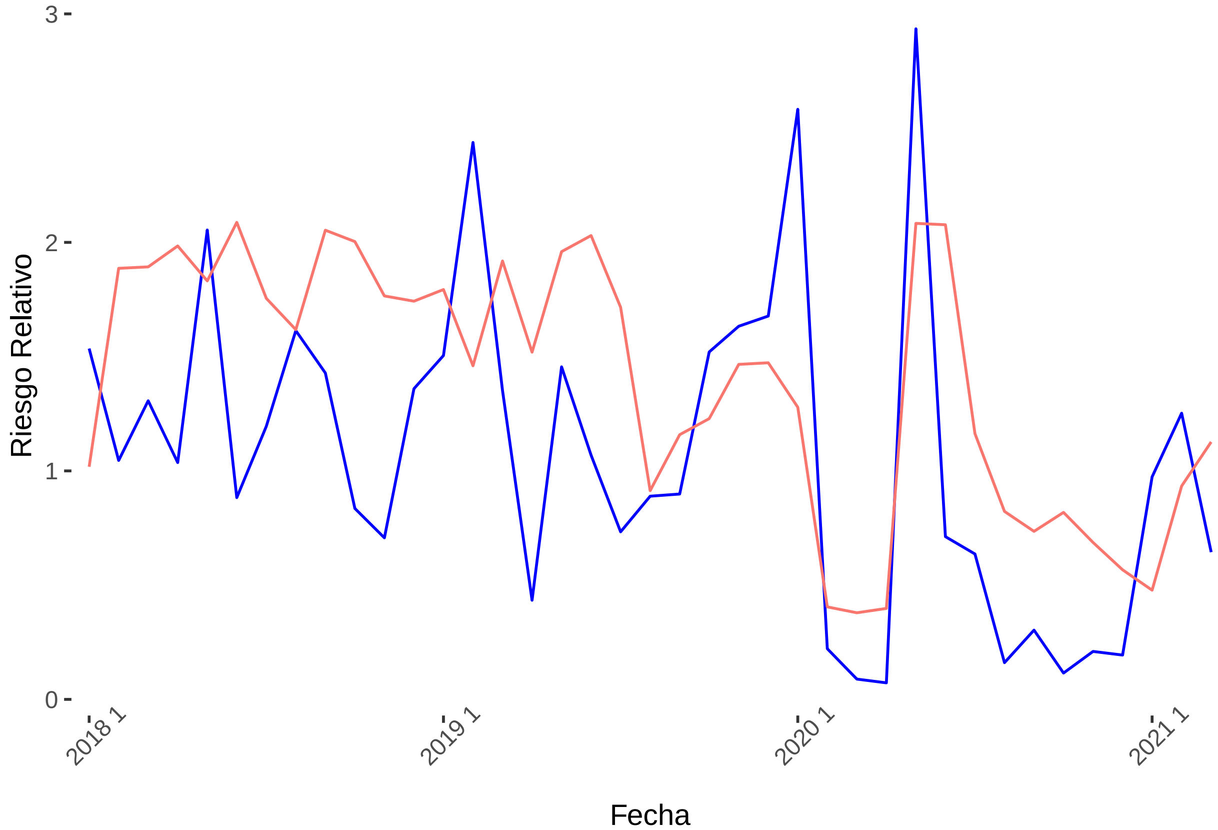
```
model7 %>% compile(loss = "mean_squared_error",
                  optimizer = "adam",
                  metric = "mean_absolute_error")

trained_model7 <- model7 %>% fit(
  x = X_train, # sequence we're using for prediction
  y = y_train, # sequence we're predicting
  batch_size = 18, # how many samples to pass to our model at a time
  epochs = 80, # how many times we'll look @ the whole dataset
  validation_split = 0.1,
  shuffle = F) # how much data to hold out for testing as we go along

mse[7] = (model7 %>% evaluate(X_test, y_test))[1]

#Escala

denorm <- function(x, max, min) {
  return (x*(max - min)+min)
}

max <- apply(Alajuela,2,max)
min <- apply(Alajuela,2,min)

#Gráfico

pred = denorm(model7 %>% predict(X_train),  max[length(Alajuela)], min[length(Alajuela)])
results = denorm(model7 %>% predict(X_test), max[length(Alajuela)], min[length(Alajuela)])

var2 = c(rep(0,length(pred)), rep(1, length(results)))

pred = rbind(pred, results)


data1 = as.data.frame(cbind(pred, Alajuela1$RR))
```

```r
names(data1) = c("fit", "RR")

MSE[7] = metricas (data1)


data2 = as.data.frame(cbind(results, Alajuela1$RR[197:235]))
names(data2) = c("fit", "RR")

MSE_results[7] = metricas(data2)

Fecha = paste(Alajuela$Year, Alajuela$Month)

everyother1 <- function(x) x[(seq_along(Fecha) + 5)%%12 == 6]
everyother2 <- function(x) x[(seq_along(Fecha))%%12 == 1]

p1 <- ggplot(data1, aes(x = Fecha, y = RR, group = 1)) + geom_line(colour = "blue") +
  geom_line( aes(x = Fecha, y = fit, colour = (var2>0)))+
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
  panel.background = element_blank(), axis.text.x = element_text(angle = 45), legend.position = "none" )
  scale_x_discrete(breaks = everyother1) + labs (x = "Fecha", y = "Riesgo Relativo")

p2 <- ggplot(data2, aes(x = Fecha[197:235], y = RR, group = 1)) + geom_line(colour = "blue") +
  geom_line( aes(x = Fecha[197:235], y = fit, colour = "red"))+
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
  panel.background = element_blank(), axis.text.x = element_text(angle = 45), legend.position = "none" )
  scale_x_discrete(breaks = everyother2) + labs (x = "Fecha", y = "Riesgo Relativo")


print(p1)
```



Riesgo Relativo
5
4
3
2
1
0
2001 10
2002 6
2003 6
2004 6
2005 6
2006 6
2007 6
2008 6
2009 6
2010 6
2011 6
2012 6
2013 6
2014 6
2015 6
2016 6
2017 6
2018 6
2019 6
2020 6
Fecha


```
print(p2)
```



MODELO 8

Tiene la estructura del modelo 6

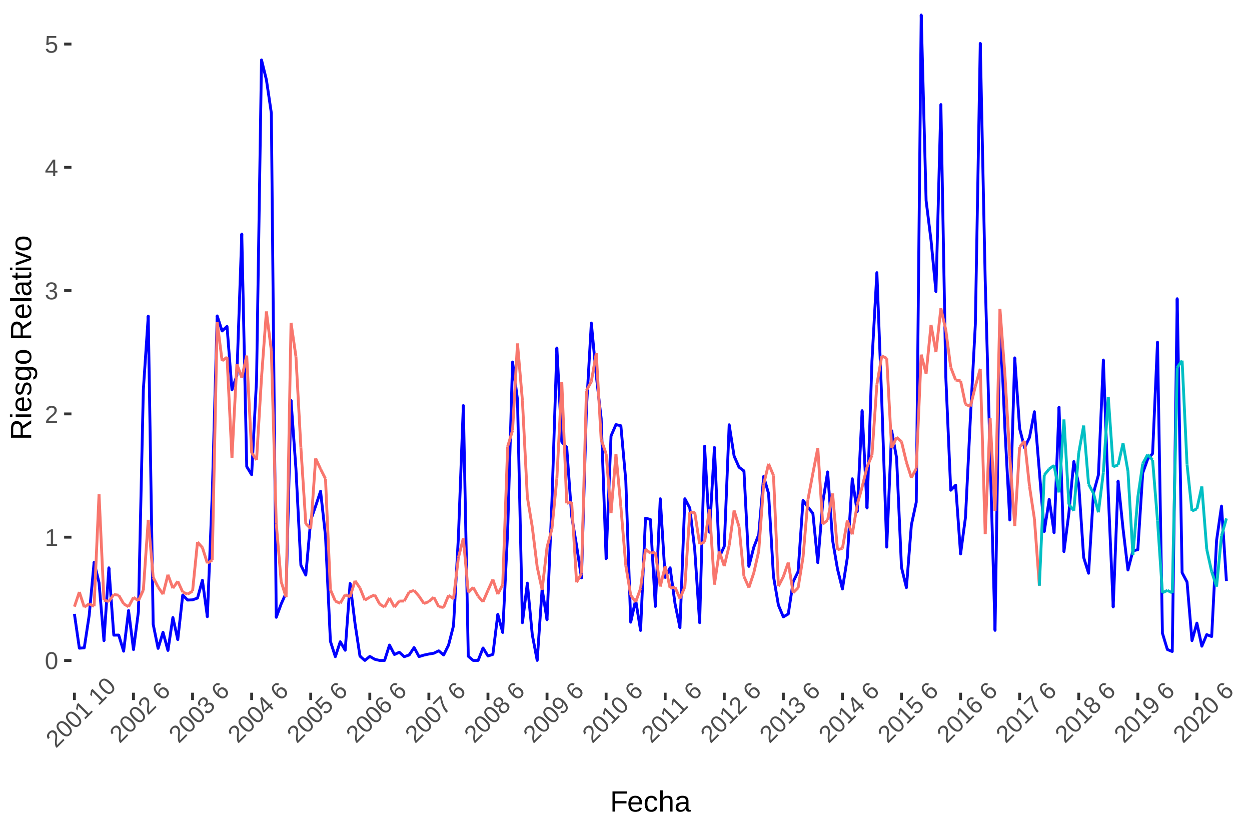
```
set.seed(123)
model8 <- keras_model_sequential()
# our input layer
model8 %>%
  layer_dense(input_shape = ncol(X_train), units = 32) %>%
  layer_dense(units = 16, activation = "tanh")%>%
  layer_dense(units = 8, activation = "relu")%>%
  layer_dense(units = 4, activation = "relu")%>%
  layer_dropout(rate = 0.15)%>%
  layer_dense(units = 1, activation = "sigmoid")


# look at our model architecture
summary(model8)
```

```
## Model: "sequential_7"
## ________________________________________________________________________________
##  Layer (type)                       Output Shape                    Param #
## ================================================================================
##  dense_25 (Dense)                   (None, 32)                      1056
##
##  dense_24 (Dense)                   (None, 16)                      528
```

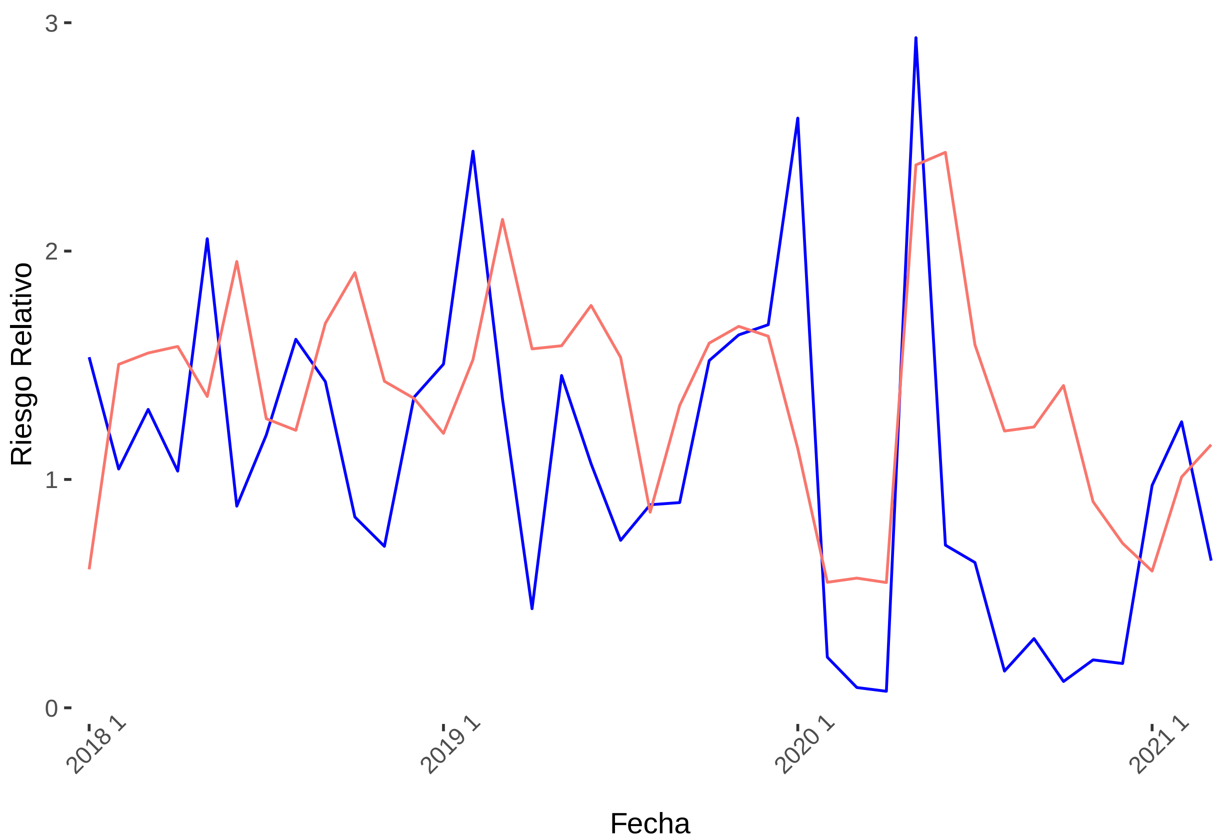
```
##
##  dense_23 (Dense)                  (None, 8)                       136
##
##  dense_22 (Dense)                  (None, 4)                       36
##
##  dropout_7 (Dropout)               (None, 4)                       0
##
##  dense_21 (Dense)                  (None, 1)                       5
##
## ================================================================================
## Total params: 1,761
## Trainable params: 1,761
## Non-trainable params: 0
## ________________________________________________________________________________
```

```
model8 %>% compile(loss = "mean_squared_error",
                  optimizer = "adam",
                  metric = "mean_absolute_error")

trained_model8 <- model8 %>% fit(
  x = X_train, # sequence we're using for prediction
  y = y_train, # sequence we're predicting
  batch_size = 18, # how many samples to pass to our model at a time
  epochs = 60, # how many times we'll look @ the whole dataset
  validation_split = 0.2) # how much data to hold out for testing as we go along

mse[8] = (model8 %>% evaluate(X_test, y_test))[1]

#Escala

denorm <- function(x, max, min) {
  return (x*(max - min)+min)
}

max <- apply(Alajuela,2,max)
min <- apply(Alajuela,2,min)

#Gráfico

pred = denorm(model8 %>% predict(X_train),  max[length(Alajuela)], min[length(Alajuela)])
results = denorm(model8 %>% predict(X_test), max[length(Alajuela)], min[length(Alajuela)])

var2 = c(rep(0,length(pred)), rep(1, length(results)))

pred = rbind(pred, results)


data1 = as.data.frame(cbind(pred, Alajuela1$RR))
names(data1) = c("fit", "RR")

MSE[8] = metricas (data1)


data2 = as.data.frame(cbind(results, Alajuela1$RR[197:235]))
```

```
names(data2) = c("fit", "RR")

MSE_results[8] = metricas(data2)

Fecha = paste(Alajuela$Year, Alajuela$Month)

everyother1 <- function(x) x[(seq_along(Fecha) + 5)%%12 == 6]
everyother2 <- function(x) x[(seq_along(Fecha))%%12 == 1]

p1 <- ggplot(data1, aes(x = Fecha, y = RR, group = 1)) + geom_line(colour = "blue") +
  geom_line( aes(x = Fecha, y = fit, colour = (var2>0)))+
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
  panel.background = element_blank(), axis.text.x = element_text(angle = 45), legend.position = "none" )
  scale_x_discrete(breaks = everyother1) + labs (x = "Fecha", y = "Riesgo Relativo")

p2 <- ggplot(data2, aes(x = Fecha[197:235], y = RR, group = 1)) + geom_line(colour = "blue") +
  geom_line( aes(x = Fecha[197:235], y = fit, colour = "red"))+
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
  panel.background = element_blank(), axis.text.x = element_text(angle = 45), legend.position = "none" )
  scale_x_discrete(breaks = everyother2) + labs (x = "Fecha", y = "Riesgo Relativo")


print(p1)
```



```
print(p2)
```



Comparación de modelos

```
evaluacion = cbind(mse,MSE,MSE_results)
rownames(evaluacion) = c("Modelo 1", "Modelo 2", "Modelo 3", "Modelo 4", "Modelo 5", "Modelo 6",
                "Modelo 7", "Modelo 8")
colnames(evaluacion) = c("mse keras", "NRMSE total", "NRMSE predicciones")
print(evaluacion)
```

```
##            mse keras  NRMSE total NRMSE predicciones
## Modelo 1 0.02238605 0.8192187   0.5738745
## Modelo 2 0.01418334 0.8699556   0.3635951
## Modelo 3 0.02006792 0.4377564   0.5144484
## Modelo 4 0.01388712 0.4330252   0.3560015
## Modelo 5 0.02170859 0.9401779   0.5565077
## Modelo 6 0.02851121 0.6993805   0.7308952
## Modelo 7 0.01759642 0.542851    0.4510906
## Modelo 8 0.01947341 0.4406103   0.499208
```

Teniendo las métricas de NRMSE podemos ver que los mejores 3 modelos son: Modelo 3, Modelo 4 y Modelo 7. El mse calculado por keras es solo para datos de training y está utilizando datos estandarizados previamente.