modelwnewvariables

Annabelle

July 7, 2020

file set up

DataSource <- "./data" # input raw dataset goes here  
output.dir <-"./output" # output dataset writes to here  
functions.dir <- "./functions" # directory for functions   
  
#Functions to read in   
library(janitor) #cleaning data

##   
## Attaching package: 'janitor'

## The following objects are masked from 'package:stats':  
##   
## chisq.test, fisher.test

library(tidyverse) #cleaning data

## ── Attaching packages ────────────────────────────────── tidyverse 1.2.1 ──

## ✓ ggplot2 3.2.1 ✓ purrr 0.3.3  
## ✓ tibble 2.1.1 ✓ dplyr 0.8.3  
## ✓ tidyr 1.0.0 ✓ stringr 1.4.0  
## ✓ readr 1.3.1 ✓ forcats 0.4.0

## ── Conflicts ───────────────────────────────────── tidyverse\_conflicts() ──  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(ggplot2)  
library(readxl) #loading in files   
library(circlize)

## ========================================  
## circlize version 0.4.8  
## CRAN page: https://cran.r-project.org/package=circlize  
## Github page: https://github.com/jokergoo/circlize  
## Documentation: http://jokergoo.github.io/circlize\_book/book/  
##   
## If you use it in published research, please cite:  
## Gu, Z. circlize implements and enhances circular visualization   
## in R. Bioinformatics 2014.  
## ========================================

library(readr)  
library(skimr)  
library(igraph)

##   
## Attaching package: 'igraph'

## The following object is masked from 'package:circlize':  
##   
## degree

## The following objects are masked from 'package:dplyr':  
##   
## as\_data\_frame, groups, union

## The following objects are masked from 'package:purrr':  
##   
## compose, simplify

## The following object is masked from 'package:tidyr':  
##   
## crossing

## The following object is masked from 'package:tibble':  
##   
## as\_data\_frame

## The following objects are masked from 'package:stats':  
##   
## decompose, spectrum

## The following object is masked from 'package:base':  
##   
## union

library(knitr) #for making tables form kable   
library(kableExtra)

##   
## Attaching package: 'kableExtra'

## The following object is masked from 'package:dplyr':  
##   
## group\_rows

library(reshape2)

##   
## Attaching package: 'reshape2'

## The following object is masked from 'package:tidyr':  
##   
## smiths

library(tidytext) #suggested in github helpform for an error  
library(data.table)

##   
## Attaching package: 'data.table'

## The following objects are masked from 'package:reshape2':  
##   
## dcast, melt

## The following objects are masked from 'package:dplyr':  
##   
## between, first, last

## The following object is masked from 'package:purrr':  
##   
## transpose

library(glmmTMB)  
library(car)

## Loading required package: carData

##   
## Attaching package: 'car'

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

## The following object is masked from 'package:purrr':  
##   
## some

#function qqPlot - car

RegData <- read\_csv(paste(DataSource,"/newdf2.csv", sep = ""))

## Warning: Missing column names filled in: 'X1' [1]

## Parsed with column specification:  
## cols(  
## X1 = col\_double(),  
## total = col\_double(),  
## area\_x = col\_double(),  
## percentpublic = col\_double(),  
## taxa = col\_double(),  
## nsumemploy = col\_double(),  
## total\_x\_x = col\_double()  
## )

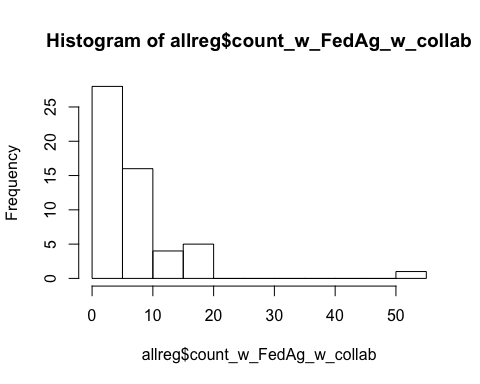
tdata <- read.csv(paste0(DataSource,"/Updated\_variables.csv"), stringsAsFactors = FALSE) #data provided by tyler   
tdata <- tdata %>% as\_tibble() #now can use stringr packages   
tdata <- tdata %>% ##cleaning/reformating colunm names --> default is lower\_case\_snake  
 clean\_names()  
  
processed <- read.csv(paste0("/usr/local/bin/store/partner\_rff/data/ProcessedPartnersDataModified.csv"),   
 stringsAsFactors = FALSE, na = c("", " ", "NA"))

modify data

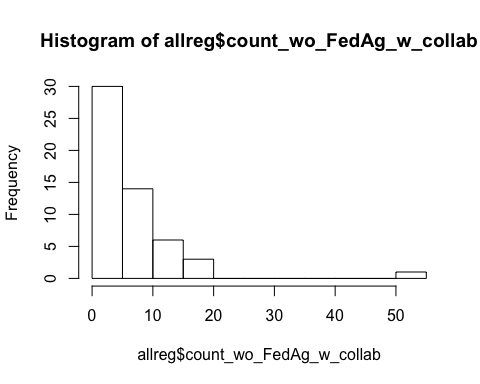
##join tdata by regdata  
  
allreg <- RegData %>% full\_join(tdata, by = c("area\_x", "total\_x\_x", "percentpublic"))  
  
allreg <- allreg[,-c(1,2,8,12:72)]  
  
allreg <- allreg %>% full\_join(processed, by = c("scientific\_name", "common\_name"))  
  
allreg <- allreg[,-c(6:16)]  
  
#remove NAs   
allreg <- allreg[-which(allreg$count\_w\_FedAg\_w\_collab == 0),]  
#note missing data for 3 species?   
  
allreg <- allreg[-which(is.na(allreg$area\_x)),]  
  
#set into two new variables   
  
allregFedAg <- allreg[,-c(7)]  
  
allregNOFA <- allreg[,-c(6)]

View new response variables as histograms

hist(allreg$count\_w\_FedAg\_w\_collab, breaks = 12)

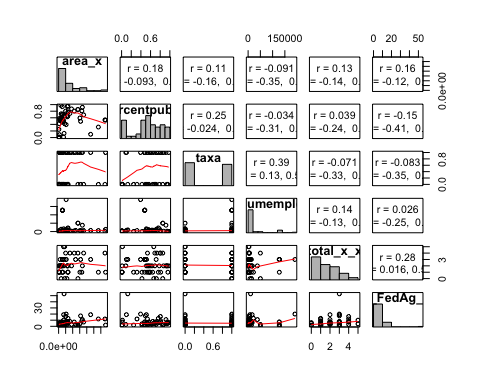


hist(allreg$count\_wo\_FedAg\_w\_collab, breaks = 12)



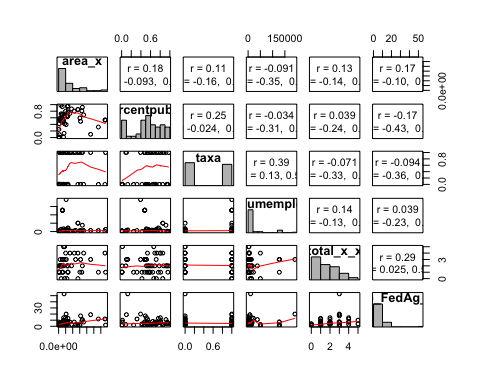
cor-mat allregFedAg

#set up  
PredictorsOnlyPixel <- allregFedAg[,c(6)]  
PredictAndResponsePixel <- allregFedAg  
PredictAndResponseGrid <- allregFedAg  
   
# put histograms on the diagonal panel   
panel.hist <- function (x,...) # define a function that says what we want to plot in the diagonal  
{  
 usr <- par("usr"); on.exit(par(usr)) # not sure what usr is for?  
 par(usr = c(usr[1:2],0,1.5))  
 h <- hist(x, plot = FALSE)  
 breaks <- h$breaks; nB <- length(breaks) # make the hist   
 y <- h$counts; y <- y/max(y)  
 rect(breaks[-nB], 0, breaks[-1], y, col="grey", ...) # defines what the histogram is going to look like  
}  
  
## put correlations & 95% CIs on the upper panels,  
panel.cor <- function(x, y, digits=2, prefix="", cex.cor, ...)  
{  
 usr <- par("usr"); on.exit(par(usr))  
 par(usr = c(0, 1, 0, 1))  
 r <- cor(x, y,use="complete.obs")  
 txt <- format(c(r, 0.123456789), digits=digits)[1]  
 prefix <- "r = "  
 rc <- cor.test(x,y)  
 rci <- rc$conf.int  
 txt2 <- format(c(rci, 0.123456789), digits=digits)[1]  
 txt3 <- format(c(rci, 0.123456789), digits=digits)[2]  
 prefix2 <- "\nCI = "  
 txt <- paste(prefix, txt, prefix2, txt2, ", ", txt3, sep="")  
 if(missing(cex.cor)) cex.cor <- 0.8/strwidth(txt)  
 text(0.5, 0.5, txt, cex = 1)  
}  
  
## plot a correlation matrix plot that uses the functions specified above to say what to plot where  
 ## this was taken directly from website and still not plotting r values for all   
pairs(PredictAndResponsePixel[1:6], lower.panel=panel.smooth, cex = .8, diag.panel=panel.hist, cex.labels = 1.2, font.labels=2, upper.panel=panel.cor)



cor-mat allregNOFA

#set up  
PredictorsOnlyPixel <- allregNOFA[,c(6)]  
PredictAndResponsePixel <- allregNOFA  
PredictAndResponseGrid <- allregNOFA  
   
# put histograms on the diagonal panel   
panel.hist <- function (x,...) # define a function that says what we want to plot in the diagonal  
{  
 usr <- par("usr"); on.exit(par(usr)) # not sure what usr is for?  
 par(usr = c(usr[1:2],0,1.5))  
 h <- hist(x, plot = FALSE)  
 breaks <- h$breaks; nB <- length(breaks) # make the hist   
 y <- h$counts; y <- y/max(y)  
 rect(breaks[-nB], 0, breaks[-1], y, col="grey", ...) # defines what the histogram is going to look like  
}  
  
## put correlations & 95% CIs on the upper panels,  
panel.cor <- function(x, y, digits=2, prefix="", cex.cor, ...)  
{  
 usr <- par("usr"); on.exit(par(usr))  
 par(usr = c(0, 1, 0, 1))  
 r <- cor(x, y,use="complete.obs")  
 txt <- format(c(r, 0.123456789), digits=digits)[1]  
 prefix <- "r = "  
 rc <- cor.test(x,y)  
 rci <- rc$conf.int  
 txt2 <- format(c(rci, 0.123456789), digits=digits)[1]  
 txt3 <- format(c(rci, 0.123456789), digits=digits)[2]  
 prefix2 <- "\nCI = "  
 txt <- paste(prefix, txt, prefix2, txt2, ", ", txt3, sep="")  
 if(missing(cex.cor)) cex.cor <- 0.8/strwidth(txt)  
 text(0.5, 0.5, txt, cex = 1)  
}  
  
## plot a correlation matrix plot that uses the functions specified above to say what to plot where  
 ## this was taken directly from website and still not plotting r values for all   
pairs(PredictAndResponsePixel[1:6], lower.panel=panel.smooth, cex = .8, diag.panel=panel.hist, cex.labels = 1.2, font.labels=2, upper.panel=panel.cor)



VIFS

vif(lm(count\_wo\_FedAg\_w\_collab ~ area\_x +percentpublic + taxa + nsumemploy + total\_x\_x,data = allregNOFA))

## area\_x percentpublic taxa nsumemploy total\_x\_x   
## 1.082155 1.120848 1.383702 1.319804 1.079181

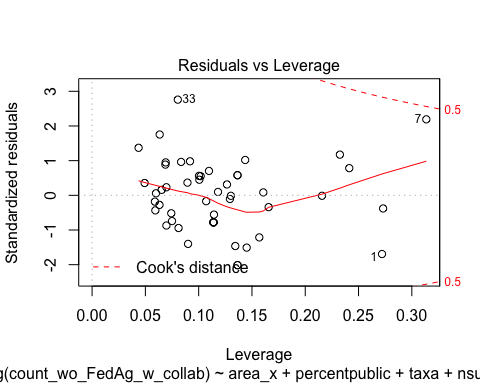
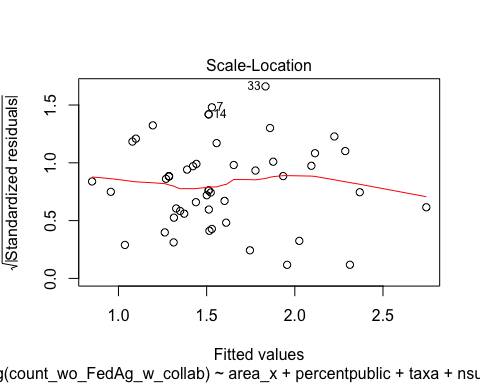
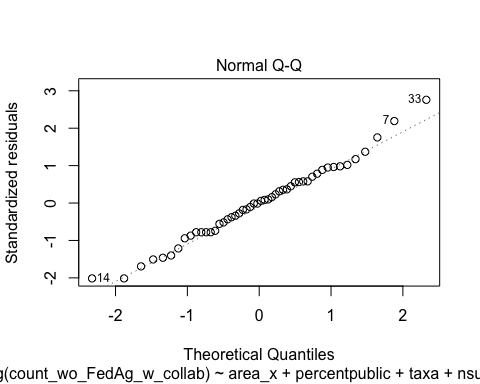
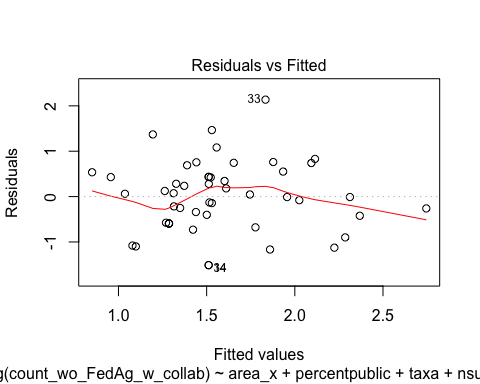
vif(lm(count\_w\_FedAg\_w\_collab ~ area\_x +percentpublic + taxa + nsumemploy + total\_x\_x,data = allregFedAg))

## area\_x percentpublic taxa nsumemploy total\_x\_x   
## 1.082155 1.120848 1.383702 1.319804 1.079181

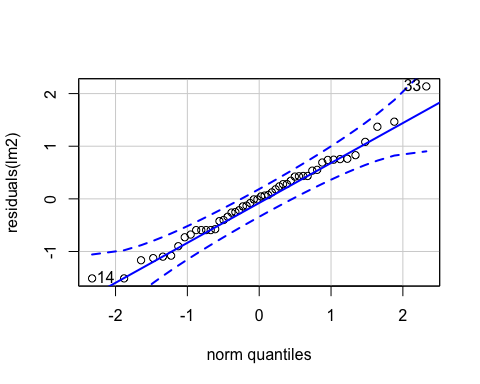
lm2 <- lm(log(count\_wo\_FedAg\_w\_collab) ~ area\_x +percentpublic + taxa + nsumemploy + total\_x\_x,data = allregNOFA)  
#withou log   
#lm2 <- lm((count\_wo\_FedAg\_w\_collab) ~ area\_x +percentpublic + taxa + nsumemploy + total\_x\_x,data = allregNOFA)  
  
summary(lm2)

##   
## Call:  
## lm(formula = log(count\_wo\_FedAg\_w\_collab) ~ area\_x + percentpublic +   
## taxa + nsumemploy + total\_x\_x, data = allregNOFA)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.51129 -0.58896 0.01815 0.43462 2.13658   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 1.046e+00 3.188e-01 3.280 0.00203 \*\*  
## area\_x 5.998e-12 3.604e-12 1.664 0.10315   
## percentpublic -7.066e-02 4.223e-01 -0.167 0.86790   
## taxa -1.766e-01 2.689e-01 -0.657 0.51477   
## nsumemploy -7.956e-07 2.957e-06 -0.269 0.78912   
## total\_x\_x 2.373e-01 9.421e-02 2.519 0.01548 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.8075 on 44 degrees of freedom  
## (5 observations deleted due to missingness)  
## Multiple R-squared: 0.2091, Adjusted R-squared: 0.1192   
## F-statistic: 2.326 on 5 and 44 DF, p-value: 0.05855

#kable(table(lm2$coefficients))  
  
par(mfrow = c(1, 1))  
plot(lm2)



qqPlot(residuals(lm2))



## 33 14   
## 31 13

par(mfrow = c(1, 1))  
  
#plot(predicted(lm2), residuals(lm2))  
#hist(residuals(lm2))  
# how add CI equivalent around q-q plot?   
AIC(lm2)

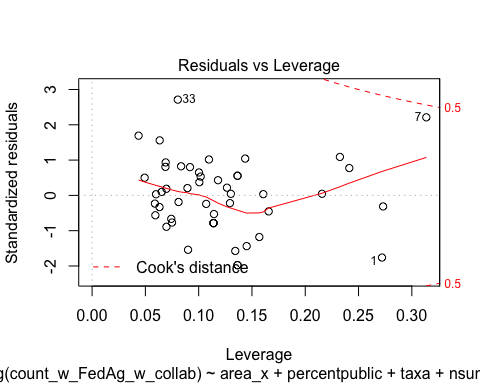
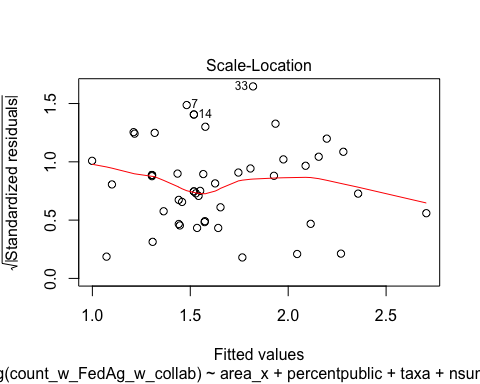
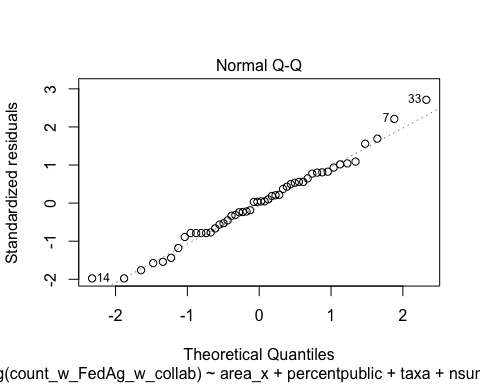
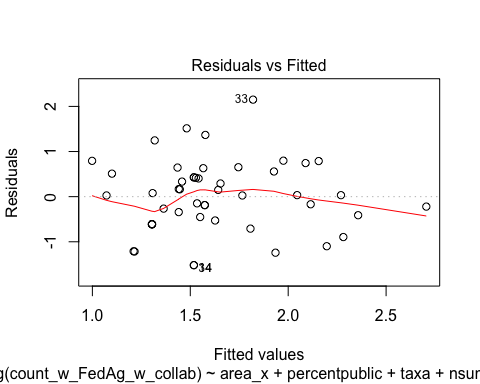
## [1] 128.118

#check\_overdispersion(lm2)

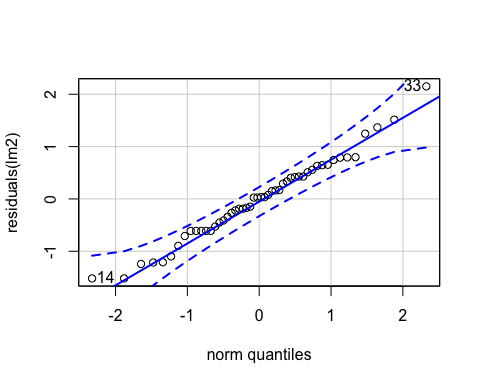
lm2 <- lm(log(count\_w\_FedAg\_w\_collab) ~ area\_x +percentpublic + taxa + nsumemploy + total\_x\_x,data = allregFedAg)  
#withou log   
#lm2 <- lm((count\_wo\_FedAg\_w\_collab) ~ area\_x +percentpublic + taxa + nsumemploy + total\_x\_x,data = allregNOFA)  
  
summary(lm2)

##   
## Call:  
## lm(formula = log(count\_w\_FedAg\_w\_collab) ~ area\_x + percentpublic +   
## taxa + nsumemploy + total\_x\_x, data = allregFedAg)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.5183 -0.5904 0.0297 0.4890 2.1499   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 1.081e+00 3.263e-01 3.312 0.00186 \*\*  
## area\_x 5.625e-12 3.689e-12 1.525 0.13444   
## percentpublic -9.997e-03 4.323e-01 -0.023 0.98165   
## taxa -9.430e-02 2.752e-01 -0.343 0.73350   
## nsumemploy -1.520e-06 3.026e-06 -0.502 0.61805   
## total\_x\_x 2.230e-01 9.643e-02 2.313 0.02547 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.8265 on 44 degrees of freedom  
## (5 observations deleted due to missingness)  
## Multiple R-squared: 0.1793, Adjusted R-squared: 0.08606   
## F-statistic: 1.923 on 5 and 44 DF, p-value: 0.1097

#kable(table(lm2$coefficients))  
  
par(mfrow = c(1, 1))  
plot(lm2)



qqPlot(residuals(lm2))



## 33 14   
## 31 13

par(mfrow = c(1, 1))  
  
#plot(predicted(lm2), residuals(lm2))  
#hist(residuals(lm2))  
# how add CI equivalent around q-q plot?   
AIC(lm2)

## [1] 130.4502

#check\_overdispersion(lm2)