Simulated data used for training ML models in the AlienZoo study series

Ulrike Kuhl

7/10/2020

Introduction

This code simulates datasets with known dependencies between features for the AlienZoo study line. Written by Ulrike Kuhl, July 2020; for comments, questions etc.: kuhl@techfak.uni-bielefeld.de.

Let's start simulating

```r
# import required modules
knitr::opts_chunk$set(echo = TRUE, warning=FALSE)
library(ggplot2)
library(dplyr)
```

```
##
## Attache Paket: 'dplyr'

## Die folgenden Objekte sind maskiert von 'package:stats':
##
##     filter, lag

## Die folgenden Objekte sind maskiert von 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(data.table)
```

```
##
## Attache Paket: 'data.table'

## Die folgenden Objekte sind maskiert von 'package:dplyr':
##
##     between, first, last
```

Define basic properties of features and variables.

```r
# minimal / maximal growth rates:
GRneg.min=0.1
GRneg.max=1.0
GRpos.min=1.1
GRpos.max=1.9

F.min=0
F.max=7
```

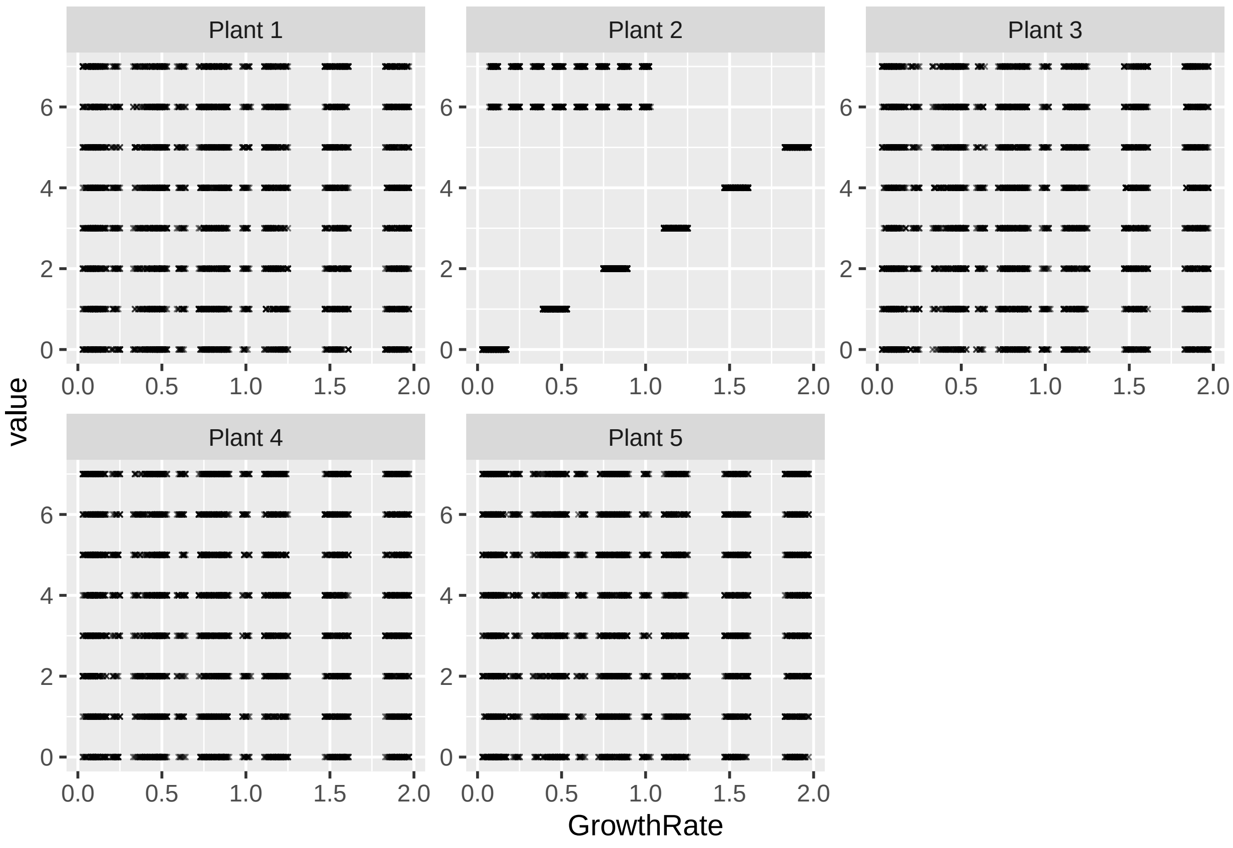
Data set 1: simple linear relationship between one feature and the outcome

Desired properties:

- 3276800 data points (given by combinatoric nature of having 5 features that may take one of 8 values = 32768 combinations; each combination being repeated 100 times);
- there is a linear relationship between plant 2 and the growth rate up until selecting 6 or more (prevent overfeeding + selecting a very stupid strategy of "the more, the better")

```r
# define the possible values each feature might take
featureVals.possible=c(F.min:F.max)
# expand to have a set with 5 features and all possible combinations of features
df.set1=expand.grid(featureVals.possible,featureVals.possible,featureVals.possible,featureVals.possible
# repeat to have each combination 100 times
df.set1=df.set1[rep(seq_len(nrow(df.set1)), each = 100), ]

# generate new field for growth rate values, that correlate with values of plant 2, where plant 2 range
# in this dataset, smaller values of plant 2 are detrimental, too!
# transform values of deciding feature 'plant 2':
# Note, the optimal feeding value should be 5: change OldRange = (F.max - F.min) --> OldRange = (5 - F.
OldRange = (5 - F.min)
NewRange = (GRpos.max - GRneg.min)
df.set1$GR = round(jitter((((df.set1$Var2 - F.min) * NewRange) / OldRange) + GRneg.min, factor = 1),2)
# destroy relationship for variable values 6 and 7 (= 'overfeeding')
OldRange = (F.max - F.min)
NewRange = (GRneg.max - GRneg.min)
# note: depending on var1 here, to destroy any dependency with var 2
df.set1$GR[df.set1$Var2==6]=round(jitter((((sample(0:7, length(df.set1$GR[df.set1$Var2==6]), replace=T)
df.set1$GR[df.set1$Var2==7]=round(jitter((((sample(0:7, length(df.set1$GR[df.set1$Var2==6]), replace=T)

# for easier Pytorch model training:
##GRfin=df.set1$GR
# how many class labels do we have?
##class_num=length(sort(unique(GRfin)))
##pytorch_class_codes=0:(class_num-1)
##for(i in 1:length(GRfin)) {
##  GRclassLabs[i]=df.set1$GR[1]
##}

df.set1$GRrank=frankv(df.set1$GR, ties.method="dense")-1

# visualize
# prepare helpful bits:
# new labels
plant.labs <- c("Var1", "Var2","Var3","Var4","Var5","Var6","Var7")
names(plant.labs) <- c("Plant 1", "Plant 2","Plant 3","Plant 4","Plant 5","Plant 6","Plant 7")

# take subset of data frame for plotting so it does not take too long:
df.set1.plot=df.set1
set.seed(42)
rows=sample(nrow(df.set1.plot))
df.set1.plot <- df.set1.plot[rows, ]
df.set1.plot = df.set1.plot[seq(1, nrow(df.set1), 1000), ]
names(df.set1.plot)=c("Plant 1","Plant 2","Plant 3","Plant 4","Plant 5","GrowthRate")
```

```
p1 = df.set1.plot %>% tidyr::gather("id", "value", 1:5) %>%
  ggplot(., aes(GrowthRate, value))+
  geom_point(shape=4, alpha=0.5, size=0.5)+
  #geom_jitter()+
  #geom_smooth(method = "lm", se=FALSE, color="black")+
  facet_wrap(~id,scales="free")

p1
```



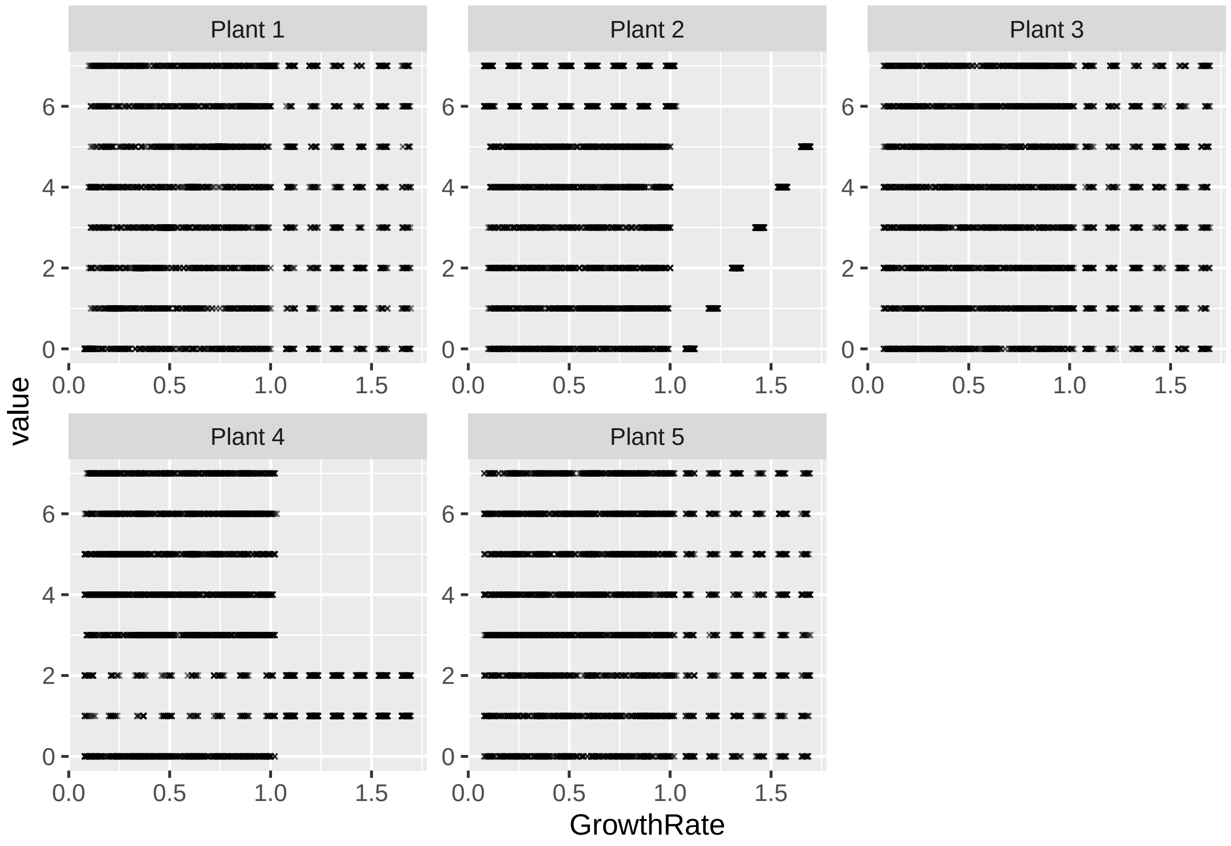
```
# save data!
#write.csv(df.set1,'AlienZooDataSet1.csv')
```

Data set 2: simple linear relationship between one feature and the outcome, depending on the value of a second feature

Desired properties:

- 3276800 data points (given by combinatoric nature of having 5 features that may take one of 8 values = 32768 combinations; each combination being repeated 100 times);
- there is a linear relationship between plant 2 and the growth rate, iff
  - plant 4 has a value of 1 or 2
  - this holds unless var2 = [6,7] (prevent overfeeding + selecting a very stupid strategy of "the more, the better")

```
# define the possible values each feature might take
featureVals.possible=c(F.min:F.max)
# expand to have a set with 5 features and all possible combinations of features
df.set2=expand.grid(featureVals.possible,featureVals.possible,featureVals.possible,featureVals.possible
# repeat to have each combination 100 times
df.set2=df.set2[rep(seq_len(nrow(df.set2)), each = 100), ]

# generate new field for growth rate values
# growth rate values per default in a negative range
set.seed(42)
df.set2$GR=round(runif(nrow(df.set2), min=GRneg.min, max=GRneg.max),2)

# transform GR values of deciding feature 'plant 2', depending on the value of 'plant 4':
df.set2$GR[df.set2$Var4==1 | df.set2$Var4==2]=df.set2$Var2[df.set2$Var4==1 | df.set2$Var4==2]

OldRange = (F.max - F.min)
NewRange = (GRpos.max - GRpos.min)

df.set2$GR[df.set2$Var4==1 | df.set2$Var4==2] = jitter((((df.set2$GR[df.set2$Var4==1 | df.set2$Var4==2]

# destroy relationship for variable values 6 and 7 (= 'overfeeding')
OldRange = (F.max - F.min)
NewRange = (GRneg.max - GRneg.min)
# note: depending on var1 here, to destroy any dependency with var 2
df.set2$GR[df.set1$Var2==6]=round(jitter((((df.set2$Var1 - F.min) * NewRange) / OldRange) + GRneg.min,
df.set2$GR[df.set1$Var2==7]=round(jitter((((df.set2$Var1 - F.min) * NewRange) / OldRange) + GRneg.min,

# take subset of data frame for plotting so it does not take too long:
df.set2.plot=df.set2
set.seed(42)
rows=sample(nrow(df.set2.plot))
df.set2.plot <- df.set2.plot[rows, ]
df.set2.plot = df.set2.plot[seq(1, nrow(df.set2), 1000), ]
names(df.set2.plot)=c("Plant 1","Plant 2","Plant 3","Plant 4","Plant 5","GrowthRate")

p2= df.set2.plot %>% tidyr::gather("id", "value", 1:5) %>%
  ggplot(., aes(GrowthRate, value))+
  geom_point(shape=4, alpha=0.5, size=0.5)+
  #geom_jitter()+
  #geom_smooth(method = "lm", se=FALSE, color="black")+
  facet_wrap(~id,scales="free")

p2
```



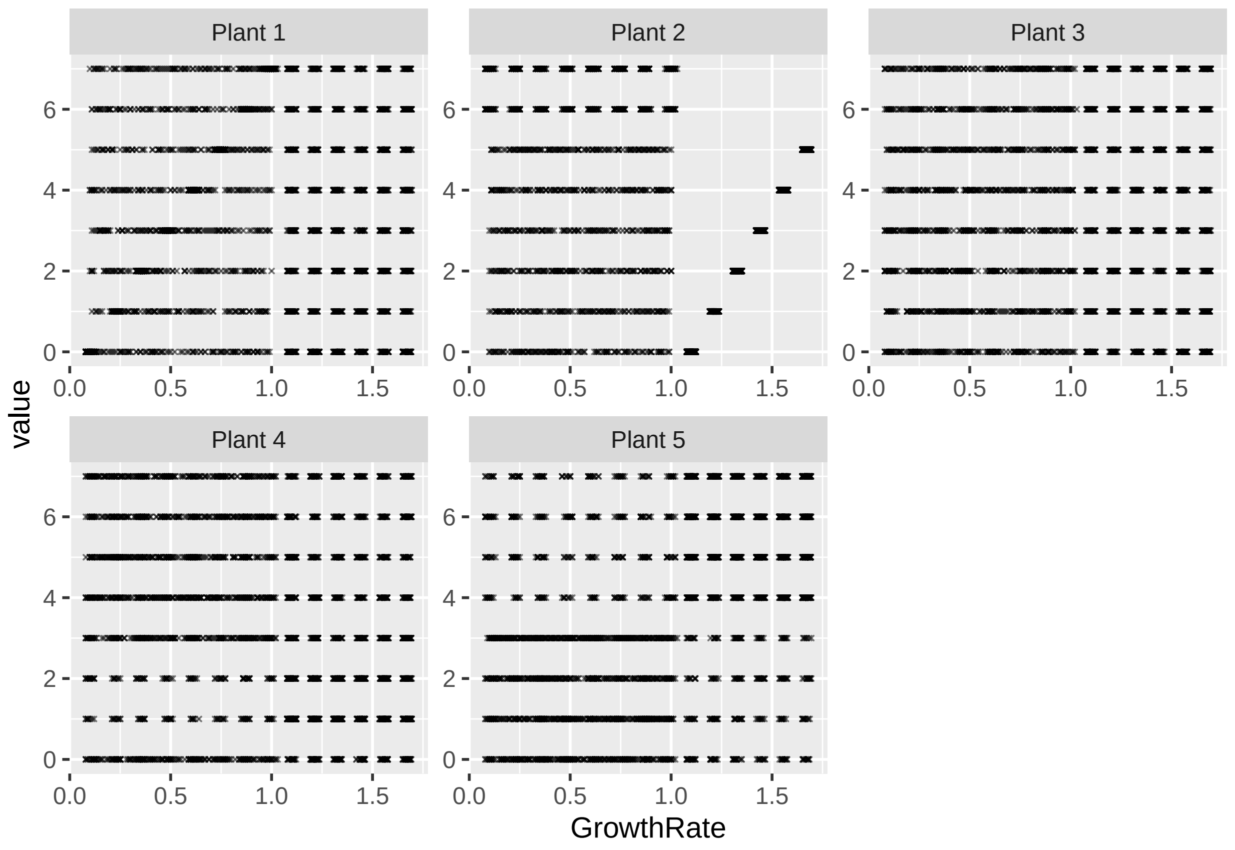
```
# save data!
#write.csv(df.set2,'AlienZooDataSet2.csv')
```

Data set 3: simple linear relationship between one feature and the outcome, depending on the value of either one of two additional variables

- 3276800 data points (given by combinatoric nature of having 5 features that may take one of 8 values = 3276800 combinations; each combination being repeated 100 times);
- there is a linear relationship between plant 2 and the growth rate, iff
  - plant 4 has a value of 1 or 2 OR
  - plant 5 is not smaller than 4

```
# define the possible values each feature might take
featureVals.possible=c(F.min:F.max)
# expand to have a set with 5 features and all possible combinations of features
df.set3=expand.grid(featureVals.possible,featureVals.possible,featureVals.possible,featureVals.possible
# repeat to have each combination 100 times
df.set3=df.set3[rep(seq_len(nrow(df.set3)), each = 100), ]

# generate new field for growth rate values
# growth rate values per default in a negative range
set.seed(42)
df.set3$GR=round(runif(nrow(df.set3), min=GRneg.min, max=GRneg.max),2)

# transform GR values of deciding feature 'plant 2',
# depending on the values of 'plant 4' and 'plant 5':
df.set3$GR[df.set3$Var4==1 | df.set3$Var4==2 | df.set3$Var5>3 ]=df.set3$Var2[df.set3$Var4==1 | df.set3$V
```

```
OldRange = (F.max - F.min)
NewRange = (GRpos.max - GRpos.min)
df.set3$GR[df.set3$Var4==1 | df.set3$Var4==2 | df.set3$Var5>3] = jitter((((df.set3$GR[df.set3$Var4==1 |


# destroy relationship for variable values 6 and 7 (= 'overfeeding')
OldRange = (F.max - F.min)
NewRange = (GRneg.max - GRneg.min)
# note: depending on var1 here, to destroy any dependency with var 2
df.set3$GR[df.set3$Var2==6]=round(jitter((((df.set3$Var1 - F.min) * NewRange) / OldRange) + GRneg.min,
df.set3$GR[df.set3$Var2==7]=round(jitter((((df.set3$Var1 - F.min) * NewRange) / OldRange) + GRneg.min,

# take subset of data frame for plotting so it does not take too long:
df.set3.plot=df.set3
set.seed(42)
rows=sample(nrow(df.set3.plot))
df.set3.plot <- df.set3.plot[rows, ]
df.set3.plot = df.set3.plot[seq(1, nrow(df.set3), 1000), ]
names(df.set3.plot)=c("Plant 1","Plant 2","Plant 3","Plant 4","Plant 5","GrowthRate")

p3 = df.set3.plot %>% tidyr::gather("id", "value", 1:5) %>%
  ggplot(., aes(GrowthRate, value))+
  geom_point(shape=4, alpha=0.5, size=0.5)+
  #geom_jitter()+
  #geom_smooth(method = "lm", se=FALSE, color="black")+
  facet_wrap(~id,scales="free")

p3
```

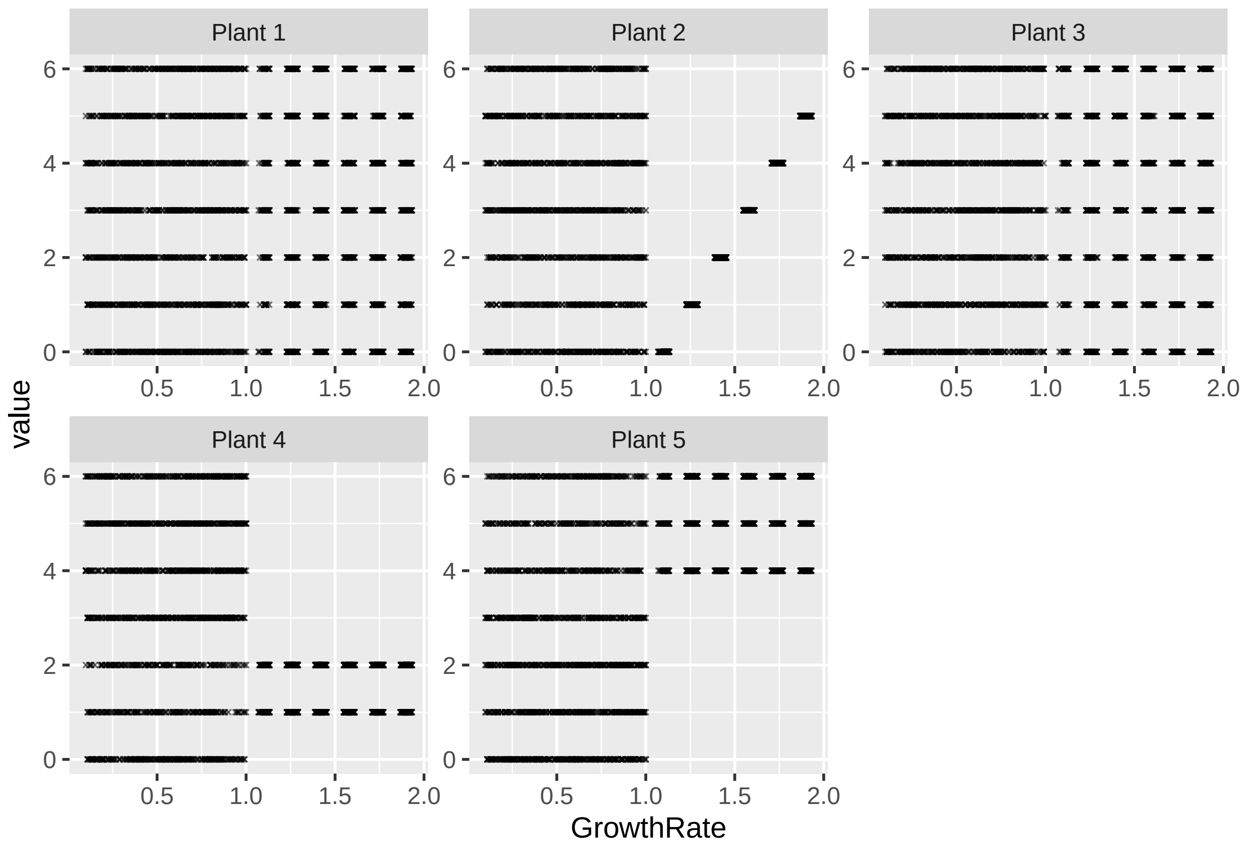


```r
# save data!
#write.csv(df.set3,'AlienZooDataSet3.csv')
```

Data set 4: simple linear relationship between one feature and the outcome, depending on the value of a second and a third variable.

- 2975292 data points;
- there is a linear relationship between plant 2 and the growth rate, iff
  - plant 4 has a value of 1 or 2 AND
  - plant 5 is not smaller than 4

```r
# get rid of 7th feature - otherwise, computed CFEs suggests them
F.max=6

# define the possible values each feature might take
featureVals.possible=c(F.min:F.max)
# expand to have a set with 5 features and all possible combinations of features
df.set4=expand.grid(featureVals.possible,featureVals.possible,featureVals.possible,featureVals.possible
# repeat to have each combination 100 times
df.set4=df.set4[rep(seq_len(nrow(df.set4)), each = 100), ]

# generate new field for growth rate values
# growth rate values per default in a negative range
set.seed(42)
df.set4$GR=round(runif(nrow(df.set4), min=GRneg.min, max=GRneg.max),2)

# transform GR values of deciding feature 'plant 2',
# depending on the values of 'plant 4' and 'plant 5':
df.set4$GR[(df.set4$Var4==1 | df.set4$Var4==2 ) & df.set4$Var5>3 ]=df.set4$Var2[(df.set4$Var4==1 | df.se

# scale GR to range of positive growth rate + add jitter
# NOTE THAT F.max that has an actual effect should be 5, not 7, to get maximal values for feeding 5x pl
F.max_effect=5
OldRange = (F.max_effect - F.min)
NewRange = (GRpos.max - GRpos.min)
# AND variant:
df.set4$GR[(df.set4$Var4==1 | df.set4$Var4==2 ) & df.set4$Var5>3] = jitter((((df.set4$GR[(df.set4$Var4=

# add on (compared to ds 3): make sure GR does not exceed 2!
maxGR=max(df.set4$GR[(df.set4$Var4==1 | df.set4$Var4==2 ) & df.set4$Var5>3])
# reduce GR values > 2:
df.set4$GR[(df.set4$Var4==1 | df.set4$Var4==2 ) & df.set4$Var5>3][df.set4$GR[(df.set4$Var4==1 | df.set4

# destroy relationship for variable values 6 (= 'overfeeding')
OldRange = (F.max - F.min)
NewRange = (GRneg.max - GRneg.min)
df.set4$GR[df.set4$Var2==6]=round(runif(length(df.set4$GR[df.set4$Var2==6]), min=GRneg.min, max=GRneg.ma

# Balance out occurences of positive and negative growth rates
# (repeat positive entries)
df.set4_negRates=df.set4[df.set4$GR<1.1,]
df.set4_posRates=df.set4[df.set4$GR>=1.1,]
df.set4_posRates=df.set4_posRates[rep(seq_len(nrow(df.set4_posRates)), each = 9), ]
df.set4=rbind(df.set4_negRates, df.set4_posRates)

# take subset of data frame for plotting so it does not take too long:
df.set4.plot=df.set4
set.seed(42)
rows=sample(nrow(df.set4.plot))
df.set4.plot <- df.set4.plot[rows, ]
df.set4.plot = df.set4.plot[seq(1, nrow(df.set4), 1000), ]
names(df.set4.plot)=c("Plant 1","Plant 2","Plant 3","Plant 4","Plant 5","GrowthRate")

p4 = df.set4.plot %>% tidyr::gather("id", "value", 1:5) %>%
  ggplot(., aes(GrowthRate, value))+
  geom_point(shape=4, alpha=0.5, size=0.5)+
  #geom_jitter()+
  #geom_smooth(method = "lm", se=FALSE, color="black")+
  facet_wrap(~id,scales="free")

p4
```



```r
# save data!
# write.csv(df.set4,'AlienZooDataSet4.csv')
```