Algoritmos de Inteligencia Artificial y el efecto sobre la fenología de *P.fendleri* bajo escenarios de cambio climático

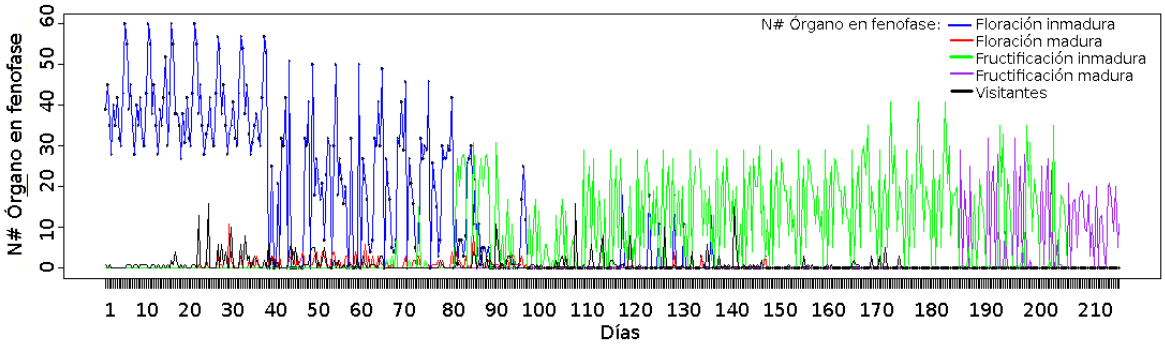
Rodrigo Diaz Lupanow

5 Julio del 2016

Descripcion y preparacion de la data

La variable dependiente es la cantidad de organos de fenofase que tenia la planta en determinado momento desde 2004 hasta 2016. Las variables independientes son: el indice de oscilacion del nino (ONI), el mes en el que se realizó el muestreo, el tipo de órgano correspondiente a la fenofase(1=Flores inmaduras,2=Flores maduras,3=frutos inmaduros, 4=frutos maduros), por último el año de muestreo. Valores de ONI mayores a 0.5 y que se extiendan durante meses corresponden a lo se conoce como el fenómeno El Niño mientras que valores menores a -0.5 se relacionan a eventos La Niña. Muchos investigadores concuerdan que el Cambio Climático altera las oscilaciones temporales que ocurren entre los Niños y las Niñas. Más aun, hipótesis sugieren que el aumento en la intensidad de ambos eventos vienen determinados por el conocido Cambio Climático (principalmente producto de las emisiones antropogénicas). Si esto es cierto, conocer el efecto de las alteraciones en el ONI sobre la fenología podría ser de suma importancia. Grandes producciones agrícolas podrían verse comprometidas, con las implicaciones sociales y economicas que esto acarrearía, además de las repercuciones sobre la conservación de ecosistemas, marítimos y terrestres.

A continuación se muestra un ciclo reproductivo de P.fendleri, una planta del bosque nublado, bajo condiciones ambientales normales.



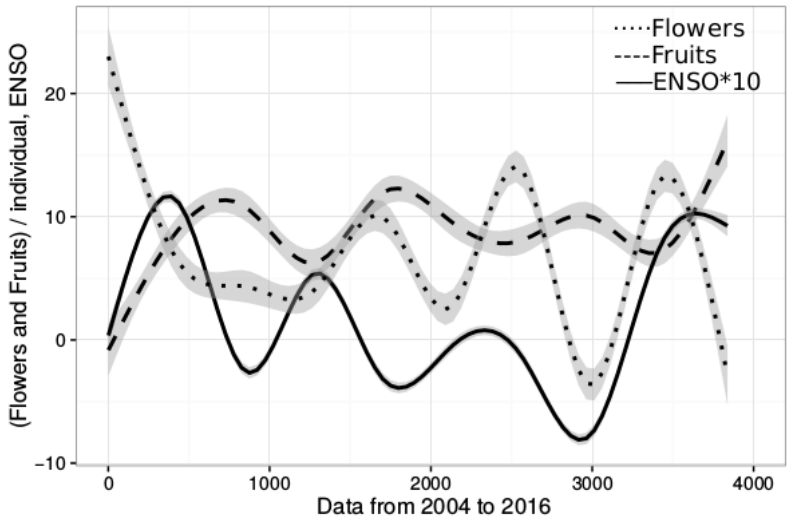
Los datos fueron normalizados(`aleatoreos=sample(1:15352)`)el n muestreal es de 15352 registros. y luegos ordenados en excel en funcion de ese numero. Esto sirvio para mezclar la data de manera aleatorea.

```
nnet_tuned <- read.csv("/media/ro/Backup Data/Dropbox/Data/Data p.fedleri 2004-2005/nnet_tuned")
z=nnet_tuned[,c(-6,-7)]
```

Pero antes, vamos a graficar nuestras variables con ggplot2. Así sabremos si vale la pena utilizar ONI como variable predictora.

```
corriegiendo_2 <- read.csv("/media/ro/Backup Data/Dropbox/Data/Data p.fedleri 2004-2005/corriegiendo_2.csv")
x=corriegiendo_2
library(ggplot2)

a=ggplot(data = x,aes(x= 1:3838,y=x$Frutos.inmaduros))
a+geom_smooth(col="black")+geom_smooth(aes(y=x$ENSO),col=2)+
  geom_smooth(aes(y=log(x$Flores.inmaduras+2)),col=7)+
  geom_smooth(aes(y=log(x$Frutos.inmaduros+1)),col=4)+
  xlab("Sample event from 2004 to 2016")+
  ylab("(log N# Flowers and Fruits)/ind.  ")
```



A partir del gráfico se presume que el ONI puede ayudar a predecir la fenología. De tal forma, procedemos a construir una red neural para predecir el efecto del ONI sobre la fenología.

Primeramente, vamos a conocer los parámetros mas afinados para nuestra red neural.

Para afinar los parámetros utilizamos la funcón `train` del paquete `caret`

```
m <- train(z$N.phe_organs~z$Phenophase+z$ONI+z$Mes+z$Ano,data=z, method = "nnet")
```

```
m
```

. . . size decay RMSE Rsquared

5 1e-01 0.1103795 0.40779077

RMSE was used to select the optimal model using the smallest value.

The final values used for the model were size = 5 and decay = 0.1.

Lo que significa que usaremos 5 capas de neuronas ocultas. Podemos continuar…

```
train <- z[1:10532,]

test <- z[10533:15352,]
```

Se dividio la data en proporcion 3:1 para entrenamiento y prueba. Se escogen las filas y columnas.

```
library(neuralnet)

set.seed(12345)
```

Se llama a la biblioteca necesaria y se ajusta la el `set.seed` para poder repetir el proceso.

```
phenophase <- neuralnet(formula =train$N.phe_organs~train$Phenophase+ train$ONI+
                          train$Mes+train$Ano,
                        data = train,hidden=5,
                        stepmax=1e7,linear.output = F)
```

Se genera un modelo de redes neurales con cinco capas de neuronas tal como señaló el resultado de la función `train` .

```
model_results <- compute(phenophase, test[2:5])
```

El comando `compute` permite predecir con matriz de prueba que habiamos creado. Se utilizan solo las variables independientes o predictoras.

```
 predicted_strength <- model_results$net.result
```

se extraen los resultados de la prediccion. Aca se almacenan las predicciones que utilizaremos despues para analizar los efectos de las variables predictoras sobre la outcome.

```
cor(predicted_strength, test$N.phe_organs)
```

Se ordena la correlacion entre los valores del modelo y los de prueba.

[,1][1,] 0.7000770453

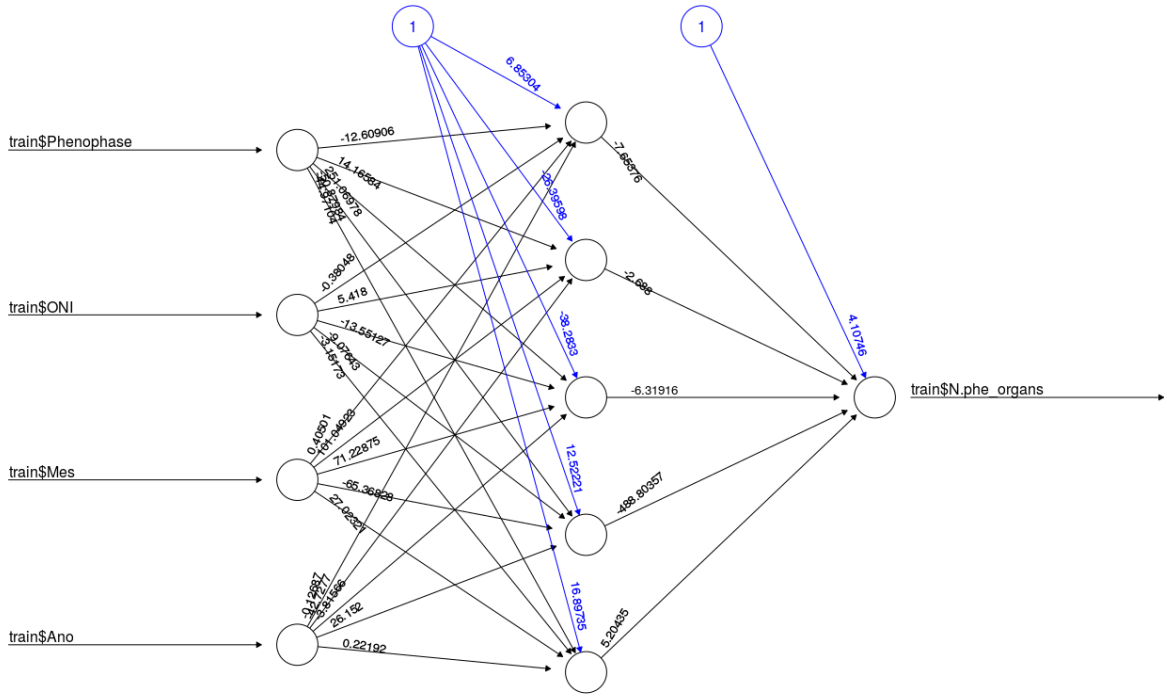
Que implica un 70% de eficiencia del modelo. `hidden=...` podria ser modificado de manera c(5,5) pero lleva mucho tiempo computacional y posiblemente las predicciones no puedan mejorarse.

Podemos ahora usar toda la data para generar la red

```
phenophase <- neuralnet(formula =z$N.phe_organs~z$Phenophase+ z$ONI+
+ z$Mes+z$Ano,
+ data = z,hidden=5,
+ stepmax=1e7,linear.output = F)
```

El tiempo computacional fue alto (45 min aprox.)

Podemos observar la red neural creada.



Error: 55.237011 Steps: 19135

Y que tal si probamos con otros algoritmos de ML??

```
#randomForest
library(randomForest)
nnet_tuned <- read.csv("/media/ro/Backup Data/Dropbox/Data/Data p.fedleri 2004-2005/nnet_tuned")
z=nnet_tuned[,c(-6,-7)]
o=randomForest(train$N.phe_organs~.,
               data = train,xtest=test[,2:5],ytest=test[,1],keep.forest=TRUE,test=test)
p=predict(o,test[,2:5],type = "response")
cor(p,test$N.phe_organs)
```

[1] 0.7611076

76% en eficiencia de predicción, algo superior a las Redes Neurales previamente creadas. Además, el tiempo computacional fue importantemente menor (1 a dos minutos max.)

```
library(randomForest)
```

```
## randomForest 4.6-12
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
nnet_tuned <- read.csv("/media/ro/Backup Data/Dropbox/Data/Data p.fedleri 2004-2005/nnet_tuned")
z=nnet_tuned[,(c(-6,-7))]
train <- z[1:10532,]
test <- z[10533:15352,]
o=(randomForest(train$N.phe_organs~.,
                data = train,xtest=test[,2:5],ytest=test[,1],keep.forest=TRUE,test=test))
summary(o)
```

```
##                 Length Class  Mode
## call                7  -none- call
## type                1  -none- character
## predicted       10532  -none- numeric
## mse               500  -none- numeric
## rsq               500  -none- numeric
## oob.times       10532  -none- numeric
## importance          4  -none- numeric
## importanceSD        0  -none- NULL
## localImportance     0  -none- NULL
## proximity           0  -none- NULL
## ntree               1  -none- numeric
## mtry                1  -none- numeric
## forest             11  -none- list
## coefs               0  -none- NULL
## y               10532  -none- numeric
## test                4  -none- list
## inbag               0  -none- NULL
## terms               3  terms  call
```

```
o
```

```
##
## Call:
##  randomForest(formula = train$N.phe_organs ~ ., data = train,      xtest = test[, 2:5], ytest = test[, 1], kee
p.forest = TRUE,      test = test)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 1
##
##           Mean of squared residuals: 0.01051419
##                     % Var explained: 48.14
##                        Test set MSE: 0.01
##                     % Var explained: 49.19
```

El valor de Mean of squared residuals= 0.01046546 debe multiplicarse por 70(max)–>desnormalizar. Lo que nos da residuales de 0.7 organos de fenofase. Es decir, el promedio de la lejania de los puntos no ajustados, a la predicción misma del modelo, es menor a 1. Anunciando que si bien el modelo no tiene precisión del 100%, el promedio de los “desaciertos” es bastante bajo. Sin embargo, la varianza % Var explained: 49.37 explicada es medio mala. Quizas agregando la variable estacionalidad real al modelo.

Vamos a revisar que tal es el error del modelo. Lo corremos 10 veces con la función `errorest` del paquete ipred Instalar si no se tiene.

```
for(i in 1:10) error.RF[i] <-errorest(N.phe_organs ~ ., data = z,model = randomForest,mtry = 2)$error
```

```
summary(error.RF)
```

Min. 1st Qu. Median Mean 3rd Qu. Max. 0.07988 0.07998 0.08018 0.08019 0.08028 0.08063

Ahora nos preguntamos como son los errores en otros modelos antes de evaluarlos. SVM `library(e1071) set.seed(563)`

```
error.neuralnet <- numeric(10)
for (i in 1:10) error.neuralnet[i] <-errorest(N.phe_organs ~ .,data = z, model = svm,cost = 10, gamma = 1.5)$error
summary(error.neuralnet)
```

Min. 1st Qu. Median Mean 3rd Qu. Max. 0.09646 0.09665 0.09675 0.09682 0.09705 0.09716

Para esto el tiempo computacional si es alto.

Bagging

```
for (i in 1:10) error.bagging[i]=errorest(N.phe_organs~.,data=z,model=bagging,cost = 10, gamma = 1.5)$error
summary(error.bagging)
```

Min. 1st Qu. Median Mean 3rd Qu. Max. 0.09602 0.09646 0.09650 0.09648 0.09658 0.09677

Nnet `for (i in 1:10) error.bagging[i]=errorest(N.phe_organs~.,data=z,model=nnet,size=5,cost = 10, gamma = 1.5)$error`

```
summary(error.neuralnet)
```

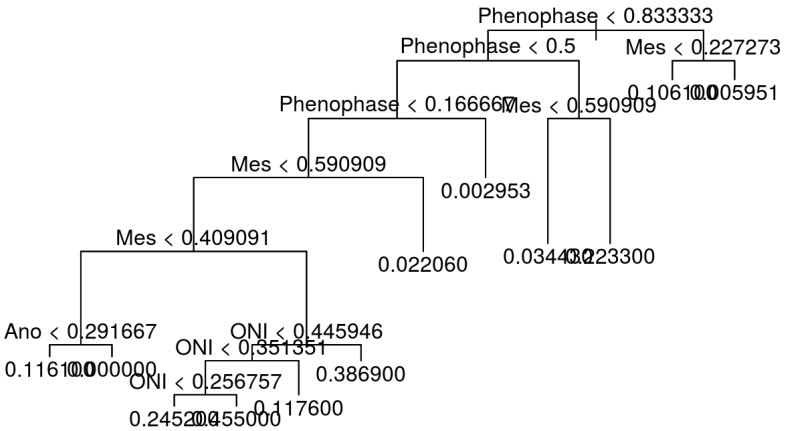
Min. 1st Qu. Median Mean 3rd Qu. Max.

0.1543 0.1543 0.1543 0.1543 0.1543 0.1543

```
library(tree)
 a=tree(N.phe_organs~.,data = train)
 summary(a)
```

```
##
## Regression tree:
## tree(formula = N.phe_organs ~ ., data = train)
## Number of terminal nodes:  12
## Residual mean deviance:  0.01055 = 111 / 10520
## Distribution of residuals:
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -0.455000 -0.022060 -0.005951  0.000000 -0.002953  0.877900
```

```
plot(a)
text(a, pretty = 0)
```




```
yhat <- predict(a, newdata = test)
cor(yhat,test$N.phe_organs)
```

```
## [1] 0.6864828
```

Qué tal son las predicciones del Nnet bajo escenarions hipotéticos?

En el caso del Neural Net evaluemos como sería la floración y fructificación si un evento El Niño se prolonga durante dos años. Evaluamos varias intensidades de El Niño, simulando un aumento del ONI de 100, 200 y 300%. A su vez, se graficaron escenarios “control”, con un ONI promedio (2004-2016) y escenarios de “valor máximo de ONI”" (entre 2004-2016).

Se ajustaron los parámetros óptimos con la función `train`

```
train <- z[1:15535,]#Ahora usamos toda la data.
```

```
phenophase <- neuralnet(formula =train$N.phe_organs~train$Phenophase+ train$ONI+ train$Mes+train$Ano, data = train,hidden=5, step
```

Se organizan las gráficas en 2x1 y ajuste de tamaño de letra en eje y etiqueta.

```
par(mfrow=c(2,1),cex.axis=1.5,cex.lab=1.5)
```

Sin una función de desescalamiento es necesario conocer los intervalos max y min de los datos para ajustarlos en la data.frame `t` . Para entender mejor el significado de los valores hay que saber que el valor 1 equivale al máximo valor en la variable y 0 al mínimo. El desescalamiento lo haremos más adelante al graficar la variable dependiente.

```
library(xtable)
```

```r
t=data.frame(c(rep(0,36),rep(0,36),rep(0,36),rep(0,36),rep(0,36)),#fenofase
c(rep(0.5,36),rep(0,36),rep(-1,36),rep(-2,36),rep(-3,36)),#ENSO
rep(c(0,0.090,0.181,0.272,0.363,0.454,0.545,0.636,0.727,0.818,0.909,1,
0,0.090,0.181,0.272,0.363,0.454,0.545,0.636,0.727,0.818,0.909,1,0,
0.090,0.181,0.272,0.363,0.454,0.545,0.636,0.727,0.818,0.909,1),5),#meses
c(rep(3.833,12),rep(3.916,12),rep(4,12)))#anio
```

Con nuestra matriz de 4 columnas (las variables independientes) por 180 filas (valores en función de los cuales queremos predecir), vamos a utilizar la función `compute` para obtener las predicciones.

```
model_results <- compute(phenophase, t)
```

Extraemos la matriz de resultados de la red `predicted_strength <- model_results$net.result` Graficamos el resultado de las predicciones bajo los escenarios planteados. En este primer caso, flores inmaduras en caso de distintas intensidades negativas de ONI (Eventos La Nina)

```
plot(1:36,predicted_strength[1:36]*70,type = "l",lty=1,lwd=2,xlab = "mounths from 2020",ylab = "n# predicted flow
ers",xlim=c(0,30),ylim = c(0,4),main = "La Nina intensity effect on phenology")
```

lines(predicted_strength[37:72]*70,lty=2,lwd=2) lines(predicted_strength[73:108]*70,lty=3,lwd=2) lines(predicted_strength[109:144]*70,lty=4,lwd=2) lines(predicted_strength[145:180]*70,lty=5,lwd=2) legend(27,4,legend = c("x1/2","0","x-1","x-2","x-3"),lty=c(1:5),cex = 0.75) ```

Ahora para los frutos inmaduros igualmente escenario La Nina.

```r
t=data.frame(c(rep(0.66,36),rep(0.66,36),rep(0.66,36),rep(0.66,36),rep(0.66,36)),#fenofase
c(rep(0.5,36),rep(0,36),rep(-1,36),rep(-2,36),rep(-3,36)),#ENSO
rep(c(0,0.090,0.181,0.272,0.363,0.454,0.545,0.636,0.727,0.818,0.909,1,
0,0.090,0.181,0.272,0.363,0.454,0.545,0.636,0.727,0.818,0.909,1,0,
0.090,0.181,0.272,0.363,0.454,0.545,0.636,0.727,0.818,0.909,1),5),#meses
c(rep(3.833,12),rep(3.916,12),rep(4,12)))#anio
```

```
model_results <- compute(phenophase, t)  predicted_strength <- model_results$net.result
```

```
     type = "l",lty=1,lwd=2,xlab = "Mounths from 2020",
     ylab = "n# predicted fruits",xlim=c(0,30),ylim = c(0,20))
```

```
lines(predicted_strength[73:108]*55,lty=3,lwd=2)
lines(predicted_strength[109:144]*55,lty=4,lwd=2)
lines(predicted_strength[145:180]*55,lty=5,lwd=2)
legend(27,20,legend = c("x1/2","x0","x-1","x-2","x-3"),lty=c(1:5),cex = 0.75)
```

Ahora escenarios Ninos:

Volvemos a ajustar los parámetros de generación de gráficos de R. `par(mfrow=c(2,1),cex.axis=1.5,cex.lab=1.5)` Generamos nuestra matriz molde para predecir pero esta vez en ENSO ajustamos valores de ONI positivos, modelando escenarios de Ninos intensos y prolongados y su efecto sobre la floración inmadura.

```
r   t=data.frame(c(rep(0,36),rep(0,36),rep(0,36),rep(0,36),rep(0,36)),#fenofase   c(rep(0.5,36),rep(1,36),rep(2,36),rep(3,36),rep
```

```
model_results <- compute(phenophase, t)  predicted_strength <- model_results$net.result
plot(1:36,predicted_strength[1:36]*70,      type = "l",lty=1,lwd=2,xlab = "Mounths from 2020",      ylab = "n# predicted flowers'
```

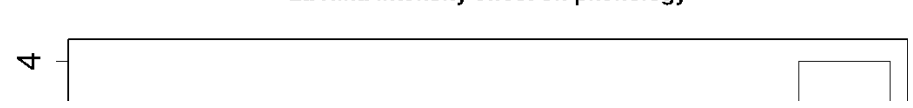
Lo mismo para los frutos inmaduros.

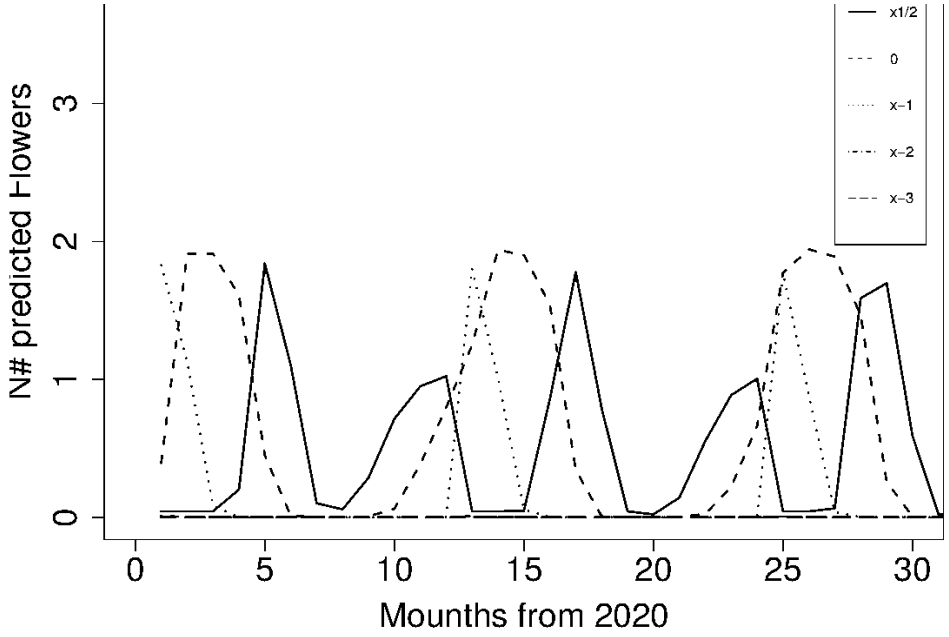
```
r   t=data.frame(c(rep(0.66,36),rep(0.66,36),rep(0.66,36),rep(0.66,36),rep(0.66,36)),#fenofase              + c(rep(0.5,36),rep(1
```

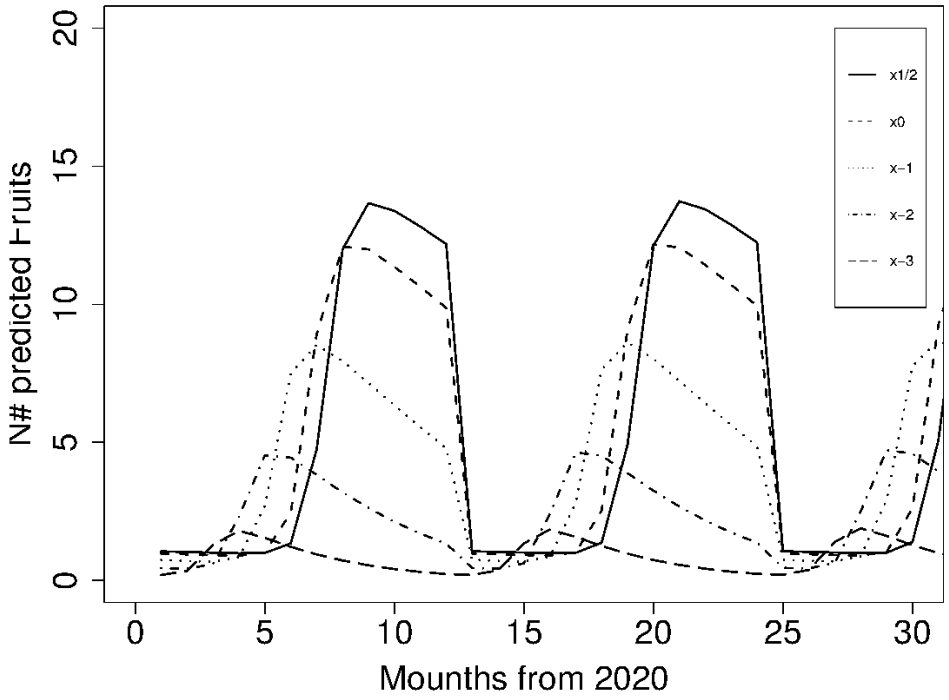
```
model_results <- compute(phenophase, t)  predicted_strength <- model_results$net.result
```

```
plot(1:36,predicted_strength[1:36]*55,
     type = "l",lty=1,lwd=2,xlab = "Mounths from 2020",
     ylab = "n# predicted fruits",xlim=c(0,30),ylim = c(0,20))
```

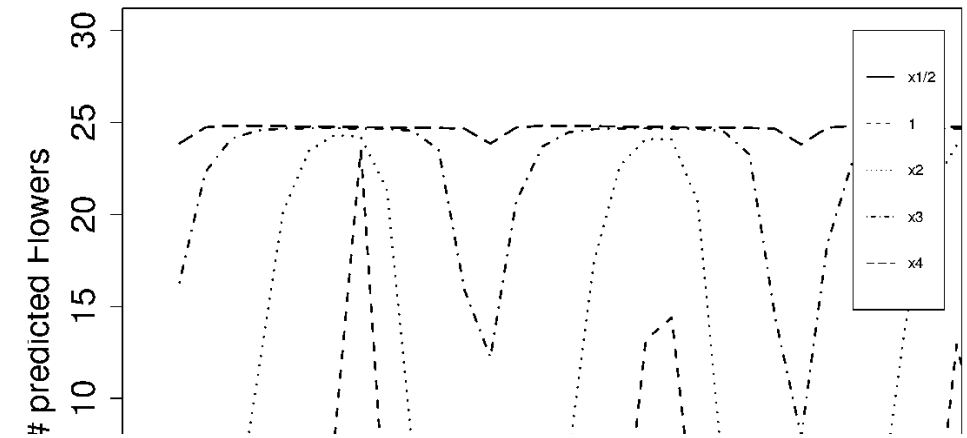
```
lines(predicted_strength[37:72]*55,lty=2,lwd=2)
lines(predicted_strength[73:108]*55,lty=3,lwd=2)
lines(predicted_strength[109:144]*55,lty=4,lwd=2)
lines(predicted_strength[145:180]*55,lty=5,lwd=2)
legend(27,20,legend = c("x1/2","x1","x2","x3","x4"),lty=c(1:5),cex = 0.75)
```

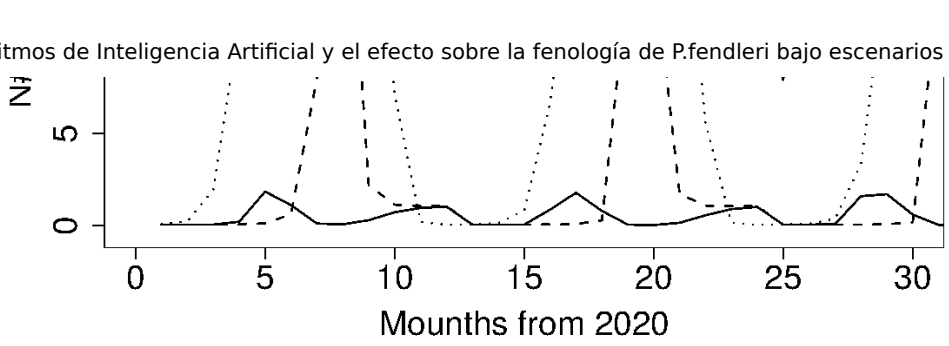


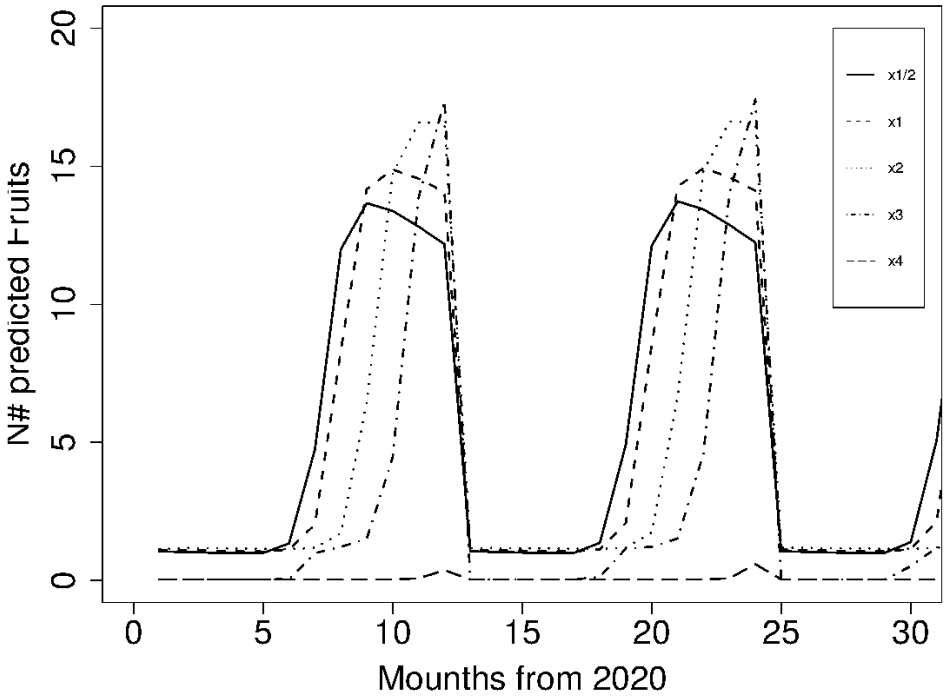




El Nino intensity effect on phenology







Comentarios finales

El modelo random Forest tuvo mejor desempeño y requirió menor tiempo computacional que las Redes Neurales. Sin embargo, la función `compute` del paquete `neuralnet` permitió calcular la producción de organos fenológicos bajo diferentes escenarios. Se plantea como objetivo próximo graficar los escenarios de la predicciones del modelo random Forest, ya que, al ser más eficiente, podría arrojar predicciones más precisas. A su vez, las predicciones deben ser grafidas con su error asociado. La generación de este error conlleva mucho tiempo computacional, por lo que se plantean a futuro.

Analizando varios algoritmos de IA, el menor error fue arrojado por el modelo de random Forest, seguido de bagging, svm y nnet. Sin embargo, para grandes cantidades de datos como esta, algoritmos de inteligencia artificial son mejor opción para predicir que la regresión clasica, con la cual se optuvo eficiencia de predicción de apenas 12% con el modelo saturado (no mostrado en scrip). Estas herramientas son sumamente útiles para modelar escenarios hipotéticos, así como para entender relaciones o asociaciones que no se ven a simple vista. Ejemplo de esto último fue lo obtenido con los árboles de regresión.

Por último, la variedad de algorimos de IA disponibles, ya sea para clasificación o predicción, permiten obtener predicciones eficientes donde la estadística clásica no. Estas poderosas herramientas ofrecen al análisis nuevas perspectivas, desenmarañando el intrincado misterio de los datos y dando a la luz resultados sorprendentes. El universo apenas empieza a ser explicado…

……
. . . . . .