Napfoltok és neurális hálók - Idősoros előrejelzés az RNN csomag felhasználásával

Előadó: Milibák Eszter

E-mail: milibakeszter@gmail.com

Budapest, 2017. november 15.

Motiváció

Ommundsen
Are you sure this is needed for time series analysis?

Az „RNN” csomag

Visszacsatoláso s (recurrent) neurális hálók

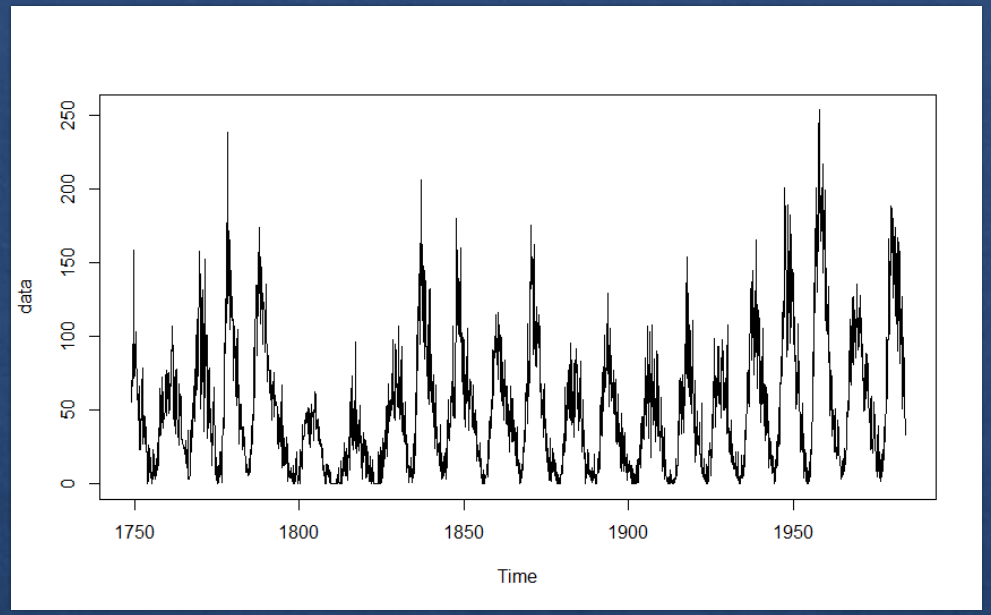
Adatelőkészítés:
(1) Data frame
(2) Data matrix
(3) 3D array

Első lépések

```r
##################################
######### Előkészítés ###########
##################################

#Adatbehívás:
data <- sunspots #Beépített R-adatokkal dolgozunk
head(data)

#Néhány általános tudnivalü:
is.ts(data)
#[1] TRUE

plot.ts(data)
mean(data)
#[1] 51.26596

sd(data)
#[1] 43.44897

#Data frame-mé alakítunk:
data_df <- as.data.frame(data)
names(data_df) <- c("qty")

###Időadatok - évszázad (c), évtized (d), év (y), hónap (month):
library(zoo)

times <- as.data.frame(as.yearmon(time(data)))
years <- as.data.frame(substr(times[, 1], 5, 8))

months <- as.data.frame(substr(times[, 1], 1, 3))
months_numeric <- match(months[, 1], month.abb)

data_df <- cbind(data_df, years, months_numeric)
names(data_df) <- c("qty", "year", "month")

data_df$c <- NA
data_df$d <- NA
data_df$y <- NA

for (i in 1:nrow(data_df)) {
  data_df$c[i] <- as.numeric(as.character(substr(as.character(data_df$year[i]), 2, 2)))
  data_df$d[i] <- as.numeric(as.character(substr(as.character(data_df$year[i]), 3, 3)))
  data_df$y[i] <- as.numeric(as.character(substr(as.character(data_df$year[i]), 4, 4)))
}

#Training Â©s test setek szÃétválasztása:
train <- as.data.frame(data_df[as.numeric(as.character(data_df$year)) < 1960,]) #Training: 1960 elott
names(train) <- names(data_df)

test <- as.data.frame(data_df[as.numeric(as.character(data_df$year)) >= 1960,]) #Test: 1960-1984
names(test) <- names(data_df)
```

A napfoltok száma 1749 és 1984 között

```
> data
       Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov   Dec
1749  58.0  62.6  70.0  55.7  85.0  83.5  94.8  66.3  75.9  75.5 158.6  85.2
1750  73.3  75.9  89.2  88.3  90.0 100.0  85.4 103.0  91.2  65.7  63.3  75.4
1751  70.0  43.5  45.3  56.4  60.7  50.7  66.3  59.8  23.5  23.2  28.5  44.0
1752  35.0  50.0  71.0  59.3  59.7  39.6  78.4  29.3  27.1  46.6  37.6  40.0
1753  44.0  32.0  45.7  38.0  36.0  31.7  22.2  39.0  28.0  25.0  20.0   6.7
1754   0.0   3.0   1.7  13.7  20.7  26.7  18.8  12.3   8.2  24.1  13.2   4.2
1755  10.2  11.2   6.8   6.5   0.0   0.0   8.6   3.2  17.8  23.7   6.8  20.0
1756  12.5   7.1   5.4   9.4  12.5  12.9   3.6   6.4  11.8  14.3  17.0   9.4
1757  14.1  21.2  26.2  30.0  38.1  12.8  25.0  51.3  39.7  32.5  64.7  33.5
1758  37.6  52.0  49.0  72.3  46.4  45.0  44.0  38.7  62.5  37.7  43.0  43.0
1759  48.3  44.0  46.8  47.0  49.0  50.0  51.0  71.3  77.2  59.7  46.3  57.0
1760  67.3  59.5  74.7  58.3  72.0  48.3  66.0  75.6  61.3  50.6  59.7  61.0
1761  70.0  91.0  80.7  71.7 107.2  99.3  94.1  91.1 100.7  88.7  89.7  46.0
1762  43.8  72.8  45.7  60.2  39.9  77.1  33.8  67.7  68.5  69.3  77.8  77.2
1763  56.5  31.9  34.2  32.9  32.7  35.8  54.2  26.5  68.1  46.3  60.9  61.4
1764  59.7  59.7  40.2  34.4  44.3  30.0  30.0  30.0  28.2  28.0  26.0  25.7
1765  24.0  26.0  25.0  22.0  20.2  20.0  27.0  29.7  16.0  14.0  14.0  13.0
1766  12.0  11.0  36.6   6.0  26.8   3.0   3.3   4.0   4.3   5.0   5.7  19.2
1767  27.4  30.0  43.0  32.9  29.8  33.3  21.9  40.8  42.7  44.1  54.7  53.3
1768  53.5  66.1  46.3  42.7  77.7  77.4  52.6  66.8  74.8  77.8  90.6 111.8
1769  73.9  64.2  64.3  96.7  73.6  94.4 118.6 120.3 148.8 158.2 148.1 112.0
1770 104.0 142.5  80.1  51.0  70.1  83.3 109.8 126.3 104.4 103.6 132.2 102.3
1771  36.0  46.2  46.7  64.9 152.7 119.5  67.7  58.5 101.4  90.0  99.7  95.7
1772 100.9  90.8  31.1  92.2  38.0  57.0  77.3  56.2  50.5  78.6  61.3  64.0
1773  54.6  29.0  51.2  32.9  41.1  28.4  27.7  12.7  29.3  26.3  40.9  43.2
1774  46.8  65.4  55.7  43.8  51.3  28.5  17.5   6.6   7.9  14.0  17.7  12.2
1775   4.4   0.0  11.6  11.2   3.9  12.3   1.0   7.9   3.2   5.6  15.1   7.9
1776  21.7  11.6   6.3  21.8  11.2  19.0   1.0  24.2  16.0  30.0  35.0  40.0
1777  45.0  36.5  39.0  95.5  80.3  80.7  95.0 112.0 116.2 106.5 146.0 157.3
1778 177.3 109.3 134.0 145.0 238.9 171.6 153.0 140.0 171.7 156.3 150.3 105.0
1779 114.7 165.7 118.0 145.0 140.0 113.7 143.0 112.0 111.0 124.0 114.0 110.0
1780  70.0  98.0  98.0  95.0 107.2  88.0  86.0  86.0  93.7  77.0  60.0  58.7
1781  98.7  74.7  53.0  68.3 104.7  97.7  73.5  66.0  51.0  27.3  67.0  35.2
1782  54.0  37.5  37.0  41.0  54.3  38.0  37.0  44.0  34.0  23.2  31.5  30.0
1783  28.0  38.7  26.7  28.3  23.0  25.2  32.2  20.0  18.0   8.0  15.0  10.5
1784  13.0   8.0  11.0  10.0   6.0   9.0   6.0  10.0  10.0   8.0  17.0  14.0
1785   6.5   8.0   9.0  15.7  20.7  26.3  36.3  20.0  32.0  47.2  40.2  27.3
1786  37.2  47.6  47.7  85.4  92.3  59.0  83.0  89.7 111.5 112.3 116.0 112.7
1787 134.7 106.0  87.4 127.2 134.8  99.2 128.0 137.2 157.3 157.0 141.5 174.0
1788 138.0 129.2 143.3 108.5 113.0 154.2 141.5 136.0 141.0 142.0  94.7 129.5
1789 114.0 125.3 120.0 123.3 123.5 120.0 117.0 103.0 112.0  89.7 134.0 135.5
1790 103.0 127.5  96.3  94.0  93.0  91.0  69.3  87.0  77.3  84.3  82.0  74.0
1791  72.7  62.0  74.0  77.2  73.7  64.2  71.0  43.0  66.5  61.7  67.0  66.0
1792  58.0  64.0  63.0  75.7  62.0  61.0  45.8  60.0  59.0  59.0  57.0  56.0
```




```
> head(data_df)
   qty year month c d y
1 58.0 1749     1 7 4 9
2 62.6 1749     2 7 4 9
3 70.0 1749     3 7 4 9
4 55.7 1749     4 7 4 9
5 85.0 1749     5 7 4 9
6 83.5 1749     6 7 4 9
```

Modellezés I. – Normalizálás

```r
##################################
## Modellezés az RNN csomaggal: ##
##################################

library(rnn) #RNN csomag behívása
set.seed(12345)


#Normalizálás:

library(plyr)
train_min <- apply(data.matrix(train), 2, FUN = min) #Training-adatok minimumainak kiszámítása
train_max <- apply(data.matrix(train), 2, FUN = max) #Training-adatok maximumainak kiszámítása

train_norm <- data.frame(matrix(nrow = nrow(train), ncol = ncol(train)))

for (i in 1:nrow(train)) {
  for (j in 1:ncol(train)) {
    train_norm[i, j] <- (as.numeric(as.character(train[i, j])) - train_min[j])/
      (train_max[j] - train_min[j])
  }
}

names(train_norm) <- names(data_df)
```

Modellezés II. – Data frame, data matrix, 3d array

```r
# Lépés 1.: Data frame

train_norm_qty_df <- data.frame(matrix(nrow = nrow(train_norm)  - 12, ncol = 12))
train_norm_c_df <- data.frame(matrix(nrow = nrow(train_norm)  - 12, ncol = 12))
train_norm_d_df <- data.frame(matrix(nrow = nrow(train_norm)  - 12, ncol = 12))
train_norm_y_df <- data.frame(matrix(nrow = nrow(train_norm)  - 12, ncol = 12))
train_norm_month_df <- data.frame(matrix(nrow = nrow(train_norm)  - 12, ncol = 12))


for (i in 1:nrow(train_norm_qty_df)) {
  train_norm_qty_df[i, ] = train_norm[i:(i+11), "qty"]
  train_norm_c_df[i, ] = train_norm[i:(i+11), "c"]
  train_norm_d_df[i, ] = train_norm[i:(i+11), "d"]
  train_norm_y_df[i, ] = train_norm[i:(i+11), "y"]
  train_norm_month_df[i, ] = train_norm[i:(i+11), "month"]

}

# Lépés 2.: Data matrix (numerikus mátrix)

train_norm_qty_m <- data.matrix(train_norm_qty_df)
train_norm_c_m <- data.matrix(train_norm_c_df)
train_norm_d_m <- data.matrix(train_norm_d_df)
train_norm_y_m <- data.matrix(train_norm_y_df)
train_norm_month_m <- data.matrix(train_norm_month_df)


# Lépés 3.: Array (3D-s tömb)

train_norm_qty_fin <- array(c(train_norm_qty_m),
                            dim=c(dim(train_norm_qty_m), 1))

train_norm_predictors_fin <- array(c(train_norm_c_m, train_norm_d_m, train_norm_y_m, train_norm_month_m),
                                   dim=c(dim(train_norm_month_m), 4))
```

Modellezés III. – trainr függvény

```
#Modellezés:
model <- trainr(Y=train_norm_qty_fin[,1:dim(train_norm_qty_fin)[2],],
                X=train_norm_predictors_fin[,1:dim(train_norm_predictors_fin)[2],],
                learningrate   =  0.001,
                hidden_dim     = 10,
                numepochs = 100,
                network_type = "lstm")
```

- Learning rate: tanulási faktor
- Hidden_dim: rejtett dimenziók száma
- Numepoch: ismétlésszám
- Network_Type: visszacsatolásos hálózat típusa, itt: „long short-term memory”

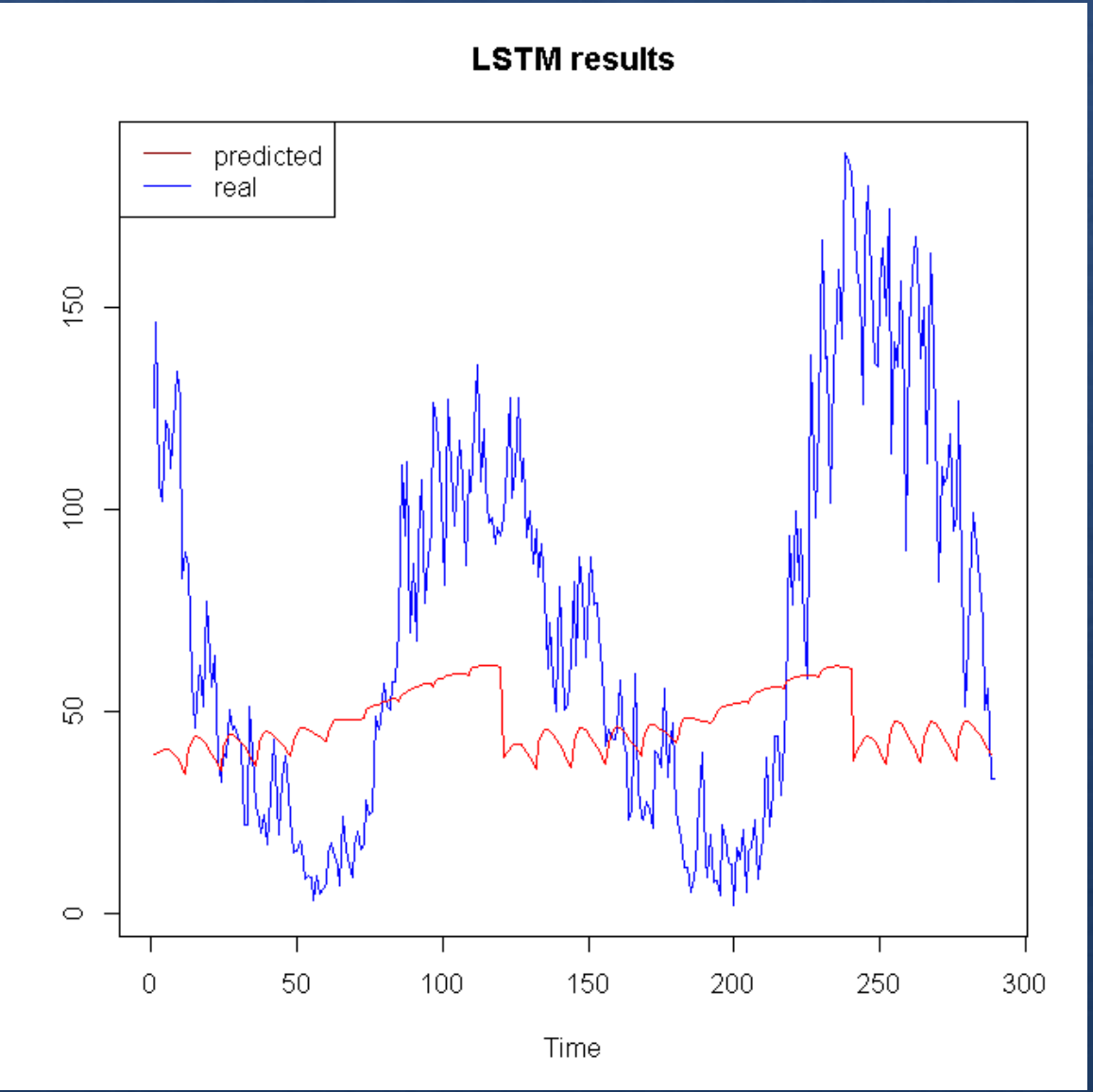
Előrejelzés I. – Normalizálás, DF/DM/3DA

```r
#################
## Előrejelzés: ##
#################

# Normalizálás:

test_norm <- data.frame(matrix(nrow = nrow(test), ncol = ncol(test)))

for (i in 1:nrow(test)) {
  for (j in 1:ncol(test)) {
    test_norm[i, j] <- (as.numeric(as.character(test[i, j])) - train_min[j])/
      (train_max[j] - train_min[j])
  }
}

names(test_norm) <- names(test)

#1. Lépés: data frame

test_norm_c_df <- data.frame(matrix(nrow = nrow(test), ncol = 12))
test_norm_d_df <- data.frame(matrix(nrow = nrow(test), ncol = 12))
test_norm_y_df <- data.frame(matrix(nrow = nrow(test), ncol = 12))
test_norm_month_df <- data.frame(matrix(nrow = nrow(test), ncol = 12))

test_norm_c_df[1, ] <- cbind(train_norm_c_df[nrow(train_norm_c_df), 2:12], test_norm$c[1])
test_norm_d_df[1, ] <- cbind(train_norm_d_df[nrow(train_norm_d_df), 2:12], test_norm$d[1])
test_norm_y_df[1, ] <- cbind(train_norm_y_df[nrow(train_norm_y_df), 2:12], test_norm$y[1])
test_norm_month_df[1, ] <- cbind(train_norm_month_df[nrow(train_norm_month_df), 2:12], test_norm$month[1])

for (i in 2:nrow(test_norm_c_df)) {
  test_norm_c_df[i, ] <- cbind(test_norm_c_df[i-1, 2:12], test_norm$c[i])
  test_norm_d_df[i, ] <- cbind(test_norm_d_df[i-1, 2:12], test_norm$d[i])
  test_norm_y_df[i, ] <- cbind(test_norm_y_df[i-1, 2:12], test_norm$y[i])
  test_norm_month_df[i, ] <- cbind(test_norm_month_df[i-1, 2:12], test_norm$month[i])
}

#2. Lépés: data.matrix

test_norm_c_m <- data.matrix(test_norm_c_df)
test_norm_d_m <- data.matrix(test_norm_d_df)
test_norm_y_m <- data.matrix(test_norm_y_df)
test_norm_month_m <- data.matrix(test_norm_month_df)

#3. Lépés: 3D array

test_norm_predictors_fin <- array(c(test_norm_c_m, test_norm_d_m, test_norm_y_m, test_norm_month_m),
                                  dim=c(dim(test_norm_month_m), 4))

```

Előrejelzés II. – predictr függvény, ellenőrzés

```r
#Előrejelzés:

predictions <- predictr(model, test_norm_predictors_fin)

predictions_final <- list()

for(i in 1:nrow(test)) {
  predictions_final[i] <- as.data.frame(predictions)[i, 12] * (train_max[1] - train_min[1]) + train_min[1]
}

#Valós és elorejelzett adatok összevetése ábrán:
real <- ts(data_df[(nrow(data_df)-(nrow(test)-1)):nrow(data_df), "qty"])
pred <- ts(unlist(predictions_final))

ts.plot(real, pred, gpars = list(col = c("blue", "red")), main = "LSTM results")
legend("topleft", legend = c("predicted", "real"), col = c("dark red", "blue"), lty = 1:1)
```

Előrejelzett vs. valós adatok



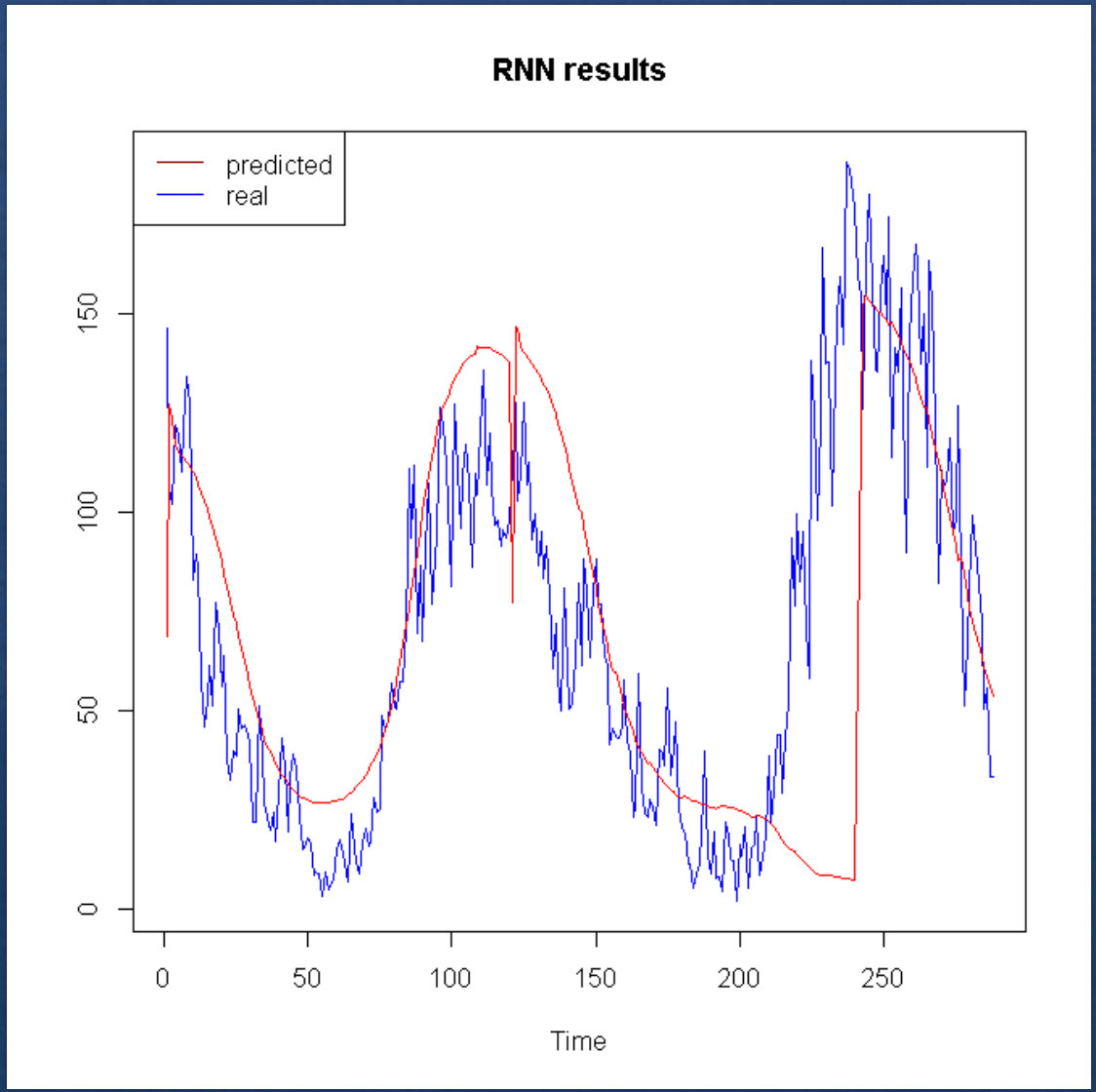
Alternatív paraméterezés

Köszönet Ujfalusi Sándornak (Ujfalusi.Sandor@gmail.com)!

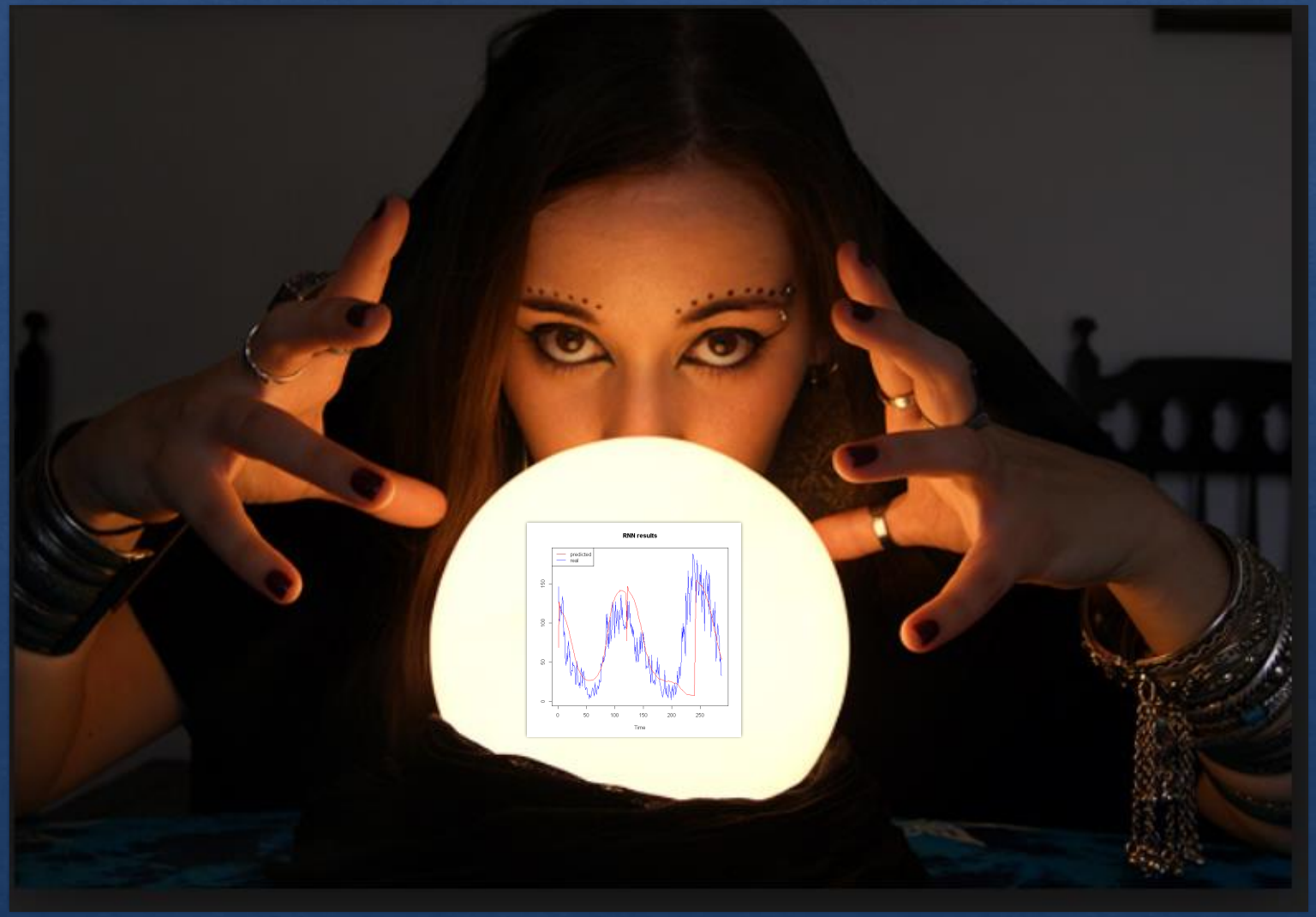
- Hosszabb, a napciklusokon átívelő ciklusok– pl. 300 hónap
- Egyszerűbb modell, gyorsabb training vs. pontosabb eredmények!

```
#Modell újrafuttatása alernatív paraméterezéssel:
model <- trainr(Y=train_norm_qty_fin[,1:dim(train_norm_qty_fin)[2],],
                X=train_norm_predictors_fin[,1:dim(train_norm_predictors_fin)[2],],
                learningrate   =  0.1,
                hidden_dim     = 10,
                numepochs = 5,
                network_type = "rnn")

```

Előrejelzett vs. valós adatok - Alternatív paraméterezés



Köszönöm a figyelmet!



Hivatkozások

- Kép 1. – Üveggömb (saját felirat): https://www.pinterest.com/pin/369435975662444401/
- Kép 2. – Napfoltok: http://pcpult.hu/it-technologia/hirek-ujdonsagok/szenzacios-napfolt-kepeket-keszitett-az-nst.html
- Kép 3. – Csomag: https://raphaelhertzog.com/2010/10/21/the-secret-plan-behind-the-3-0-quilt-debian-source-package-format/
- Kép 4. – Jósnő: http://www.joylaneclinic.co.uk/mystic-meg-not

A bemutatott kód letölthető innen:

https://github.com/milibakeszter/rnn_sunspots