

# Proyecto Final Modulo IV

José de Jesús Rodríguez Barreto

20/05/2022

## Introducción

Las redes neuronales son un método de cálculo que busca imitar el funcionamiento de las neuronas en un organismo, con unidades conectadas entre si que generan y refuerzan conceptos para llegar a conclusiones, en este proyecto se busca usar las cualidades de las redes neuronales para hacer un modelo de predicción sobre la precipitación en una zona determinada de la republica.

## Trabajando con los datos

Nuestra base de datos original es una con 7 variables las cuales son fecha, precipitación, temperatura media, temperatura máxima, temperatura minima, evaporación y municipio, en las fechas tenemos fechas desde el siglo pasado las cuales no son relevantes para nuestro estudio por lo que hay que desecharlas como se hace en la siguiente linea de código:

```
library(tidyverse)
```

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

```
## v ggplot2 3.3.5      v purrr   0.3.4  
## v tibble  3.1.6      v dplyr  1.0.7  
## v tidyr   1.1.4      v stringr 1.4.0  
## v readr   2.1.1      v forcats 0.5.1
```

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

```
library(skimr)  
library(naniar)
```

```
## Warning: package 'naniar' was built under R version 4.1.3
```

```
##  
## Attaching package: 'naniar'
```

```
## The following object is masked from 'package:skimr':  
##  
## n_complete
```

```
library(visdat)
```

```
## Warning: package 'visdat' was built under R version 4.1.3
```

```
library(simputation)
```

```
## Warning: package 'simputation' was built under R version 4.1.3
```

```
##  
## Attaching package: 'simputation'
```

```
## The following object is masked from 'package:naniar':  
##  
##      impute_median
```

```
library(readxl)  
BasePrec<-read_excel("BasePrec.xlsx",col_types = c("date",  
  "numeric", "numeric", "numeric", "numeric",  
  "numeric", "text"))  
BasePrec<-BasePrec[BasePrec$fecha>="2020-01-01",]
```

Con esto dejamos solo las fechas a partir del 2020 reduciendo la base de datos de 120 mil entradas a solo 4285 pero el siguiente problema a resolver es el de los datos faltantes ya que nuestra base de datos cuenta con varios como se puede ver a continuación:

```
apply(is.na(BasePrec), 2, sum)
```

```
##      fecha precipitacion    temp.media    temp.max    temp.min  
##      0          658          732          732          732  
##  evaporacion    municipio  
##      1336          0
```

Para poder resolver como imputar los datos faltantes analizamos los datos que tenemos

```
skim(BasePrec)
```

#### Data summary






Name	BasePrec
Number of rows	4285
Number of columns	7
Column type frequency:	
character	1

numeric	5
POSIXct	1
<hr/>	
Group variables	None

#### Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
municipio	0	1	7	17	0	5	0

#### Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
precipitacion	658	0.85	1.44	5.66	0.00	0.0	0.00	0.00	108.00	
temp.media	732	0.83	17.06	3.58	2.25	14.0	17.00	20.00	27.50	
temp.max	732	0.83	25.79	4.02	12.00	23.0	26.00	29.00	38.00	
temp.min	732	0.83	8.45	4.37	0.00	5.0	8.00	12.00	17.50	
evaporacion	1336	0.69	4.57	1.86	0.00	3.2	4.47	5.87	13.66	

#### Variable type: POSIXct

skim_variable	n_missing	complete_rate	min	max	median	n_unique
fecha	0	1	2020-01-02	2022-05-07	2021-03-05	857

Como los datos estan algo sesgados entontces usaremos la mediana en lugar de la media para completar los datos faltantes usando las siguientes lineas de codigo:

```
nn<-impute_median_all(BasePrec)
apply(is.na(BasePrec), 2, sum) #comprobamos que ya no existan valores faltantes
```

```
##      fecha precipitacion  temp.media  temp.max  temp.min
##      0          658          732          732          732
##  evaporacion  municipio
##      1336          0
```

Ya que imputamos los datos, entonces es tiempo de reducir las fechas a 4 valores que serán las estaciones del año, primavera, verano, otoño e invierno:

```
Fecha<-if_else(nn$fecha<"2020-03-19", "Invierno",
              if_else(nn$fecha<"2020-06-20", "Primavera",
                    if_else(nn$fecha<"2020-09-22", "Verano",
                          if_else(nn$fecha<"2020-12-21", "Otoño",
                                if_else(nn$fecha<"2021-03-20", "Invierno",
                                      if_else(nn$fecha<"2021-06-20", "Primavera",
                                            if_else(nn$fecha<"2021-09-22", "Verano",
```

```

ano",
                                                                    if_else(nn$fecha<"2021-12-
21","Otoño",
                                                                    if_else(nn$fecha<
"2022-03-20","Invierno","Primavera"))))))))
nn$fecha<-Fecha

```

Ahora convertimos a factores las variables que contienen texto y posteriormente todas las unidades a numéricas lo que hará que los factores tomen valores entre 1 y 4 en las fechas y entre 1 y 5 en los municipios quedando los valores de la siguiente forma: en fechas invierno=1, otoño=2, primavera=3, verano=4, en los municipios Actopan=1, Cuautitlán Izcalli=2, Ixmiquilpan=3, Jilotepec=4, Tepetitlán=5. Ahora solo queda escalar los datos de manera que el mínimo sea 0 y el máximo 1 y con esto nuestra base de datos esta lista para entrar en la red neuronal.

```

#Para escalar usaremos la siguiente función
min.max<-function(v)
{
  return((v-min(v))/(max(v)-min(v)))
}

```

```

nn<-mutate_at(nn,c("fecha","municipio"),as.factor)
nn<-mutate_at(nn,c("fecha","municipio"),as.numeric)
max.prec<-max(nn$precipitacion)
min.prec<-min(nn$precipitacion) #estas variables las usaremos para reescalar la variable a su valor original
nn<-mutate_all(nn,min.max)
str(nn)

```

```

## tibble [4,285 x 7] (S3: tbl_df/tbl/data.frame)
## $ fecha      : num [1:4285] 0 0 0 0 0 0 0 0 0 0 ...
## $ precipitacion: num [1:4285] 0 0 0 0 0 0 0 0 0 0 ...
## $ temp.media  : num [1:4285] 0.624 0.564 0.485 0.228 0.366 ...
## $ temp.max    : num [1:4285] 0.692 0.5 0.423 0.192 0.462 ...
## $ temp.min    : num [1:4285] 0.3429 0.4571 0.3429 0.0571 0.0571 ...
## $ evaporacion : num [1:4285] 0.275 0.416 0.207 0.106 0.193 ...
## $ municipio   : num [1:4285] 0 0 0 0 0 0 0 0 0 0 ...

```

## Red Neuronal

Para nuestra red neuronal usamos una librería de r llamada neural net la cual nos da un toolbox al cual ingresar nuestros datos arrojándonos el modelo por si sola.

```

library(neuralnet)

```

```

##
## Attaching package: 'neuralnet'

```

```

## The following object is masked from 'package:dplyr':
##

```

```
##      compute
```

Antes de meter los datos en la toolbox separamos nuestra base de datos en dos sets uno de entrenamiento y uno de prueba, en el de entrenamiento pondremos el 80% de nuestras entradas y en el de prueba el 20% restante

```
N<-dim(nn) [1]

id.train<-sample(1:N,0.8*N)
id.test<-setdiff(1:N,id.train)

set.train<-nn[id.train,]
set.test<-nn[id.test,]
```

Ahora ingresamos el set de entrenamiento en la toolbox para esto tenemos que elegir que clase de modelo preferimos, en nuestro caso usaremos precipitacion~. El cual nos dice que nuestra resultante es igual a la suma de todas las variables, como una combinación lineal, ahora elegimos el numero de capas ocultas en nuestro caso se eligió 7,6 y5

```
precipitacion.modelo<-neuralnet(precipitacion ~., set.train,
                                hidden = c(7,6,5),
                                err.fct = "sse",
                                linear.output = TRUE
)
```

Ahora lo que sigue es probar nuestra red neuronal con el set de prueba poniéndola a hacer predicciones con este

```
modelo_resultados<-predict(precipitacion.modelo,set.test)
pred<-modelo_resultados*(max.prec - min.prec) + min.prec
prec_test<-set.test$precipitacion*(max.prec - min.prec) + min.prec
head(pred)
```

```
##      [,1]
## [1,] 0.1771969
## [2,] 0.2328895
## [3,] 0.1615266
## [4,] 0.1862874
## [5,] 0.1980103
## [6,] 0.1970177
```

Ya hechas las predicciones, calculamos el índice de error que tienen estas, en nuestro caso al ser un problema de tipo regresión usamos 3 opciones, el error cuadrático medio, la raíz del error cuadrático medio y el error absoluto medio.

```
mean((pred-prec_test)^2)      # Este es el error cuadrático medio
```

```
## [1] 58.00474
```

```
(mean((pred-prec_test)^2))^(1/2)      # Esta es la raíz cuadrada del error cuadrático medio
```

```
## [1] 7.616084
```

```
mean(abs(pred-prec_test))
```

```
## [1] 2.197766
```

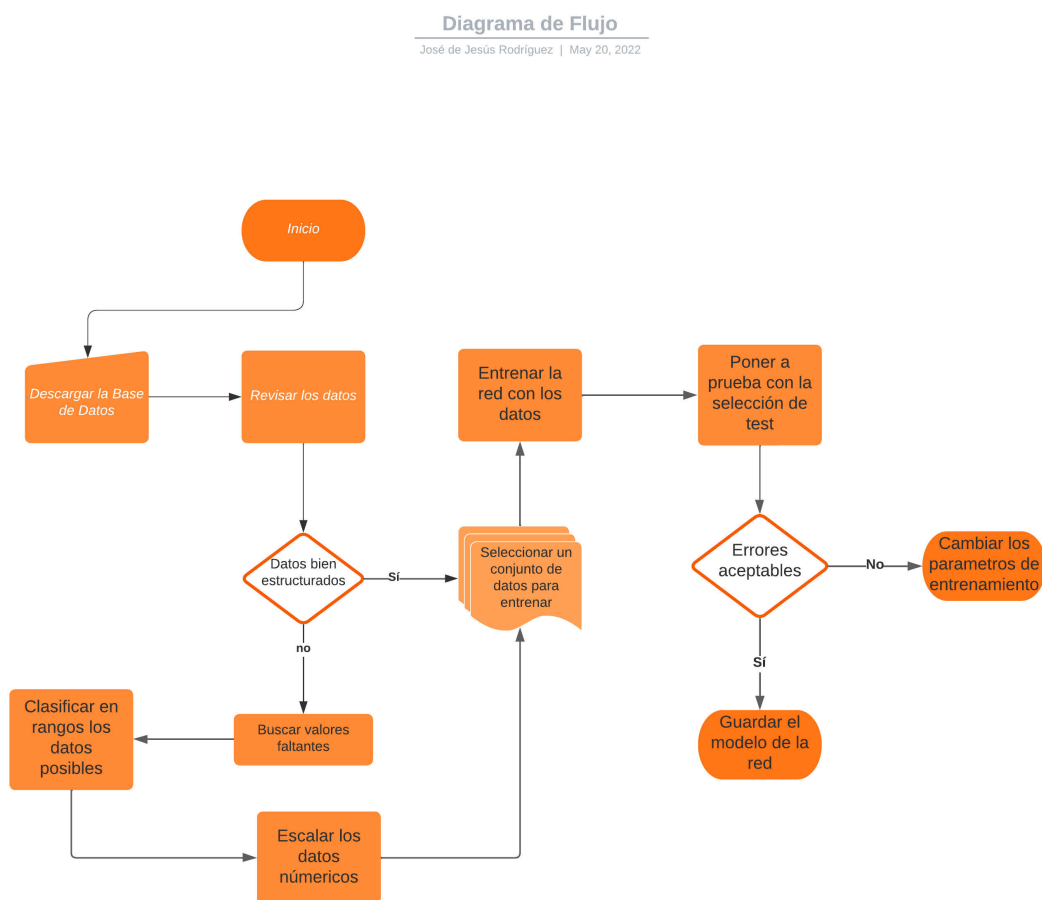
Con la medida de los errores podemos concluir si nuestro modelo nos es útil o hay que trabajar de nuevo en él, en nuestro caso, el modelo tiene un error en las precipitaciones bajo lo cual nos dice que se puede trabajar con este modelo.

## Conclusiones

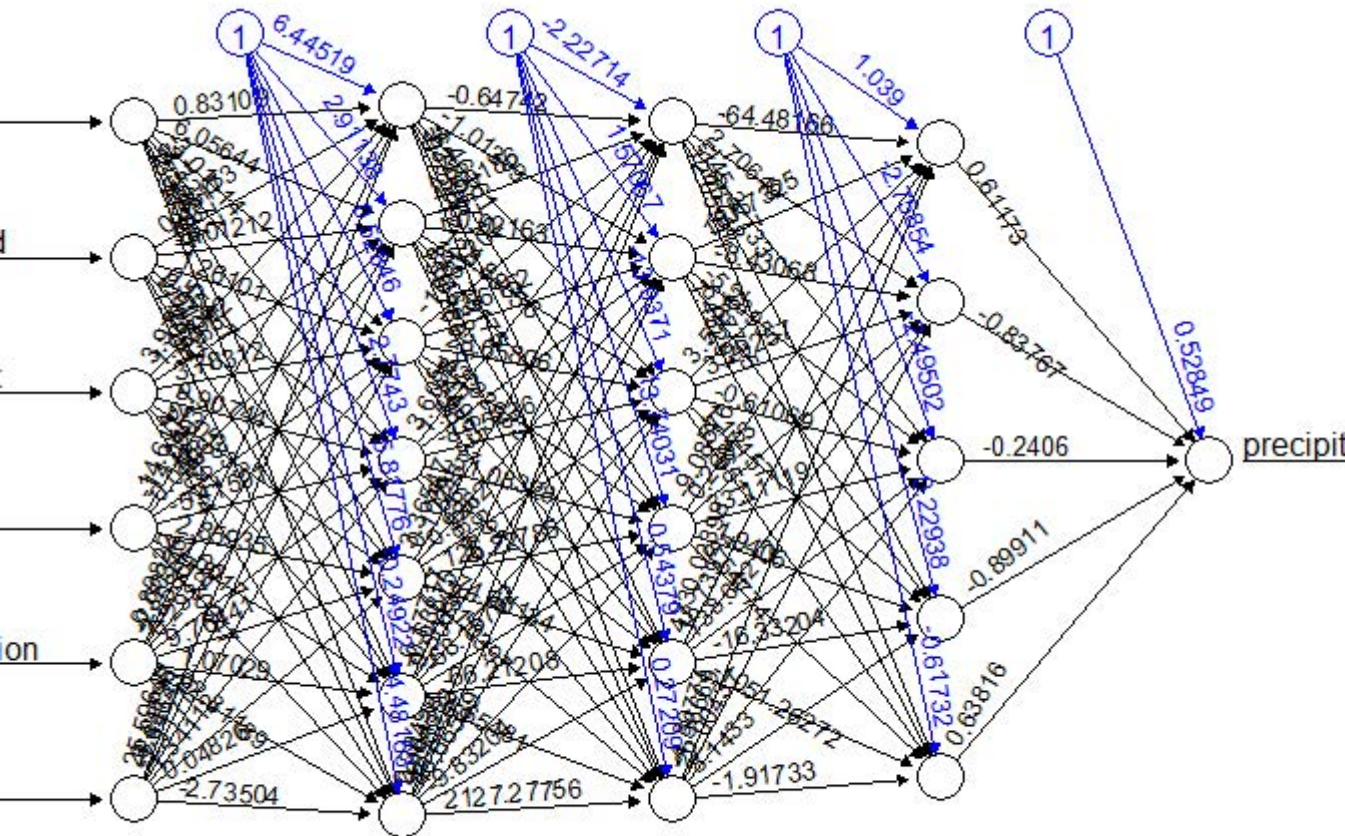
En conclusión la red neuronal nos puede ayudar para obtener predicciones, que aunque con cierto rango de error, nos ayuden a ver que tan grande sera la precipitación dependiendo de la zona, la epoca del año y la temperatura, esto nos puede ayudar a saber que tanto se van a recargar a lo largo del año los mantos acuíferos de la zona.

## Anexo

### Diagrama de Flujo



### Plot Red Neuronal



Error: 2.482134 Steps: 62668