Random Forest Regression

Humberto Vaquera Huerta

31/7/2020

// Bosques Aleatorios(Random Forest)

Los bosques aleatorios son similares a una famosa técnica de Ensemble llamada “Bagging”, pero tienen un ajuste diferente. En Random Forests, la idea es relacionar los varios árboles que se generan en las diferentes muestras de bootstrap de los datos de entrenamiento, y luego simplemente reducimos la varianza en los árboles promediando. Promediar los árboles nos ayuda a reducir la variación y también a mejorar el rendimiento de los árboles de decisión en el conjunto de pruebas y, finalmente, evitar el sobreajuste.

// Bosques Aleatorios(Random Forest)

- Random Forest se considera como la “panacea” en todos los problemas de ciencia de datos.
- Util para regresión y clasificación.
- Un grupo de modelos “débiles”, se combinan en un modelo robusto.
- Sirve como una técnica para reducción de la dimensionalidad.
- Se generan múltiples árboles (a diferencia de CART).
- Cada árbol da una classificación (vota por una clase). Y el resultado es la clase con mayor número de votos en todo el bosque (forest).
- Para regresión, se toma el promedio de las salidas (predicciones) de todos los árboles.

// Ventajas de Random Forest

- Existen muy pocas suposiciones y por lo tanto la preparación de los datos es mínima.
- Puede manejar hasta miles de variables de entrada e identificar las más significativas. Método de reducción de dimensionalidad.
- Una de las salidas del modelo es la importancia de variables. Incorpora métodos efectivos para estimar valores faltantes.
- Es posible usarlo como método no supervisado (clustering) y detección de outliers.

// Desventajas de Random Forest

- Pérdida de interpretación
- Bueno para clasificación, no tanto para regresión. Las predicciones no son de naturaleza continua.
- En regresión, no puede predecir más allá del rango de valores del conjunto de entrenamiento.
- Poco control en lo que hace el modelo (modelo caja negra para modeladores estadísticos)

// caracteristicas:

- Aplicable tanto a problemas de regresión como de clasificación.
- Maneja predictores categóricos de forma natural.
- Computacionalmente simple y rápido de instalar, incluso para problemas grandes.
- Sin supuestos de distribución formales (no paramétricos).
- Puede manejar interacciones altamente no lineales y límites de clasificación.
- Selección automática de variables. sí. Pero también necesita una importancia variable.
- Maneja valores perdidos

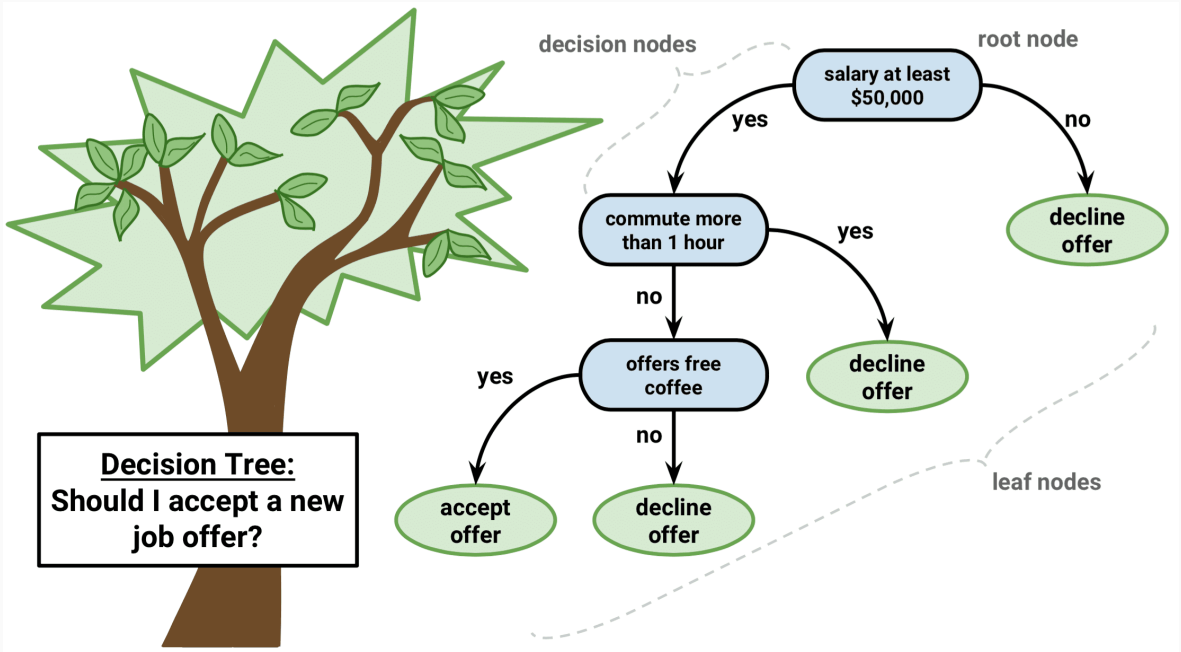
// Contruccion de un modelo random forest

Cada árbol se construye así:

- Dado que el número de casos en el conjunto de entrenamiento es N. Una muestra de esos N casos se toma aleatoriamente pero CON REEMPLAZO. Esta muestra será el conjunto de entrenamiento para construir el árbol i.

- Si existen M varibles de entrada, un número m<M se especifica tal que para cada nodo, m variables se seleccionan aleatoriamente de M. La mejor división de estos m atributos es usado para ramificar el árbol. El valor m se mantiene constante durante la generación de todo el bosque.
- Cada árbol crece hasta su máxima extensión posible y NO hay proceso de poda.
- Nuevas instancias se predicen a partir de la agregación de las predicciones de los x árboles (i.e., mayoría de votos para clasificación, promedio para regresión)

// Arbol de decision

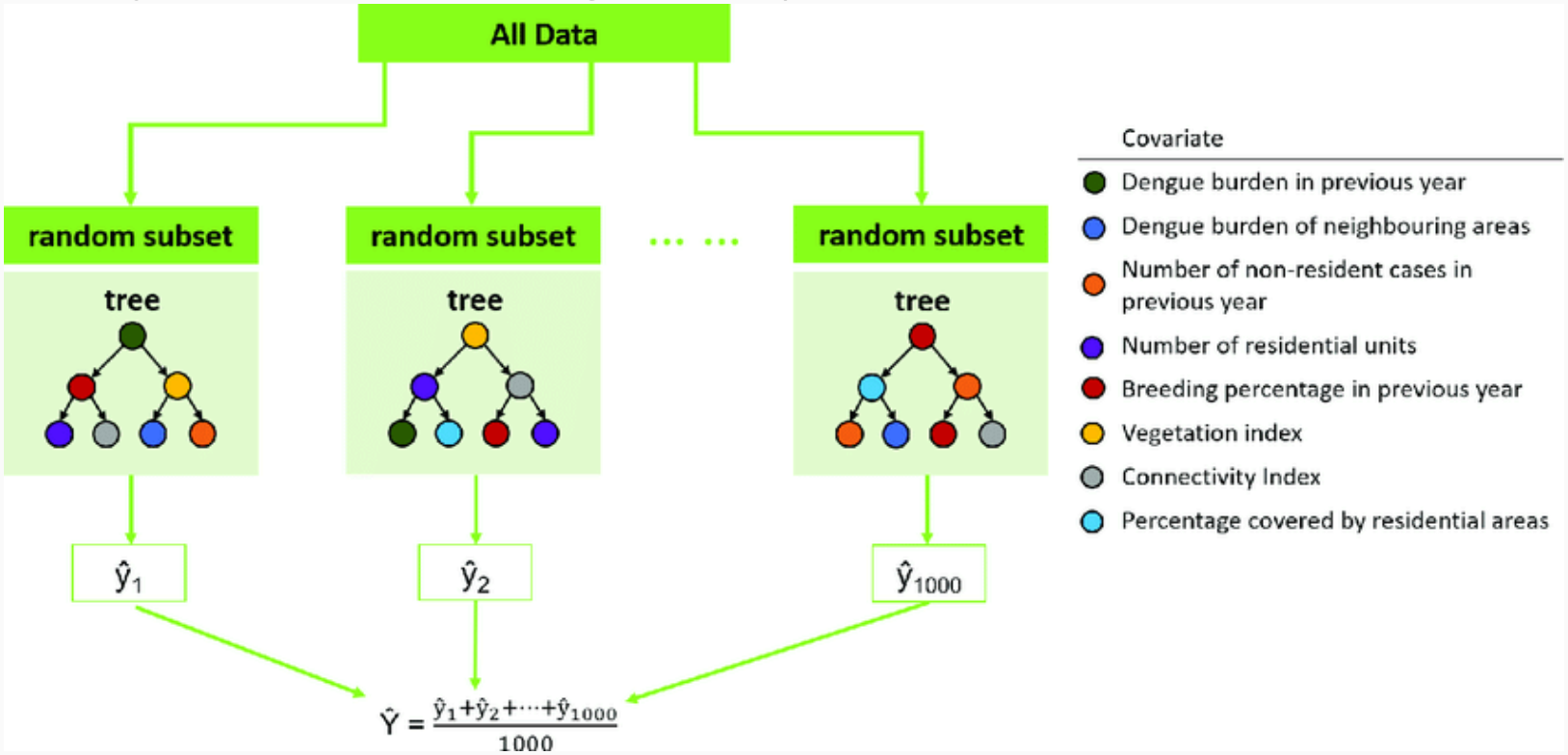


arbol de decision

// Random Forest Regression y Bootstrap

- El proceso de muestreo de los datos con reemplazo se denomina bootstrap.
- Un tercio de los datos no se usan para el entrenamiento y pueden ser usados para test.

- Este conjunto se denomina out of bag (OOB)samples.



// Impementacion en R de Random Forest

Librerias requeridas

```
library(randomForest)
library(MASS)# datos Boston housing
attach(Boston)
set.seed(101)
head(Boston)
```

// Separando conjunto de datos de entrenamiento y prueba

Muestra de entrenamiento 300 datos

```r
str(Boston)  #info de datos Boston
```

Ajustando la regresion Random Forest

Usando todos los predictores.

```r
Boston.rf=randomForest(medv~.,data=Boston)
Boston.rf
plot(Boston.rf)
pred_randomForest <- predict(Boston.rf, Boston)
head(pred_randomForest)
```

// Ejemplo 2

Librerias requeridas

```r
library(AmesHousing) # datos de ejemplo
library(rsample)     # data splitting
library(dplyr)       # data wrangling
library(ranger)      # más rápida que randomforest
library(h2o)         # computación distribuida
library(ggplot2)
#Cargamos los datos y los separamos en entreanmiento y test.
# Entrenamiento (70%) y test (30%) a partir de AmesHousing::make_ames() data.
# Usar semilla para reproducibilidad: set.seed

set.seed(123)
ames_split<-initial_split(AmesHousing::make_ames(),
                          prop =.7)
ames_train <- training(ames_split)
```

```r
ames_test  <- testing(ames_split)
str(ames_train)
```

// Ajuste de dafaut ejemplo2 RF

```r
set.seed(123)
library(randomForest)
# default RF model
modeloRF_def <- randomForest(
  formula = Sale_Price ~ .,
  data    = ames_train
)

modeloRF_def
# Numero de arboles con minimo mse
which.min(modeloRF_def$mse)
# RMSE de este modelo óptimo RF
sqrt(modeloRF_def$mse[which.min(m1$mse)])
```

// Utilizando la librería ranger

Esta librería puede ser 6 veces más rápida que randomforest.

```r
library(ranger)
# default RF model
modeloranger_def <- ranger(
    formula         = Sale_Price ~ .,
    data            = ames_train)
```

```
summary(modeloranger_def)
modeloranger_def
```

// Rendimiento de Maiz en Mexico por Hectarea

Datos: Datos de FAO http://www.fao.org/faostat/en/#data/QC

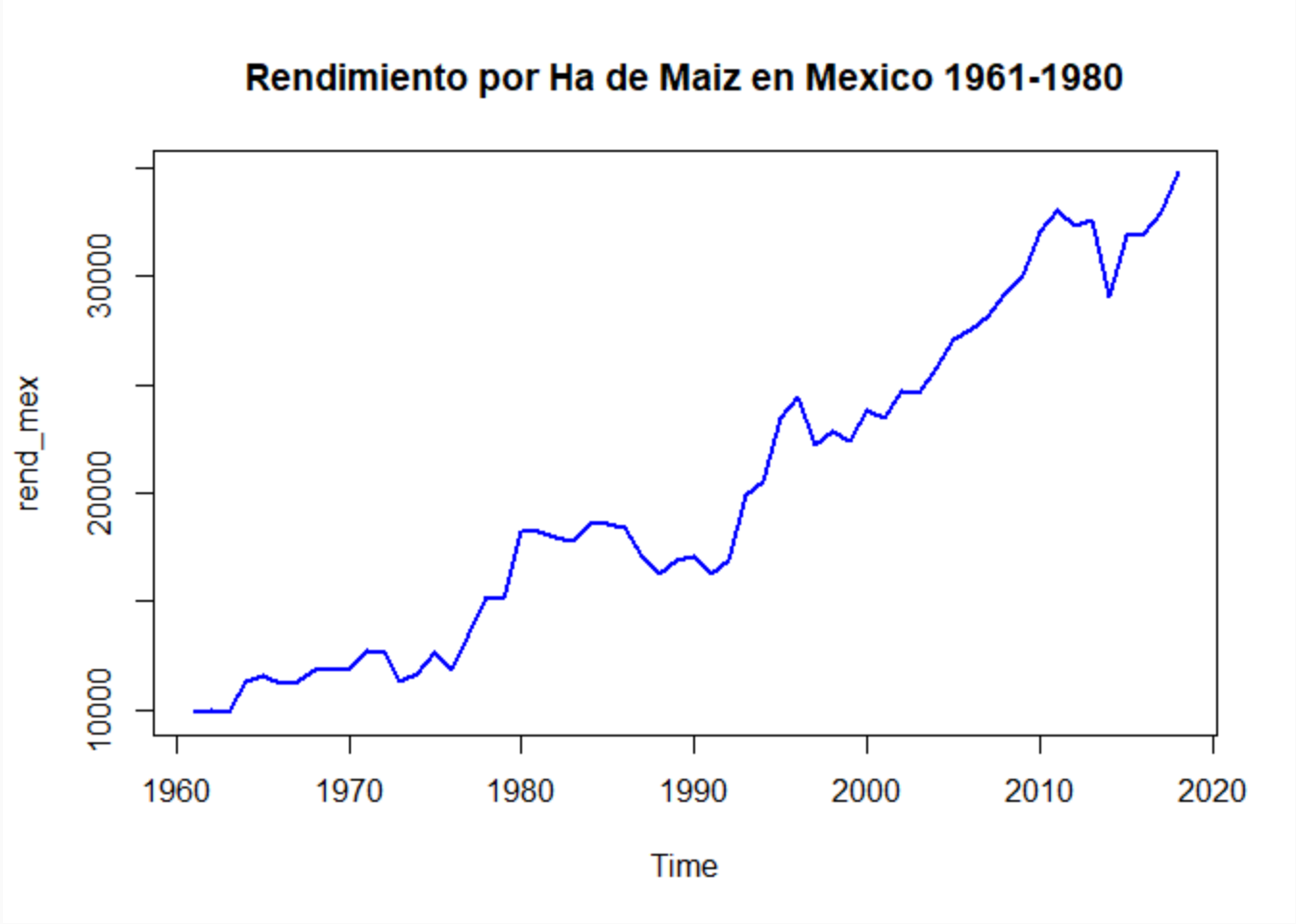
Year	Rend_maiz	Year	Rend_maiz
1961	9934	1989	16929
1962	9946	1990	19942
1963	9867	1991	20515
1964	11332	1992	23450
1965	11578	1993	24401
1966	11188	1994	22255
1967	11304	1995	22883
1968	11806	1996	22387
1969	11840	1997	23840
1970	11935	1998	23429
1971	12723	1999	24720
1972	12648	2000	24620

Year	Rend_maiz	Year	Rend_maiz
1973	11318	2001	25777
1974	11683	2002	27105
1975	12621	2003	27525
1976	11819	2004	28188
1977	13572	2005	29276
1978	15199	2006	30012
1979	15154	2007	32063
1980	18261	2008	33071
1981	18240	2009	32368
1982	17976	2010	32599
1983	17770	2011	29058
1984	18554	2012	31874
1985	18583	2013	31941
1986	18406	2014	32964
1987	17058	2015	34782
1988	16289	2016	37180

Year	Rend_maiz	Year	Rend_maiz
1989	16929	2017	37888
1987	17058	2018	38146
1988	16289		

// Rendimiento de Maiz en Mexico Hg/Hectarea

```
        maiz.rend <- read.csv("D:/cursos/no param/noparam/ejemplo gam/maiz-rend.csv")
        rend_mex <- ts(maiz.rend$Rend_maiz,start=c(1961),end=c(2018),frequency=1)
        plot(rend_mex,col="blue",lwd=2,main="Rendimiento por Ha de Maiz en Mexico 1961-
1980")
```



// Ajuste con Random Forest regression

```
library(randomForest)
```

```
## randomForest 4.7-1.1
```

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

```r
RF_Maiz <- randomForest(Rend_maiz~ Year,
                        data =maiz.rend)
RF_Maiz
```

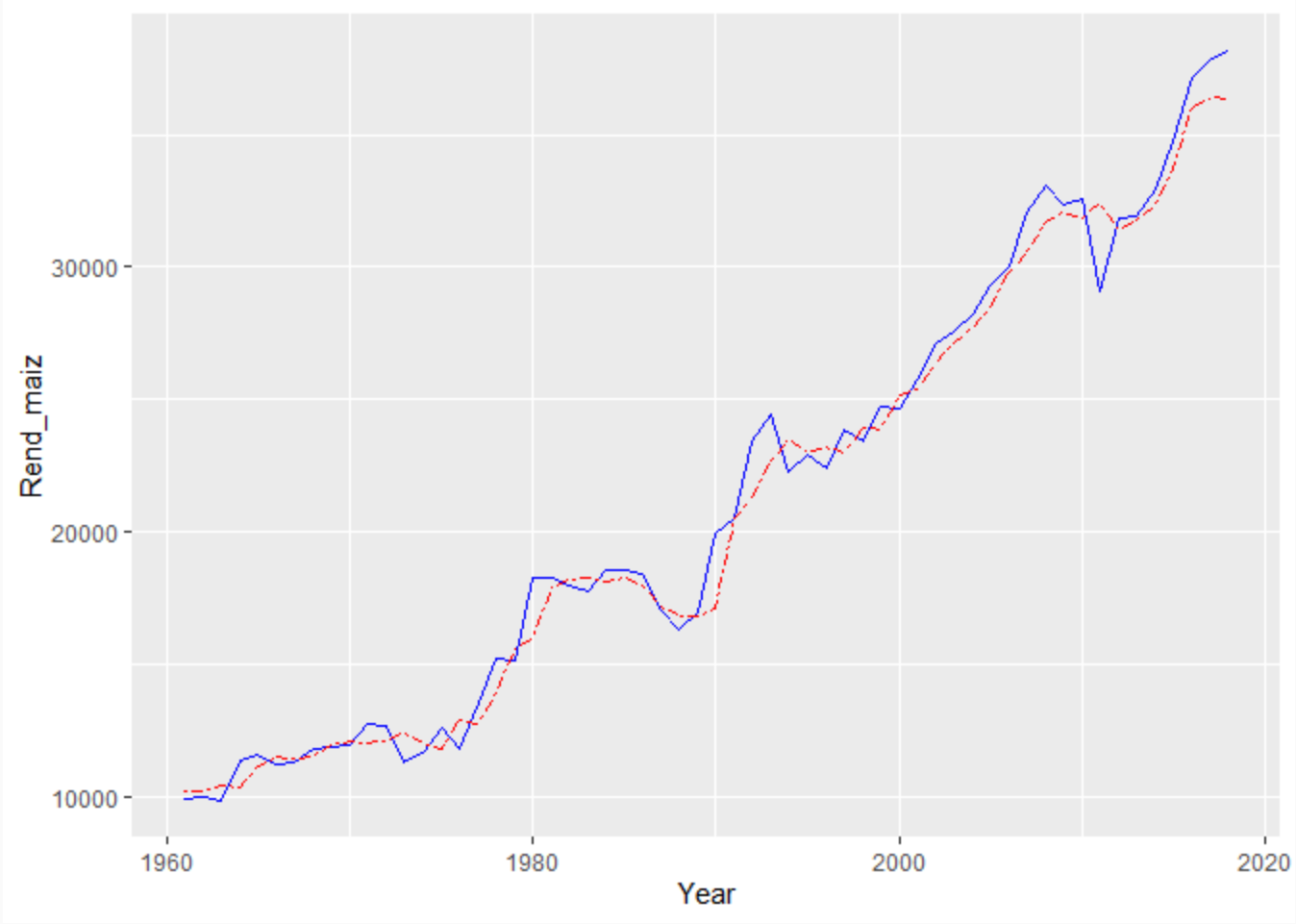
```
##
## Call:
##  randomForest(formula = Rend_maiz ~ Year, data = maiz.rend)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 1
##
##           Mean of squared residuals: 1022738
##                     % Var explained: 98.49
```

// Prediccion

```r
library(ggplot2)
layout(matrix(1:2, nrow = 1))
predic=data.frame(cbind(predic_rend=predict(RF_Maiz),maiz.rend))
ggplot(predic, aes(x=Year)) +
  geom_line(aes(y = Rend_maiz),color ="blue") +
  geom_line(aes(y = predic_rend), color="red", linetype="twodash")
```

// Grafica

```
## Warning: package 'ggplot2' was built under R version 4.2.3
```



// prediccion rendimiento de Maiz Hg/Ha 2019-2020

```
a_2019_2020=data.frame(Year=c(2019,2020))
p_2019_2020=predict(RF_Maiz,a_2019_2020)
data.frame(cbind(a_2019_2020,p_2019_2020))
```

```
##   Year p_2019_2020
## 1 2019    37054.74
## 2 2020    37054.74
```

```
        predict(RF_Maiz,a_2019_2020)
```

```
##        1        2
## 37054.74 37054.74
```

// Datos de propiedades del suelo

Varias propiedades químicas del suelo medidas en una cuadrícula regular con 10x25 puntos espaciados por 5 metros.

250 observaciones sobre las siguientes 22 variables:

- Coordenada x de Linha
- Coordenada y de Coluna
- Elevación de la cota
- AGrossa un vecto numérico, una porción de arena de la muestra.
- Silte un vector numérico, porción de limo de la muestra.
- Argila un vector numérico, una porción de arena de la muestra.
- pHAgua un vector numérico, pH del suelo en agua
- pHKCl un vector numérico, pH del suelo por KCl
- Ca un vector numérico, contenido de calcio
- Mg a vector numérico, contenido de magnesio
- K un vector numérico, contenido de potasio
- Al un vector numérico, contenido de aluminio
- H un vector numérico, contenido de hidrógeno
- C un vector numérico, contenido de carbono
- N un vector numérico, contenido de nitrógeno
- CTC un vector numérico, capacidad de intercambio de cationico
- S un vector numérico, contenido enxofrar

- V un vector numérico
- M un vector numérico
- NC un vector numérico
- CEC un vector numérico
- CN un vector numérico, relación carbono / nitrógeno

dela libreria **geoR**

```
library(geoR)
```

```
## Warning: package 'geoR' was built under R version 4.2.2
```

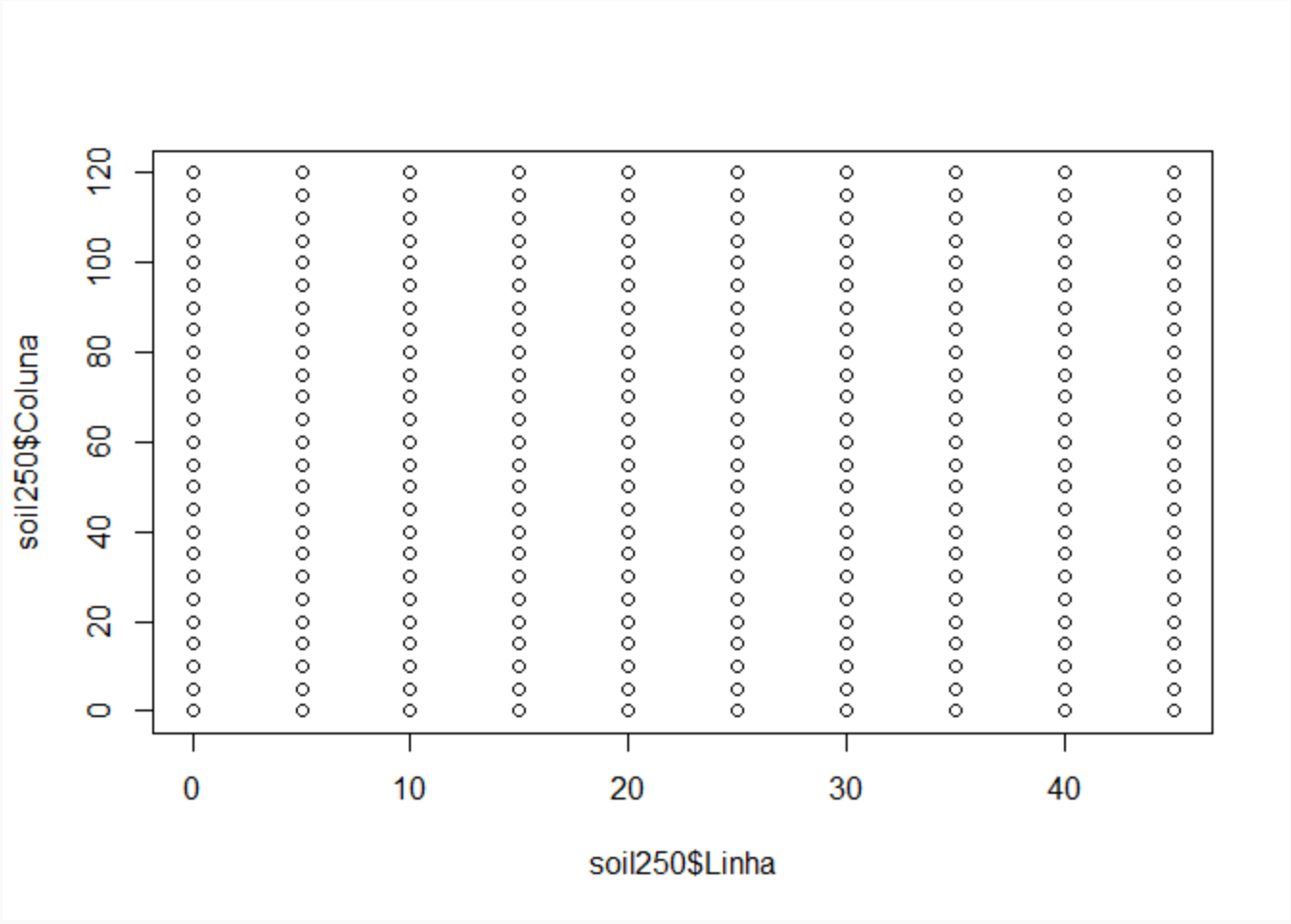
```
## ----------------------------------------------------------
##  Analysis of Geostatistical Data
##  For an Introduction to geoR go to http://www.leg.ufpr.br/geoR
##  geoR version 1.9-2 (built on 2022-08-09) is now loaded
## ----------------------------------------------------------
```

```
data(soil250)
head(soil250)
```

```
##   Linha Coluna    Cota AGrossa Silte Argila pHAgua pHKCl  Ca  Mg    K Al   H
## 1     0      0 578.295       9    26     43    5.8   4.9 3.6 0.8 0.50  0 3.1
## 2     0      5 578.460       9    26     42    5.9   4.9 3.4 0.8 0.44  0 3.2
## 3     0     10 578.491       9    25     41    5.9   4.9 3.7 0.9 0.59  0 2.5
## 4     0     15 578.699      11    28     40    5.8   4.9 3.7 0.8 0.52  0 3.5
## 5     0     20 578.749       9    27     41    5.8   5.0 4.2 0.9 0.56  0 3.4
```

```
## 6     0     25 578.726       8    27     43    6.1   5.1 4.1 0.9 0.56  0 3.1
##     C    N  CTC    S      V M    NC  CEC CN
## 1 1.2 0.12 8.00 4.90 61.250 0 1.050 4.90 10
## 2 1.1 0.12 7.84 4.64 59.184 0 1.272 4.64  9
## 3 1.2 0.13 7.69 5.19 67.490 0 0.289 5.19  9
## 4 1.3 0.12 8.52 5.02 58.920 0 1.416 5.02 10
## 5 1.4 0.13 9.06 5.66 62.472 0 1.023 5.66 10
## 6 1.2 0.13 8.66 5.56 64.203 0 0.753 5.56  9
```

```
plot(soil250$Linha, soil250$Coluna)
```

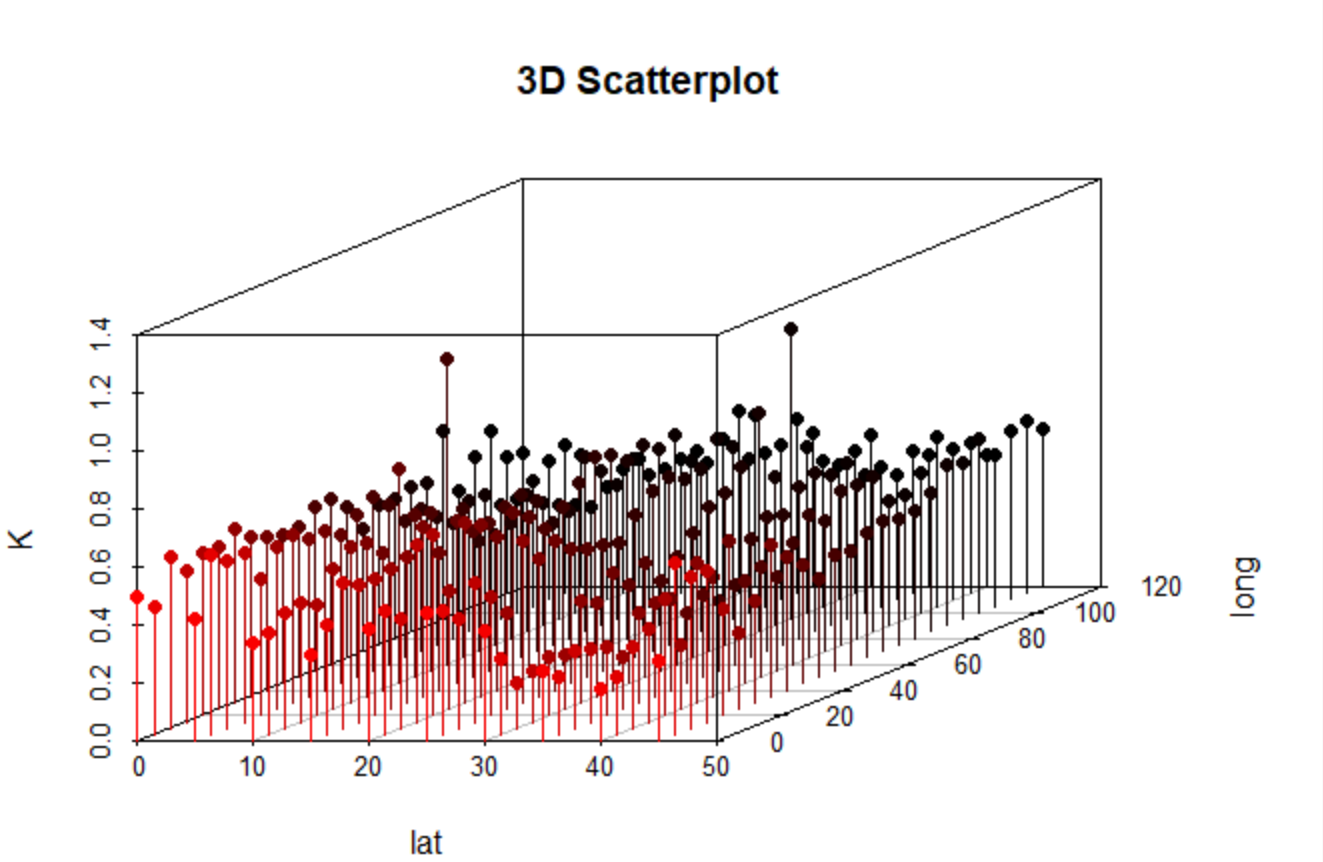


```r
lat=soil250$Linha
long=soil250$Coluna
K=soil250$K
```

```r
library(scatterplot3d)
```

```
## Warning: package 'scatterplot3d' was built under R version 4.2.3
```

```r
scatterplot3d(lat,long,K, pch=16, highlight.3d=TRUE,
              type="h", main="3D Scatterplot")
```




```
library(randomForest)
RF_K <- randomForest(K~lat+long)
RF_K
```

```
##
## Call:
##  randomForest(formula = K ~ lat + long)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 1
```

```
##
##           Mean of squared residuals: 0.006011621
##                     % Var explained: 57.63
```

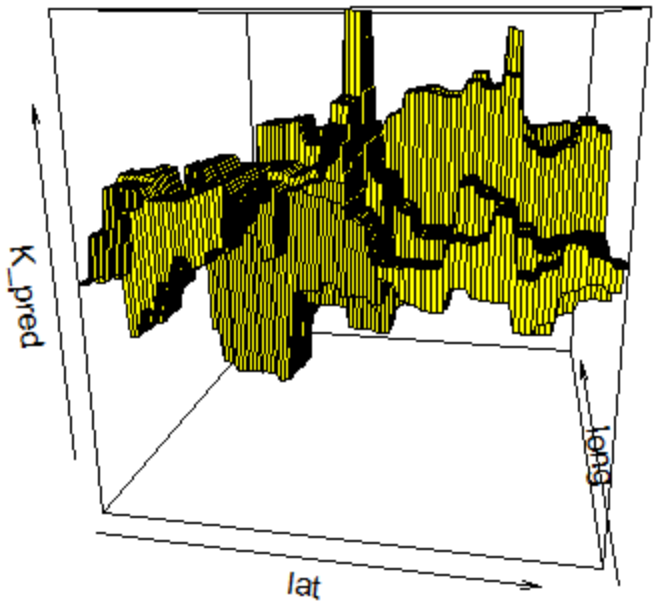
```r
numero_div <- 100
lat=seq(min(lat),max(lat),length=numero_div)
long=seq(min(long), max(long),length=numero_div)
datos_malla=expand.grid(long = long, lat=lat)


# prediccion de SO4 en la malla
K_pred =matrix(predict(RF_K, datos_malla),numero_div,numero_div)
head(K_pred,3)
```

```
##           [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]
## [1,] 0.4536924 0.4536924 0.4536924 0.4536924 0.4536924 0.4536924 0.4461692
## [2,] 0.4536924 0.4536924 0.4536924 0.4536924 0.4536924 0.4536924 0.4461692
## [3,] 0.4536924 0.4536924 0.4536924 0.4536924 0.4536924 0.4536924 0.4461692
##           [,8]      [,9]     [,10]     [,11]     [,12]     [,13]     [,14]
## [1,] 0.4461692 0.4461692 0.4461692 0.4461692 0.4461692 0.4414538 0.4414538
## [2,] 0.4461692 0.4461692 0.4461692 0.4461692 0.4461692 0.4414538 0.4414538
## [3,] 0.4461692 0.4461692 0.4461692 0.4461692 0.4461692 0.4414538 0.4414538
##          [,15]     [,16]     [,17]    [,18]    [,19]    [,20]    [,21]    [,22]
## [1,] 0.4414538 0.4414538 0.4414538 0.376602 0.376602 0.376602 0.376602 0.376602
## [2,] 0.4414538 0.4414538 0.4414538 0.376602 0.376602 0.376602 0.376602 0.376602
## [3,] 0.4414538 0.4414538 0.4414538 0.376602 0.376602 0.376602 0.376602 0.376602
##         [,23]     [,24]     [,25]     [,26]     [,27]     [,28]     [,29]
## [1,] 0.376602 0.3750837 0.3750837 0.3750837 0.3750837 0.3750837 0.3710279
## [2,] 0.376602 0.3750837 0.3750837 0.3750837 0.3750837 0.3750837 0.3710279
## [3,] 0.376602 0.3750837 0.3750837 0.3750837 0.3750837 0.3750837 0.3710279
##          [,30]     [,31]     [,32]     [,33]     [,34]     [,35]     [,36]
```

```
## [1,] 0.3710279 0.3710279 0.3710279 0.3710279 0.3710279 0.3719683 0.3719683
## [2,] 0.3710279 0.3710279 0.3710279 0.3710279 0.3710279 0.3719683 0.3719683
## [3,] 0.3710279 0.3710279 0.3710279 0.3710279 0.3710279 0.3719683 0.3719683
##          [,37]     [,38]     [,39]     [,40]     [,41]     [,42]     [,43]
## [1,] 0.3719683 0.3719683 0.3719683 0.3972987 0.3972987 0.3972987 0.3972987
## [2,] 0.3719683 0.3719683 0.3719683 0.3972987 0.3972987 0.3972987 0.3972987
## [3,] 0.3719683 0.3719683 0.3719683 0.3972987 0.3972987 0.3972987 0.3972987
##          [,44]     [,45]     [,46]     [,47]     [,48]     [,49]     [,50]
## [1,] 0.3972987 0.3972987 0.3979377 0.3979377 0.3979377 0.3979377 0.3979377
## [2,] 0.3972987 0.3972987 0.3979377 0.3979377 0.3979377 0.3979377 0.3979377
## [3,] 0.3972987 0.3972987 0.3979377 0.3979377 0.3979377 0.3979377 0.3979377
##          [,51]     [,52]     [,53]     [,54]     [,55]     [,56]    [,57]
## [1,] 0.3980949 0.3980949 0.3980949 0.3980949 0.3980949 0.3980949 0.397062
## [2,] 0.3980949 0.3980949 0.3980949 0.3980949 0.3980949 0.3980949 0.397062
## [3,] 0.3980949 0.3980949 0.3980949 0.3980949 0.3980949 0.3980949 0.397062
##         [,58]    [,59]    [,60]    [,61]     [,62]     [,63]     [,64]
## [1,] 0.397062 0.397062 0.397062 0.397062 0.3057132 0.3057132 0.3057132
## [2,] 0.397062 0.397062 0.397062 0.397062 0.3057132 0.3057132 0.3057132
## [3,] 0.397062 0.397062 0.397062 0.397062 0.3057132 0.3057132 0.3057132
##          [,65]     [,66]     [,67]    [,68]    [,69]    [,70]    [,71]    [,72]
## [1,] 0.3057132 0.3057132 0.3057132 0.302315 0.302315 0.302315 0.302315 0.302315
## [2,] 0.3057132 0.3057132 0.3057132 0.302315 0.302315 0.302315 0.302315 0.302315
## [3,] 0.3057132 0.3057132 0.3057132 0.302315 0.302315 0.302315 0.302315 0.302315
##          [,73]     [,74]     [,75]     [,76]     [,77]     [,78]     [,79]
## [1,] 0.2722554 0.2722554 0.2722554 0.2722554 0.2722554 0.2722554 0.2700268
## [2,] 0.2722554 0.2722554 0.2722554 0.2722554 0.2722554 0.2722554 0.2700268
## [3,] 0.2722554 0.2722554 0.2722554 0.2722554 0.2722554 0.2722554 0.2700268
##          [,80]     [,81]     [,82]     [,83]     [,84]     [,85]     [,86]
## [1,] 0.2700268 0.2700268 0.2700268 0.2700268 0.2814333 0.2814333 0.2814333
## [2,] 0.2700268 0.2700268 0.2700268 0.2700268 0.2814333 0.2814333 0.2814333
## [3,] 0.2700268 0.2700268 0.2700268 0.2700268 0.2814333 0.2814333 0.2814333
##          [,87]     [,88]     [,89]     [,90]     [,91]     [,92]     [,93]
## [1,] 0.2814333 0.2814333 0.2814333 0.2820099 0.2820099 0.2820099 0.2820099
## [2,] 0.2814333 0.2814333 0.2814333 0.2820099 0.2820099 0.2820099 0.2820099
```

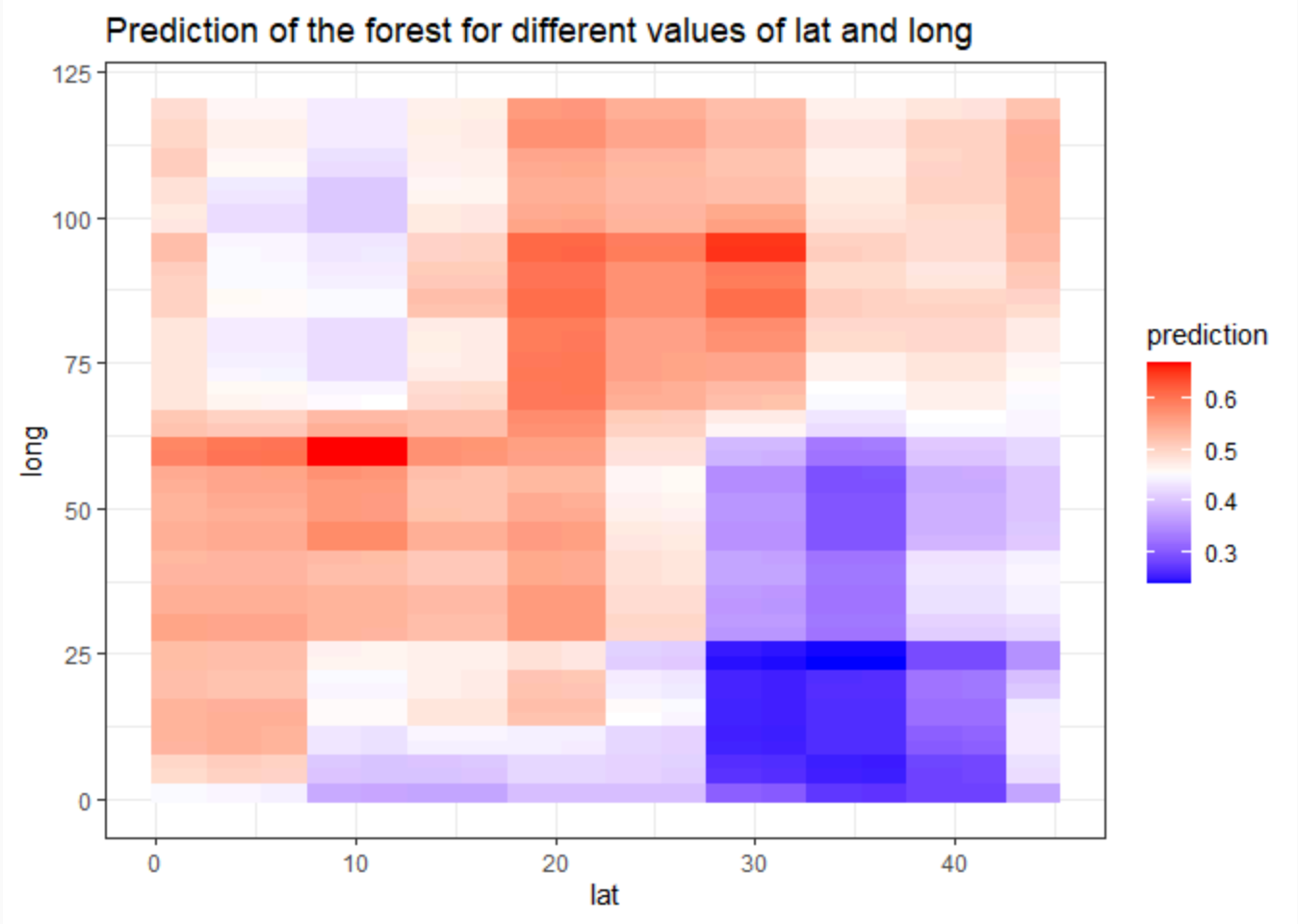
```
## [3,] 0.2814333 0.2814333 0.2814333 0.2820099 0.2820099 0.2820099 0.2820099
##          [,94]     [,95]     [,96]     [,97]     [,98]     [,99]    [,100]
## [1,] 0.2820099 0.3719564 0.3719564 0.3719564 0.3719564 0.3719564 0.3719564
## [2,] 0.2820099 0.3719564 0.3719564 0.3719564 0.3719564 0.3719564 0.3719564
## [3,] 0.2820099 0.3719564 0.3719564 0.3719564 0.3719564 0.3719564 0.3719564
```

```
p <- persp(lat, long, K_pred, theta = 10, col = "yellow")
```




```
library(randomForestExplainer)
```

```
plot_predict_interaction(RF_K, datos_malla,"lat", "long")
```

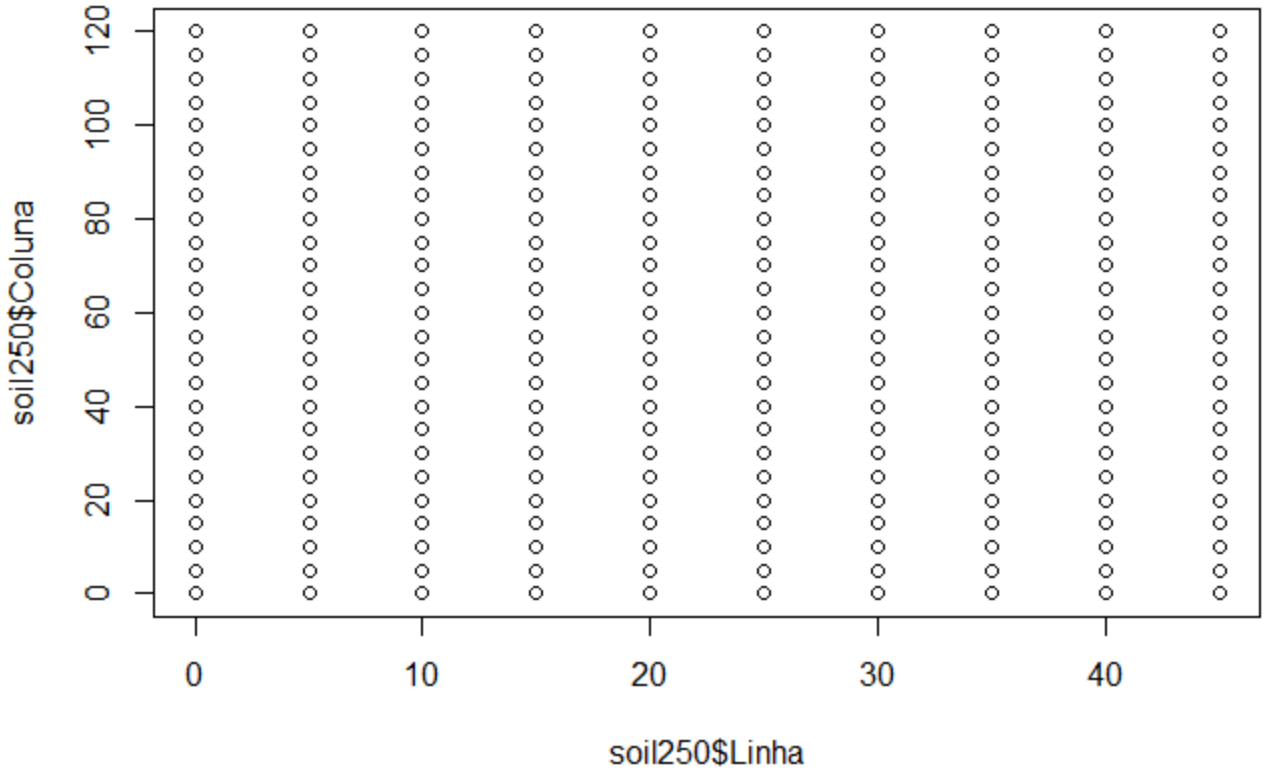



```
data(soil250)
head(soil250)
```

```
##   Linha Coluna    Cota AGrossa Silte Argila pHAgua pHKCl  Ca  Mg    K Al   H
## 1     0      0 578.295       9    26     43    5.8   4.9 3.6 0.8 0.50  0 3.1
## 2     0      5 578.460       9    26     42    5.9   4.9 3.4 0.8 0.44  0 3.2
## 3     0     10 578.491       9    25     41    5.9   4.9 3.7 0.9 0.59  0 2.5
## 4     0     15 578.699      11    28     40    5.8   4.9 3.7 0.8 0.52  0 3.5
```

```
## 5     0     20 578.749       9    27     41    5.8   5.0 4.2 0.9 0.56  0 3.4
## 6     0     25 578.726       8    27     43    6.1   5.1 4.1 0.9 0.56  0 3.1
##     C    N  CTC    S      V M    NC  CEC CN
## 1 1.2 0.12 8.00 4.90 61.250 0 1.050 4.90 10
## 2 1.1 0.12 7.84 4.64 59.184 0 1.272 4.64  9
## 3 1.2 0.13 7.69 5.19 67.490 0 0.289 5.19  9
## 4 1.3 0.12 8.52 5.02 58.920 0 1.416 5.02 10
## 5 1.4 0.13 9.06 5.66 62.472 0 1.023 5.66 10
## 6 1.2 0.13 8.66 5.56 64.203 0 0.753 5.56  9
```

```
plot(soil250$Linha, soil250$Coluna)
```




```
lat=soil250$Linha
long=soil250$Coluna
K=soil250$K
RF_K2 <- lm(K~lat+long)
summary(RF_K2)
```

```
##
## Call:
## lm(formula = K ~ lat + long)
##
```

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.23754 -0.06957  0.00513  0.06724  0.55213
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.4535035  0.0170593  26.584  < 2e-16 ***
## lat         -0.0022361  0.0004761  -4.696 4.40e-06 ***
## long         0.0011122  0.0001897   5.864 1.44e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1081 on 247 degrees of freedom
## Multiple R-squared:  0.186,  Adjusted R-squared:  0.1794
## F-statistic: 28.22 on 2 and 247 DF,  p-value: 9.151e-12
```

// Ejemplo de Fertilidad de Suelos

Datos de África occidental sobre la fertilidad del suelo y la respuesta de los cultivos a los fertilizantes. Estos datos son parte de un estudio más amplio que se describe en un artículo de próxima publicación (Bonilla et al.). Hijmans, R.J., 2019. Statistical modeling. In: Hijmans, R.J. and J. Chamberlin. Regional Agronomy: a pratical handbook. CIMMYT. https:/reagro.org/tools/statistical/

Estas son las variables que tenemos.

emp	Average temperature
precip	Annual precipitation
ExchP	Soil exchangeble P
TotK	Soil total K

emp	Average temperature
ExchAl	Soil exchangeble Al
TotN	Soil total N
sand	Soil franction sand (%)
clay	Soil fraction clay (%)
SOC	Soil organic carbon (g/kg)
pH	Soil pH
AWC	Soil water holding capacity
fert	fertilizer (index) kg/ha

```
#remotes::install_github("reagro/agrodata")
#remotes::install_github("reagro/agro")
library(agrodata)
library(agro)
library(randomForest)
library(rpart)
datos_fert=reagro_data("soilfert")
pander::pander(head(datos_fert))
```

Table continues below

	temp	precip	ExchP	TotK	ExchAl	TotN	sand	clay	SOC
1463	27	1260	933	120	610	1157	65	17	13.5

	temp	precip	ExchP	TotK	ExchAl	TotN	sand	clay	SOC
1464	27	1260	933	120	610	1157	65	17	13.5
1465	27	1260	933	120	610	1157	65	17	13.5
1466	27	1260	933	120	610	1157	65	17	13.5
1467	27	1260	933	120	610	1157	65	17	13.5
1468	27	1260	933	120	610	1157	65	17	13.5

	pH	AWC	fert
1463	6.3	26	120
1464	6.3	26	120
1465	6.3	26	120
1466	6.3	26	120
1467	6.3	26	120
1468	6.3	26	120

```
model <- SOC~pH+precip+temp+sand+clay
lrm <- lm(model, data=datos_fert)
summary(lrm)
```

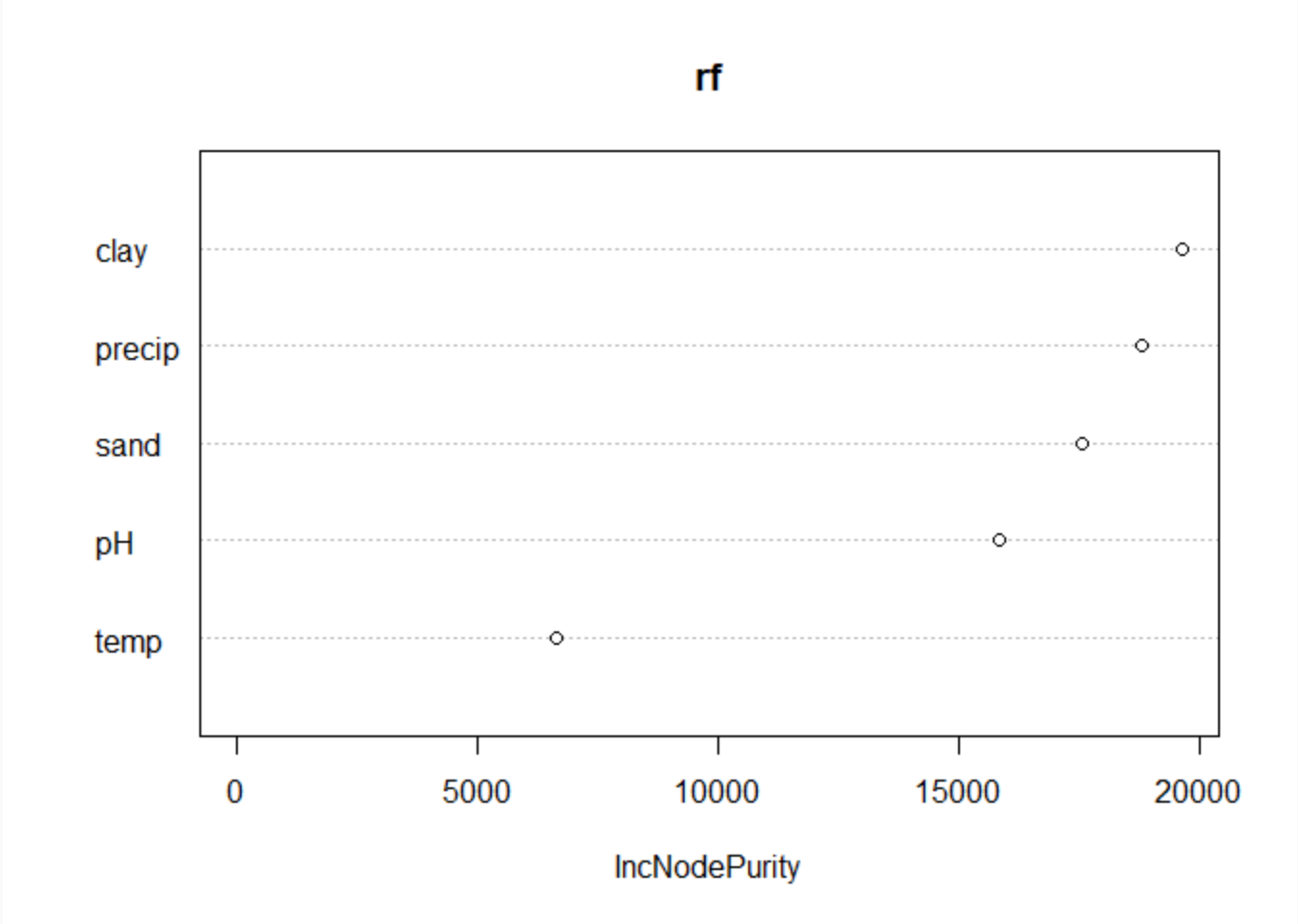
```
##
## Call:
## lm(formula = model, data = datos_fert)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.281  -2.713  -1.125   2.054  27.067
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -7.6781312  4.0759621  -1.884   0.0598 .
## pH          -1.8263969  0.3761195  -4.856 1.31e-06 ***
## precip       0.0052787  0.0003756  14.053  < 2e-16 ***
## temp        -0.0684204  0.1128496  -0.606   0.5444
## sand         0.1741979  0.0236394   7.369 2.69e-13 ***
## clay         0.7917492  0.0333736  23.724  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.733 on 1678 degrees of freedom
## Multiple R-squared:  0.5379, Adjusted R-squared:  0.5365
## F-statistic: 390.7 on 5 and 1678 DF,  p-value: < 2.2e-16
```

```
library(randomForest)
rf <- randomForest(model, data=datos_fert)
rf
```

```
##
## Call:
##  randomForest(formula = model, data = datos_fert)
```

```
##               Type of random forest: regression
##                     Number of trees: 500
## No. of variables tried at each split: 1
##
##           Mean of squared residuals: 2.572302
##                     % Var explained: 94.68
```

```
varImpPlot(rf)
```



```r
library(randomForestExplainer)
forest1 <- randomForest(model, data = datos_fert, localImp = TRUE)
#measure_importance(forest1)
#frame <- measure_importance(forest1, measures = c("mean_min_depth",
"times_a_root"))
#plot_importance_ggpairs(frame, measures = c("mean_min_depth", "times_a_root"))

#plot_multi_way_importance(measure_importance(forest1))
#plot_min_depth_distribution(min_depth_distribution(forest1))
```

// Seleccion de variables en random forest

```r
library(randomForestExplainer)
forest1 <- randomForest(model, data = datos_fert, localImp = TRUE)
set.seed(1234)
library(VSURF)
vozone <- VSURF(model, data = datos_fert)
```

```
## Thresholding step
## Estimated computational time (on one core): 20 sec.
##
  |
  |                                                                      |   0%
  |
  |====                                                                  |   5%
  |
  |=======                                                               |  10%
  |
  |==========                                                            |  15%
```


|
|============== | 20%
|
|================== | 25%
|
|====================== | 30%
|
|========================= | 35%
|
|============================ | 40%
|
|================================ | 45%
|
|================================== | 50%
|
|====================================== | 55%
|
|== | 60%
|
|== | 65%
|
|== | 70%
|
|== | 75%
|
|== | 80%
|
|== | 85%
|
|==| 90%
|
|==| 95%
|
|==| 100%

```
## Interpretation step (on 5 variables)
## Estimated computational time (on one core): between 8 sec. and  7.5 sec.
##
  |
  |                                                                      |   0%
  |
  |==============                                                        |  20%
  |
  |============================                                          |  40%
  |
  |==========================================                            |  60%
  |
  |========================================================              |  80%
  |
  |======================================================================| 100%
## Prediction step (on 4 variables)
## Maximum estimated computational time (on one core): 6 sec.
##
  |
  |                                                                      |   0%
  |
  |==================                                                    |  25%
  |
  |==================================                                    |  50%
  |
  |================================================                      |  75%
  |
  |======================================================================| 100%
```

```
summary(vozone)
```

```
##
##  VSURF computation time: 41.5 secs
##
##  VSURF selected:
##  5 variables at thresholding step (in 25.3 secs)
##  4 variables at interpretation step (in 9 secs)
##  3 variables at prediction step (in 7.2 secs)
```