Código Wankara

Luis Balderas Ruiz

```r
###########################################################
# INTRODUCCIÓN A LA CIENCIA DE DATOS
# Autor: Luis Balderas Ruiz
# EDA+Regresion
# Dataset: wankara (01/01/1994 to 28/05/1998)
############################################################

library(ggplot2)
library(tidyverse)


binwd = function(data){
  size = length(data)
  dt = sd(data)
  cr = size^(1/3)
  return(1/(cr)*dt*3.49)
}
wankara = read.csv("./data/wankara/wankara.dat",header=FALSE, comment.char = "@")
colnames(wankara) = c("Max_temperature", "Min_temperature",
  "Dewpoint", "Precipitation", "Sea_level_pressure", "Standard_pressure",
  "Visibility", "Wind_speed", "Max_wind_speed","Mean_temperature")


# Resumen estadístico
summary(wankara)

# Visualización de las variables respecto de mean_temperature
temp <- wankara
plotY <- function (x,y) {
  plot(temp[,y]~temp[,x], xlab=paste(names(temp)[x]),
       ylab=names(temp)[y])
}
par(mfrow=c(3,4)) #Si margin too large => (2,3)
x <- sapply(1:(dim(temp)[2]-1), plotY, dim(temp)[2])
par(mfrow=c(1,1))


##########################################################################################
# HISTOGRAMAS

library(e1071)
# Max-temperature
skewness(wankara$Max_temperature)
kurtosis(wankara$Max_temperature)
ggplot(data=wankara, aes(x=Max_temperature)) +
  geom_histogram(binwidth = binwd(wankara$Max_temperature),fill="blue") +
  ggtitle("Histograma de temperatura máxima") +
  labs(x="Temperatura máxima", y="Count\nof Records")
```

```r
# Min-temperature
skewness(wankara$Min_temperature)
kurtosis(wankara$Min_temperature)
ggplot(data=wankara, aes(x=Min_temperature)) +
  geom_histogram(binwidth = binwd(wankara$Min_temperature),fill="blue") +
  ggtitle("Histograma de temperatura mínima") +
  labs(x="Temperatura mínima", y="Count\nof Records")

# Dewpoint
skewness(wankara$Dewpoint)
kurtosis(wankara$Dewpoint)
ggplot(data=wankara, aes(x=Dewpoint)) +
  geom_histogram(binwidth = binwd(wankara$Dewpoint),fill="blue") +
  ggtitle("Histograma Dewpoint") +
  labs(x="Dewpoint", y="Count\nof Records")

# Precipitation
skewness(wankara$Precipitation)
kurtosis(wankara$Precipitation)
ggplot(data=wankara, aes(x=Precipitation)) +
  geom_histogram(binwidth = binwd(wankara$Precipitation),fill="blue") +
  ggtitle("Histograma Precipitaciones") +
  labs(x="Precipitaciones", y="Count\nof Records")

# Sea level pressure
skewness(wankara$Sea_level_pressure)
kurtosis(wankara$Sea_level_pressure)
ggplot(data=wankara, aes(x=Sea_level_pressure)) +
  geom_histogram(binwidth = binwd(wankara$Sea_level_pressure),fill="blue") +
  ggtitle("Histograma Sea_level_pressure") +
  labs(x="Sea_level_pressure", y="Count\nof Records")

# Standard pressure
skewness(wankara$Standard_pressure)
kurtosis(wankara$Standard_pressure)
ggplot(data=wankara, aes(x=Standard_pressure)) +
  geom_histogram(binwidth = binwd(wankara$Standard_pressure),fill="blue") +
  ggtitle("Histograma Standard pressure") +
  labs(x="Standard pressure", y="Count\nof Records")

# Visibility
skewness(wankara$Visibility)
kurtosis(wankara$Visibility)
ggplot(data=wankara, aes(x=Visibility)) +
  geom_histogram(binwidth = binwd(wankara$Visibility),fill="blue") +
  ggtitle("Histograma Visibility") +
  labs(x="Visibility", y="Count\nof Records")

# Wind speed
skewness(wankara$Wind_speed)
kurtosis(wankara$Wind_speed)
ggplot(data=wankara, aes(x=Wind_speed)) +
  geom_histogram(binwidth = binwd(wankara$Wind_speed),fill="blue") +
```

```
  ggtitle("Histograma Wind speed") +
  labs(x="Wind speed", y="Count\nof Records")

# Max Wind speed
skewness(wankara$Max_wind_speed)
kurtosis(wankara$Max_wind_speed)
ggplot(data=wankara, aes(x=Max_wind_speed)) +
  geom_histogram(binwidth = binwd(wankara$Max_wind_speed),fill="blue") +
  ggtitle("Histograma Max Wind speed") +
  labs(x="Max Wind speed", y="Count\nof Records")

# Histograma temperatura media
skewness(wankara$Mean_temperature)
kurtosis(wankara$Mean_temperature)
ggplot(data=wankara, aes(x=Mean_temperature)) +
  geom_histogram(binwidth=binwd(wankara$Mean_temperature),fill="blue") +
  ggtitle("Histograma de temperatura media") +
  labs(x="Temperatura média", y="Count\nof Records")


###############################################################################################
# VALORES PERDIDOS

wankara[is.na(wankara)]

###############################################################################################
# OUTLIERS
install.packages("outliers")
library(outliers)

sapply(wankara,outlier)
sapply(wankara,outlier,opposite=TRUE)


###############################################################################################
# DIVISIÓN POR MESES

# Días en cada mes
dias_mes = c(31,28,31,30,31,30,31,31,30,31,30,31)
# Días del dataset
dias = rep(dias_mes,5)[1:53]
# 1996 fue bisiesto
dias[12*3+2] = 29

max_temp = c()
min_temp = c()
dewp = c()
precip = c()
slp = c()
sp = c()
visib = c()
Ws = c()
Msp = c()
```

```r
Mean_temp = c()
actual = 1
for(i in 1:53){
  if(i == 53){
    max_temp = append(max_temp, mean(wankara$Max_temperature[actual:1609]))
    min_temp = append(min_temp,mean(wankara$Min_temperature[actual:1609]))
    dewp = append(dewp, mean(wankara$Dewpoint[actual:1609]))
    precip = append(precip, mean(wankara$Precipitation[actual:1609]))
    slp = append(slp,mean(wankara$Sea_level_pressure[actual:1609]))
    sp = append(sp, mean(wankara$Standard_pressure[actual:1609]))
    visib = append(visib, mean(wankara$Visibility[actual:1609]))
    Ws = append(Ws, mean(wankara$Wind_speed[actual:1609]))
    Msp = append(Msp, mean(wankara$Max_wind_speed[actual:1609]))
    Mean_temp = append(Mean_temp, mean(wankara$Mean_temperature[actual:1609]))
  }
  else{
    max_temp = append(max_temp, mean(wankara$Max_temperature[actual:actual+dias[i]-1]))
    min_temp = append(min_temp,mean(wankara$Min_temperature[actual:actual+dias[i]-1]))
    dewp = append(dewp, mean(wankara$Dewpoint[actual:actual+dias[i]-1]))
    precip = append(precip, mean(wankara$Precipitation[actual:actual+dias[i]-1]))
    slp = append(slp,mean(wankara$Sea_level_pressure[actual:actual+dias[i]-1]))
    sp = append(sp, mean(wankara$Standard_pressure[actual:actual+dias[i]-1]))
    visib = append(visib, mean(wankara$Visibility[actual:actual+dias[i]-1]))
    Ws = append(Ws, mean(wankara$Wind_speed[actual:actual+dias[i]-1]))
    Msp = append(Msp, mean(wankara$Max_wind_speed[actual:actual+dias[i]-1]))
    Mean_temp = append(Mean_temp, mean(wankara$Mean_temperature[actual:actual+dias[i]-1]))
    actual = actual+dias[i]
    print(actual)
  }
}
meses = rep(month.abb,5)[1:53]
j=94
for(i in 1:53){
  if(i%%12 == 1 && i > 12){
    j = j+1
  }
  meses[i] = paste(meses[i],j,sep="")
}
df = as.data.frame(cbind(meses, max_temp,min_temp))
df$meses = factor(df$meses,levels=df$meses)
ggplot(df[1:12,],aes(x=meses, y=max_temp))+geom_point()
ggplot(df[1:12,],aes(x=meses,y=min_temp)) + geom_point()
# Los datos no parecen tener un orden cronológico

###########################################################################################
# Normalidad
library(nortest)
library(car)

# Ninguna variable es normal
sapply(wankara,shapiro.test)
sapply(wankara,lillie.test)
sapply(wankara,qqPlot)
```

```
###############################################################################################
# HIPÓTESIS

# Hipótesis: mayor temperatura máxima, mayor temperatura media
ggplot(data=wankara, aes(x=wankara$Max_temperature, y=wankara$Mean_temperature)) +
  geom_point(alpha=.4, size=4, color="#880011") +
  ggtitle("Temperatura máxima vs Temperatura media") +
  labs(x="Temperatura máxima", y="Temperatura media")

# Hipótesis: mayor temperatura mínima, mayor temperatura media
ggplot(data=wankara, aes(x=wankara$Min_temperature, y=wankara$Mean_temperature)) +
  geom_point(alpha=.4, size=4, color="#880011") +
  ggtitle("Temperatura mínima vs Temperatura media") +
  labs(x="Temperatura mínima", y="Temperatura media")


#############################################################################################
# CORRELACIÓN

install.packages("PerformanceAnalytics")
library("PerformanceAnalytics")
chart.Correlation(wankara, histogram=TRUE, pch=19)

###############################################################################################3
# REESCALADO

library("scales")
wankara_scale = sapply(wankara,rescale)
wankara_scale = as.data.frame(wankara_scale)
summary(wankara_scale)

########################################################################
# R.1
# Modelo lineal simple con la variable con más correlación: Max_temperature
fit_mls1 = lm(wankara_scale$Mean_temperature~wankara_scale$Max_temperature)
summary(fit_mls1)

par(mfrow=c(1,1))
plot(wankara_scale$Mean_temperature~wankara_scale$Max_temperature)
abline(fit_mls1,col="red")
confint(fit_mls1)

# Error cuadrático medio
yprime=predict(fit_mls1,data.frame(Max_temp=wankara_scale$Max_temperature))
sqrt(sum(abs(wankara_scale$Mean_temperature-yprime)^2)/length(yprime))

# Cross-validation

nombre <- "./data/wankara/wankara"
run_lm1_fold <- function(i, x, tt = "test") {
  file <- paste(x, "-5-", i, "tra.dat", sep="")
  x_tra <- read.csv(file, comment.char="@", header=FALSE)
  file <- paste(x, "-5-", i, "tst.dat", sep="")
```

```
  x_tst <- read.csv(file, comment.char="@", header=FALSE)
  In <- length(names(x_tra)) - 1
  names(x_tra)[1:In] <- paste ("X", 1:In, sep="")
  names(x_tra)[In+1] <- "Y"
  names(x_tst)[1:In] <- paste ("X", 1:In, sep="")
  names(x_tst)[In+1] <- "Y"
  if (tt == "train") {
    test <- x_tra
  }
  else {
    test <- x_tst
  }
  fitMulti=lm(Y~X1,x_tra)
  yprime=predict(fitMulti,test)
  sum(abs(test$Y-yprime)^2)/length(yprime) ##MSE
}
resultados_mls1_train = sapply(1:5,run_lm1_fold,nombre,"train")
resultados_mls1_test = sapply(1:5,run_lm1_fold,nombre,"test")
lmMSEtrain1<-mean(resultados_mls1_train)
lmMSEtest1<-mean(resultados_mls1_test)


# Modelo lineal simple con la variable con más correlación: Min_temperature
fit_mls2 = lm(wankara_scale$Mean_temperature~wankara_scale$Min_temperature)
summary(fit_mls2)

par(mfrow=c(1,1))
plot(wankara_scale$Mean_temperature~wankara_scale$Min_temperature)
abline(fit_mls2,col="red")
confint(fit_mls2)

# Error cuadrático medio
yprime=predict(fit_mls2,data.frame(Max_temp=wankara_scale$Min_temperature))
sqrt(sum(abs(wankara_scale$Mean_temperature-yprime)^2)/length(yprime))

# Cross-validation

nombre <- "./data/wankara/wankara"
run_lm2_fold <- function(i, x, tt = "test") {
  file <- paste(x, "-5-", i, "tra.dat", sep="")
  x_tra <- read.csv(file, comment.char="@", header=FALSE)
  file <- paste(x, "-5-", i, "tst.dat", sep="")
  x_tst <- read.csv(file, comment.char="@", header=FALSE)
  In <- length(names(x_tra)) - 1
  names(x_tra)[1:In] <- paste ("X", 1:In, sep="")
  names(x_tra)[In+1] <- "Y"
  names(x_tst)[1:In] <- paste ("X", 1:In, sep="")
  names(x_tst)[In+1] <- "Y"
  if (tt == "train") {
    test <- x_tra
  }
  else {
    test <- x_tst
```

```r
  }
  fitMulti=lm(Y~X2,x_tra)
  yprime=predict(fitMulti,test)
  sum(abs(test$Y-yprime)^2)/length(yprime) ##MSE
}
resultados_mls2_train = sapply(1:5,run_lm2_fold,nombre,"train")
resultados_mls2_test = sapply(1:5,run_lm2_fold,nombre,"test")
lmMSEtrain2<-mean(resultados_mls2_train)
lmMSEtest2<-mean(resultados_mls2_test)

# Dewpoint
fit_mls3 = lm(wankara_scale$Mean_temperature~wankara_scale$Dewpoint)
summary(fit_mls3)

par(mfrow=c(1,1))
plot(wankara_scale$Mean_temperature~wankara_scale$Dewpoint)
abline(fit_mls3,col="red")
confint(fit_mls3)

# Error cuadrático medio
yprime=predict(fit_mls3,data.frame(Max_temp=wankara_scale$Dewpoint))
sqrt(sum(abs(wankara_scale$Mean_temperature-yprime)^2)/length(yprime))

# Cross-validation

nombre <- "./data/wankara/wankara"
run_lm3_fold <- function(i, x, tt = "test") {
  file <- paste(x, "-5-", i, "tra.dat", sep="")
  x_tra <- read.csv(file, comment.char="@", header=FALSE)
  file <- paste(x, "-5-", i, "tst.dat", sep="")
  x_tst <- read.csv(file, comment.char="@", header=FALSE)
  In <- length(names(x_tra)) - 1
  names(x_tra)[1:In] <- paste ("X", 1:In, sep="")
  names(x_tra)[In+1] <- "Y"
  names(x_tst)[1:In] <- paste ("X", 1:In, sep="")
  names(x_tst)[In+1] <- "Y"
  if (tt == "train") {
    test <- x_tra
  }
  else {
    test <- x_tst
  }
  fitMulti=lm(Y~X3,x_tra)
  yprime=predict(fitMulti,test)
  sum(abs(test$Y-yprime)^2)/length(yprime) ##MSE
}
resultados_mls3_train = sapply(1:5,run_lm3_fold,nombre,"train")
resultados_mls3_test = sapply(1:5,run_lm3_fold,nombre,"test")
lmMSEtrain3<-mean(resultados_mls3_train)
lmMSEtest3<-mean(resultados_mls3_test)


# Sea_level_pressure
```

```r
fit_mls4 = lm(wankara_scale$Mean_temperature~wankara_scale$Sea_level_pressure)
summary(fit_mls4)

par(mfrow=c(1,1))
plot(wankara_scale$Mean_temperature~wankara_scale$Sea_level_pressure)
abline(fit_mls4,col="red")
confint(fit_mls4)

# Error cuadrático medio
yprime=predict(fit_mls4,data.frame(SLP=wankara_scale$Sea_level_pressure))
sqrt(sum(abs(wankara_scale$Mean_temperature-yprime)^2)/length(yprime))

# Cross-validation

nombre <- "./data/wankara/wankara"
run_lm4_fold <- function(i, x, tt = "test") {
  file <- paste(x, "-5-", i, "tra.dat", sep="")
  x_tra <- read.csv(file, comment.char="@", header=FALSE)
  file <- paste(x, "-5-", i, "tst.dat", sep="")
  x_tst <- read.csv(file, comment.char="@", header=FALSE)
  In <- length(names(x_tra)) - 1
  names(x_tra)[1:In] <- paste ("X", 1:In, sep="")
  names(x_tra)[In+1] <- "Y"
  names(x_tst)[1:In] <- paste ("X", 1:In, sep="")
  names(x_tst)[In+1] <- "Y"
  if (tt == "train") {
    test <- x_tra
  }
  else {
    test <- x_tst
  }
  fitMulti=lm(Y~X5,x_tra)
  yprime=predict(fitMulti,test)
  sum(abs(test$Y-yprime)^2)/length(yprime) ##MSE
}
resultados_mls4_train = sapply(1:5,run_lm4_fold,nombre,"train")
resultados_mls4_test = sapply(1:5,run_lm4_fold,nombre,"test")
lmMSEtrain4<-mean(resultados_mls4_train)
lmMSEtest4<-mean(resultados_mls4_test)

# Visibility
fit_mls5 = lm(wankara_scale$Mean_temperature~wankara_scale$Visibility)
summary(fit_mls5)

par(mfrow=c(1,1))
plot(wankara_scale$Mean_temperature~wankara_scale$Visibility)
abline(fit_mls5,col="red")
confint(fit_mls5)

# Error cuadrático medio
yprime=predict(fit_mls5,data.frame(Vis=wankara_scale$Visibility))
sqrt(sum(abs(wankara_scale$Mean_temperature-yprime)^2)/length(yprime))
```

```r
# Cross-validation

nombre <- "./data/wankara/wankara"
run_lm5_fold <- function(i, x, tt = "test") {
  file <- paste(x, "-5-", i, "tra.dat", sep="")
  x_tra <- read.csv(file, comment.char="@", header=FALSE)
  file <- paste(x, "-5-", i, "tst.dat", sep="")
  x_tst <- read.csv(file, comment.char="@", header=FALSE)
  In <- length(names(x_tra)) - 1
  names(x_tra)[1:In] <- paste ("X", 1:In, sep="")
  names(x_tra)[In+1] <- "Y"
  names(x_tst)[1:In] <- paste ("X", 1:In, sep="")
  names(x_tst)[In+1] <- "Y"
  if (tt == "train") {
    test <- x_tra
  }
  else {
    test <- x_tst
  }
  fitMulti=lm(Y~X7,x_tra)
  yprime=predict(fitMulti,test)
  sum(abs(test$Y-yprime)^2)/length(yprime) ##MSE
}
resultados_mls5_train = sapply(1:5,run_lm5_fold,nombre,"train")
resultados_mls5_test = sapply(1:5,run_lm5_fold,nombre,"test")
lmMSEtrain5<-mean(resultados_mls5_train)
lmMSEtest5<-mean(resultados_mls5_test)

#####################################################################################
# R.2
# MODELO LINEAL MÚLTIPLE
# BACKWARD MODEL


# Elimino Precipitation por tener un p-valor de 0.885
fit_mlm2=lm(wankara_scale$Mean_temperature~.
              -Precipitation,data=wankara_scale)
summary(fit_mlm2)

# Elimino Sea_level_pressure por tener el mayor error standard
fit_mlm3=lm(wankara_scale$Mean_temperature~.-Precipitation
            -Sea_level_pressure,data=wankara_scale)
summary(fit_mlm3)

# Elimino Max wind speed por tener el mayor error standard
fit_mlm4 = lm(wankara_scale$Mean_temperature~.-Precipitation
              -Sea_level_pressure-Max_wind_speed,data=wankara_scale)
summary(fit_mlm4)

# Elimino visibility
fit_mlm5 = lm(wankara_scale$Mean_temperature~.-Precipitation
              -Sea_level_pressure-Max_wind_speed-Visibility,data=wankara_scale)
summary(fit_mlm5)
```

```r
# Elimino standard pressure
fit_mlm6 = lm(wankara_scale$Mean_temperature~.-Standard_pressure-Precipitation
              -Sea_level_pressure-Max_wind_speed-Visibility,data=wankara_scale)
summary(fit_mlm6)

# Elimino Wind speed --> Modelo más interpretable
fit_mlm7 = lm(wankara_scale$Mean_temperature~.-Wind_speed-Standard_pressure
                -Precipitation-Sea_level_pressure-Max_wind_speed-Visibility,data=wankara_scale)
summary(fit_mlm7)

# INTERACCIONES

# Interacción entre la presión a nivel del mar y la estándar. Mejor resultado hasta ahora: 0.9899
fit_i1=lm(wankara_scale$Mean_temperature~.-Precipitation
          +Sea_level_pressure*Standard_pressure,data=wankara_scale)
summary(fit_i1)

# Interacción entre la temperatura mínima y dewpoint
fit_i2=lm(wankara_scale$Mean_temperature~.-Precipitation
          +Min_temperature*Dewpoint,data=wankara_scale)
summary(fit_i2)

# Mejor resultado hasta el momento 0.99
fit_i3 = lm(wankara_scale$Mean_temperature~.-Precipitation
            +Min_temperature*Dewpoint+I(Dewpoint^2)
            -Dewpoint,data=wankara_scale)
summary(fit_i3)

# Eliminamos Visibility por su alto p-value
fit_i4 = lm(wankara_scale$Mean_temperature~.-Precipitation
            +Min_temperature*Dewpoint+I(Dewpoint^2)-Dewpoint
            -Visibility,data=wankara_scale)
summary(fit_i4)

# Mejor resultado hasta el momento, 0.9916
fit_i5 = lm(wankara_scale$Mean_temperature~.-Precipitation
            +I(Max_temperature^2)+Min_temperature*Dewpoint
            +I(Dewpoint^2)-Dewpoint,data=wankara_scale)
summary(fit_i5)

# Mejor resultado --> 0.9923
fit_i6 = lm(wankara_scale$Mean_temperature~.-Precipitation
            +I(Min_temperature^2)+I(Max_temperature^2)
            +Min_temperature*Dewpoint+I(Dewpoint^2)
            -Dewpoint,data=wankara_scale)
summary(fit_i6)

# 0.9923
fit_i7 = lm(wankara_scale$Mean_temperature~.-Precipitation
            +Max_wind_speed*Wind_speed+I(Min_temperature^2)
            +I(Max_temperature^2)+Min_temperature*Dewpoint
            +I(Dewpoint^2)-Dewpoint-Max_wind_speed,data=wankara_scale)
summary(fit_i7)
```

```r
# Cálculo de MSE
yprime_i7 = predict(fit_i7,wankara_scale)
sqrt(sum(abs(wankara_scale$Mean_temperature-yprime_i7)^2)/length(yprime_i7))

# CV del model más satisfactorio
nombre <- "./data/wankara/wankara"
run_i7_fold <- function(i, x, tt = "test") {
  file <- paste(x, "-5-", i, "tra.dat", sep="")
  x_tra <- read.csv(file, comment.char="@"
                , header=FALSE)
  file <- paste(x, "-5-", i, "tst.dat", sep="")
  x_tst <- read.csv(file, comment.char="@"
                 , header=FALSE)
  In <- length(names(x_tra)) - 1
  names(x_tra)[1:In] <- paste ("X", 1:In, sep="")
  names(x_tra)[In+1] <- "Y"
  names(x_tst)[1:In] <- paste ("X", 1:In, sep="")
  names(x_tst)[In+1] <- "Y"
  if (tt == "train") {
    test <- x_tra
  }
  else {
    test <- x_tst
  }
  fit_i7 = lm(Y~.-X4+X9*X8+I(X2^2)+I(X1^2)+X2*X3+I(X3^2)-X3-X7-X9,data=test)
  yprime=predict(fit_i7,test)
  sum(abs(test$Y-yprime)^2)/length(yprime) ##MSE
}
resultados_i7_train = sapply(1:5,run_i7_fold,nombre,"train")
resultados_i7_test = sapply(1:5,run_i7_fold,nombre,"test")
i7MSEtrain<-mean(resultados_i7_train)
i7MSEtest<-mean(resultados_i7_test)

# Resultados de las interacción 7 (mejor resultado)
plot(wankara_scale$Mean_temperature~wankara_scale$Max_temperature)
points(wankara_scale$Max_temperature,fitted(fit_i7),col="green",pch=20)

#################################################################################################
# R.3
# KNN
install.packages("kknn")
library("kknn")
fitknn1 <- kknn(wankara_scale$Mean_temperature ~ ., wankara_scale, wankara_scale)
names(fitknn1)

# Visualización
plot(wankara_scale$Mean_temperature~wankara_scale$Max_temperature)
points(wankara_scale$Max_temperature,fitknn1$fitted.values,col="blue",pch=20)

# ECM
yprime = fitknn1$fitted.values
sqrt(sum((wankara_scale$Mean_temperature-yprime)^2)/length(yprime)) #RMSE
```

```r
# Uso el mejor resultado anterior
fitknn2 = kknn(wankara_scale$Mean_temperature~.
               -Precipitation+Max_wind_speed*Wind_speed+I(Min_temperature^2)
               +I(Max_temperature^2)+Min_temperature*Dewpoint+I(Dewpoint^2)
               -Dewpoint-Visibility-Max_wind_speed,wankara_scale,wankara_scale)
yprime = fitknn2$fitted.values
sqrt(sum((wankara_scale$Mean_temperature-yprime)^2)/length(yprime))

plot(wankara_scale$Mean_temperature~wankara_scale$Max_temperature)
points(wankara_scale$Max_temperature,fitknn2$fitted.values,col="red",pch=20)

# Modelo más interpretable anterior --> Mejor resultado aún
fitknn3 = kknn(wankara_scale$Mean_temperature~.-Wind_speed
               -Standard_pressure-Precipitation-Sea_level_pressure
               -Max_wind_speed-Visibility,wankara_scale,wankara_scale)
yprime = fitknn3$fitted.values
sqrt(sum((wankara_scale$Mean_temperature-yprime)^2)/length(yprime))
plot(wankara_scale$Mean_temperature~wankara_scale$Max_temperature)
points(wankara_scale$Max_temperature,fitknn3$fitted.values,col="green",pch=20)


##################################################################################################
# R.4
# Comparación de algoritmos

nombre <- "./data/wankara/wankara"
run_lm_fold <- function(i, x, tt = "test") {
  file <- paste(x, "-5-", i, "tra.dat", sep="");
  x_tra <- read.csv(file, comment.char="@" , header=FALSE )
  file <- paste(x, "-5-", i, "tst.dat", sep="");
  x_tst <- read.csv(file, comment.char="@" , header=FALSE )
  In <- length(names(x_tra)) - 1
  names(x_tra)[1:In] <- paste ("X", 1:In, sep="");
  names(x_tra)[In+1] <- "Y"
  names(x_tst)[1:In] <- paste ("X", 1:In, sep="");
  names(x_tst)[In+1] <- "Y"
  if (tt == "train") { test <- x_tra }
  else { test <- x_tst }
  fitMulti=lm(Y~.,x_tra)
  yprime=predict(fitMulti,test)
  sum(abs(test$Y-yprime)^2)/length(yprime) ##MSE
}
lmMSEtrain<-sapply(1:5,run_lm_fold,nombre,"train")
medialmMSEtrain = mean(lmMSEtrain)
lmMSEtest<-sapply(1:5,run_lm_fold,nombre,"test")
medialmMSEtest = mean(lmMSEtest)

run_kknn_fold <- function(i, x, tt = "test") {
  file <- paste(x, "-5-", i, "tra.dat", sep="");
  x_tra <- read.csv(file, comment.char="@" , header=FALSE )
  file <- paste(x, "-5-", i, "tst.dat", sep="");
  x_tst <- read.csv(file, comment.char="@" , header=FALSE )
  In <- length(names(x_tra)) - 1
```

```
  names(x_tra)[1:In] <- paste ("X", 1:In, sep="");
  names(x_tra)[In+1] <- "Y"
  names(x_tst)[1:In] <- paste ("X", 1:In, sep="");
  names(x_tst)[In+1] <- "Y"
  if (tt == "train") { test <- x_tra }
  else { test <- x_tst }
  fitKNN=kknn(Y~.,x_tra,test)
  yprime=fitKNN$fitted.values
  sum(abs(test$Y-yprime)^2)/length(yprime) ##MSE
}
kknnMSEtrain<-sapply(1:5,run_kknn_fold,nombre,"train")
mediakknnMSEtrain = mean(kknnMSEtrain)
kknnMSEtest<-sapply(1:5,run_kknn_fold,nombre,"test")
mediakknnMSEtest= mean(kknnMSEtest)

# Random Forest
install.packages("randomForest")
library(randomForest)
run_rf_fold <- function(i, x, tt = "test") {
  file <- paste(x, "-5-", i, "tra.dat", sep="");
  x_tra <- read.csv(file, comment.char="@" , header=FALSE )
  file <- paste(x, "-5-", i, "tst.dat", sep="");
  x_tst <- read.csv(file, comment.char="@" , header=FALSE )
  In <- length(names(x_tra)) - 1
  names(x_tra)[1:In] <- paste ("X", 1:In, sep="");
  names(x_tra)[In+1] <- "Y"
  names(x_tst)[1:In] <- paste ("X", 1:In, sep="");
  names(x_tst)[In+1] <- "Y"
  if (tt == "train") { test <- x_tra }
  else { test <- x_tst }
  fitrf=randomForest(Y~.,data=x_tra)
  yprime = predict(fitrf, newdata=test)
  sum(abs(test$Y-yprime)^2)/length(yprime) ##MSE
}
rfMSEtrain<-sapply(1:5,run_rf_fold,nombre,"train")
rfMSEtest<-sapply(1:5,run_rf_fold,nombre,"test")


# COMPARATIVA EN TEST

resultados <- read.csv("./data/regr_test_alumnos.csv")
tablatst <- cbind(resultados[,2:dim(resultados)[2]])
colnames(tablatst) <- names(resultados)[2:dim(resultados)[2]]
rownames(tablatst) <- resultados[,1]
#leemos la tabla con los errores medios de entrenamiento
resultados <- read.csv("./data/regr_train_alumnos.csv")
tablatr <- cbind(resultados[,2:dim(resultados)[2]])
colnames(tablatr) <- names(resultados)[2:dim(resultados)[2]]
rownames(tablatr) <- resultados[,1]

# Añadiendo a las tablas mis resultados
tablatst[17,1] = medialmMSEtest
tablatst[17,2] = mediakknnMSEtest
```

```r
tablatr[17,1] = medialmMSEtrain
tablatr[17,2] = mediakknnMSEtrain

##lm (other) vs knn (ref)
# + 0.1 porque wilcox R falla para valores == 0 en la tabla
difs <- (tablatst[,1] - tablatst[,2]) / tablatst[,1]
wilc_1_2 <- cbind(ifelse (difs<0, abs(difs)+0.1, 0+0.1),
                  ifelse (difs>0, abs(difs)+0.1, 0+0.1))
colnames(wilc_1_2) <- c(colnames(tablatst)[1], colnames(tablatst)[2])
head(wilc_1_2)

LMvsKNNtst <- wilcox.test(wilc_1_2[,1], wilc_1_2[,2],
                          alternative = "two.sided", paired=TRUE)
Rmas <- LMvsKNNtst$statistic
pvalue <- LMvsKNNtst$p.value
LMvsKNNtst <- wilcox.test(wilc_1_2[,2], wilc_1_2[,1],
                          alternative = "two.sided", paired=TRUE)
Rmenos <- LMvsKNNtst$statistic
Rmas
Rmenos
pvalue

## lm (other) vs m5p (ref)
difs <- (tablatst[,1] - tablatst[,3]) / tablatst[,1]
wilc_1_3 <- cbind(ifelse (difs<0, abs(difs)+0.1, 0+0.1),
                  ifelse (difs>0, abs(difs)+0.1, 0+0.1))
colnames(wilc_1_3) <- c(colnames(tablatst)[1], colnames(tablatst)[3])
head(wilc_1_3)

LMvsM5Ptst <- wilcox.test(wilc_1_3[,1], wilc_1_3[,2],
                          alternative = "two.sided", paired=TRUE)
Rmas <- LMvsM5Ptst$statistic
pvalue <- LMvsM5Ptst$p.value
LMvsM5Ptst <- wilcox.test(wilc_1_3[,2], wilc_1_3[,1],
                          alternative = "two.sided", paired=TRUE)
Rmenos <- LMvsM5Ptst$statistic
Rmas
Rmenos
pvalue

## kknn (other) vs m5p (ref)
difs <- (tablatst[,2] - tablatst[,3]) / tablatst[,2]
wilc_2_3 <- cbind(ifelse (difs<0, abs(difs)+0.1, 0+0.1),
                  ifelse (difs>0, abs(difs)+0.1, 0+0.1))
colnames(wilc_2_3) <- c(colnames(tablatst)[2], colnames(tablatst)[3])
head(wilc_2_3)

KKNNvsM5Ptst <- wilcox.test(wilc_2_3[,1], wilc_2_3[,2],
                            alternative = "two.sided", paired=TRUE)
Rmas <- KKNNvsM5Ptst$statistic
pvalue <- KKNNvsM5Ptst$p.value
KKNNvsM5Ptst <- wilcox.test(wilc_2_3[,2], wilc_2_3[,1],
                            alternative = "two.sided", paired=TRUE)
```

```r
Rmenos <- KKNNvsM5Ptst$statistic
Rmas
Rmenos
pvalue

# Comparativa general con Friedman
test_friedman <- friedman.test(as.matrix(tablatst))
test_friedman

tam <- dim(tablatst)
groups <- rep(1:tam[2], each=tam[1])
pairwise.wilcox.test(as.matrix(tablatst), groups,
                     p.adjust = "holm", paired = TRUE)

# COMPARATIVAS EN TRAINING
# lm (other) vs kknn (reference)
difs_tr <- (tablatr[,1] - tablatr[,2]) / tablatr[,1]
wilc_1_2_tr <- cbind(ifelse (difs_tr<0, abs(difs_tr)+0.1, 0+0.1),
                     ifelse (difs_tr>0, abs(difs_tr)+0.1, 0+0.1))
colnames(wilc_1_2_tr) <- c(colnames(tablatr)[1], colnames(tablatr)[2])
head(wilc_1_2_tr)

LMvsKNNtr <- wilcox.test(wilc_1_2_tr[,1], wilc_1_2_tr[,2],
                         alternative = "two.sided", paired=TRUE)
Rmas_tr <- LMvsKNNtr$statistic
pvalue_tr <- LMvsKNNtr$p.value
LMvsKNNtr <- wilcox.test(wilc_1_2_tr[,2], wilc_1_2_tr[,1],
                         alternative = "two.sided", paired=TRUE)
Rmenos_tr <- LMvsKNNtr$statistic
Rmas_tr
Rmenos_tr
pvalue_tr

# lm (other) vs m5p (reference)
difs_tr <- (tablatr[,1] - tablatr[,3]) / tablatr[,1]
wilc_1_3_tr <- cbind(ifelse (difs_tr<0, abs(difs_tr)+0.1, 0+0.1),
                     ifelse (difs_tr>0, abs(difs_tr)+0.1, 0+0.1))
colnames(wilc_1_3_tr) <- c(colnames(tablatra)[1], colnames(tablatra)[2])
head(wilc_1_3_tr)

LMvsM5Ptr <- wilcox.test(wilc_1_3_tr[,1], wilc_1_3_tr[,2],
                         alternative = "two.sided", paired=TRUE)
Rmas_tr <- LMvsM5Ptr$statistic
pvalue_tr <- LMvsM5Ptr$p.value
LMvsM5Ptr <- wilcox.test(wilc_1_3_tr[,2], wilc_1_3_tr[,1],
                         alternative = "two.sided", paired=TRUE)
Rmenos_tr <- LMvsM5Ptr$statistic
Rmas_tr
Rmenos_tr
pvalue_tr

# kknn(other) vs m5p (reference)
difs_tr <- (tablatr[,2] - tablatr[,3]) / tablatr[,2]
```

```r
wilc_2_3_tr <- cbind(ifelse (difs_tr<0, abs(difs_tr)+0.1, 0+0.1),
                     ifelse (difs_tr>0, abs(difs_tr)+0.1, 0+0.1))
colnames(wilc_2_3_tr) <- c(colnames(tablatra)[1], colnames(tablatra)[2])
head(wilc_2_3_tr)

KKNNvsM5Ptr <- wilcox.test(wilc_2_3_tr[,1], wilc_2_3_tr[,2],
                           alternative = "two.sided", paired=TRUE)
Rmas_tr <- KKNNvsM5Ptr$statistic
pvalue_tr <- KKNNvsM5Ptr$p.value
KKNNvsM5Ptr <- wilcox.test(wilc_2_3_tr[,2], wilc_2_3_tr[,1],
                           alternative = "two.sided", paired=TRUE)
Rmenos_tr <- KKNNvsM5Ptr$statistic
Rmas_tr
Rmenos_tr
pvalue_tr

# Comparativa conjunta
test_friedman_tr <- friedman.test(as.matrix(tablatr))
test_friedman_tr

tam <- dim(tablatst)
groups_ <- rep(1:tam[2], each=tam[1])
pairwise.wilcox.test(as.matrix(tablatr), groups,
                     p.adjust = "holm", paired = TRUE)

##################################################################################################
# EXTRA: Comparación con algoritmos sobre los resultados del 5-fold añadiendo Random Forest

resultados_train = cbind(lmMSEtrain,kknnMSEtrain,rfMSEtrain)
tablatra = as.data.frame(resultados_train,
                         col.names=c("lm_MSE_train","kknn_MSE_train","rf_MSE_train"))
resultados_test = cbind(lmMSEtest,kknnMSEtest,rfMSEtest)
tablatst = as.data.frame(resultados_test,
                         col.names=c("lm_MSE_test","kknn_MSE_test","rf_MSE_test"))

##lm (other) vs knn (ref)
# + 0.1 porque wilcox R falla para valores == 0 en la tabla
difs <- (tablatst[,1] - tablatst[,2]) / tablatst[,1]
wilc_1_2 <- cbind(ifelse (difs<0, abs(difs)+0.1, 0+0.1),
                  ifelse (difs>0, abs(difs)+0.1, 0+0.1))
colnames(wilc_1_2) <- c(colnames(tablatst)[1], colnames(tablatst)[2])
head(wilc_1_2)

LMvsKNNtst <- wilcox.test(wilc_1_2[,1], wilc_1_2[,2],
                          alternative = "two.sided", paired=TRUE)
Rmas <- LMvsKNNtst$statistic
pvalue <- LMvsKNNtst$p.value
LMvsKNNtst <- wilcox.test(wilc_1_2[,2], wilc_1_2[,1],
                          alternative = "two.sided", paired=TRUE)
Rmenos <- LMvsKNNtst$statistic
Rmas
Rmenos
pvalue
```

```
##lm (other) vs rf (ref)
# + 0.1 porque wilcox R falla para valores == 0 en la tabla
difs <- (tablatst[,1] - tablatst[,3]) / tablatst[,1]
wilc_1_3 <- cbind(ifelse (difs<0, abs(difs)+0.1, 0+0.1),
                  ifelse (difs>0, abs(difs)+0.1, 0+0.1))
colnames(wilc_1_3) <- c(colnames(tablatst)[1], colnames(tablatst)[2])
head(wilc_1_3)

LMvsRFtst <- wilcox.test(wilc_1_3[,1], wilc_1_3[,2],
                         alternative = "two.sided", paired=TRUE)
Rmas <- LMvsRFtst$statistic
pvalue <- LMvsRFtst$p.value
LMvsRFtst <- wilcox.test(wilc_1_3[,2], wilc_1_3[,1],
                         alternative = "two.sided", paired=TRUE)
Rmenos <- LMvsRFtst$statistic
Rmas
Rmenos
pvalue

##kknn (other) vs rf (ref)
# + 0.1 porque wilcox R falla para valores == 0 en la tabla
difs <- (tablatst[,2] - tablatst[,3]) / tablatst[,2]
wilc_2_3 <- cbind(ifelse (difs<0, abs(difs)+0.1, 0+0.1),
                  ifelse (difs>0, abs(difs)+0.1, 0+0.1))
colnames(wilc_2_3) <- c(colnames(tablatst)[1], colnames(tablatst)[2])
head(wilc_2_3)

KKNNvsRFtst <- wilcox.test(wilc_2_3[,1], wilc_2_3[,2],
                           alternative = "two.sided", paired=TRUE)
Rmas <- KKNNvsRFtst$statistic
pvalue <- KKNNvsRFtst$p.value
KKNNvsRFtst <- wilcox.test(wilc_2_3[,2], wilc_2_3[,1],
                           alternative = "two.sided", paired=TRUE)
Rmenos <- KKNNvsRFtst$statistic
Rmas
Rmenos
pvalue

# COMPARATIVA GENERAL TEST

test_friedman <- friedman.test(as.matrix(tablatst))
test_friedman

tam <- dim(tablatst)
groups <- rep(1:tam[2], each=tam[1])
pairwise.wilcox.test(as.matrix(tablatst), groups,
                     p.adjust = "holm", paired = TRUE)

# COMPARATIVA EN TRAINING

# EXAMINAMOS TRAINING PARA COMPROBAR SI HAY SOBREAPRENDIZAJE
difs_tr <- (tablatra[,1] - tablatra[,2]) / tablatra[,1]
wilc_1_2_tr <- cbind(ifelse (difs_tr<0, abs(difs_tr)+0.1, 0+0.1),
```

```r
                     ifelse (difs_tr>0, abs(difs_tr)+0.1, 0+0.1))
colnames(wilc_1_2_tr) <- c(colnames(tablatra)[1], colnames(tablatra)[2])
head(wilc_1_2_tr)

LMvsKNNtr <- wilcox.test(wilc_1_2_tr[,1], wilc_1_2_tr[,2],
                         alternative = "two.sided", paired=TRUE)
Rmas_tr <- LMvsKNNtr$statistic
pvalue_tr <- LMvsKNNtr$p.value
LMvsKNNtr <- wilcox.test(wilc_1_2_tr[,2], wilc_1_2_tr[,1],
                         alternative = "two.sided", paired=TRUE)
Rmenos_tr <- LMvsKNNtr$statistic
Rmas_tr
Rmenos_tr
pvalue_tr


# Comparativa general
test_friedman_tr <- friedman.test(as.matrix(tablatra))
test_friedman_tr

tam_tr <- dim(tablatra)
groups_tr <- rep(1:tam_tr[2], each=tam_tr[1])
pairwise.wilcox.test(as.matrix(tablatr), groups,
                     p.adjust = "holm", paired = TRUE)
```