ADS2 Problem Set 4

Wanlu Liu

October 5, 2019

This week, we learnt the importance to have shareable data-set including how to share raw data, what is considered as tidy data-set, and how to format a code book. We also learnt the importance of simulating data using synthetically.

In the lecture, I have showed two example of simulating a linear model and DNA sequence with python.

In this problem set, I hope you can try to generate some synthetic data-set with both python and R. Ideally, it shouldn't take more than one hour to complete. If you do, please let me or Melanie know, so we can adjust and plan accordingly.

Generate synthetic data for linear model in python

In the lecture, the synthetic data-set for linear model is generated with python (scikit-learn python package/module). The python code are listed below and the results are shown in Figure 1:

```
>>> from sklearn import linear_model, datasets
>>> from matplotlib import pyplot as plt
>>> n_samples = 1000
>>> X, y, coef = datasets.make_regression(n_samples=n_samples,
                n_features=1,n_informative=1, noise=10,coef=True, random_state=0)
>>> plt.scatter(X,y,color='yellowgreen', marker='.')
>>> plt.show()
```

Now please try it yourself, see whether you can reproduce my code. Note, if you have trouble import sklearn, try to install it first through pip or conda. (https://scikit-learn.org/stable/install.html)

Generate synthetic data for linear model in R

(1) Now it's your turn to try to generate a synthetic data for linear model in R and then visualize it with scatter-plot. The linear model you generate should fit the following formula:

$$y = 10.8x + 0.6$$

Hint: First generate some random x with runif() or rnorm().

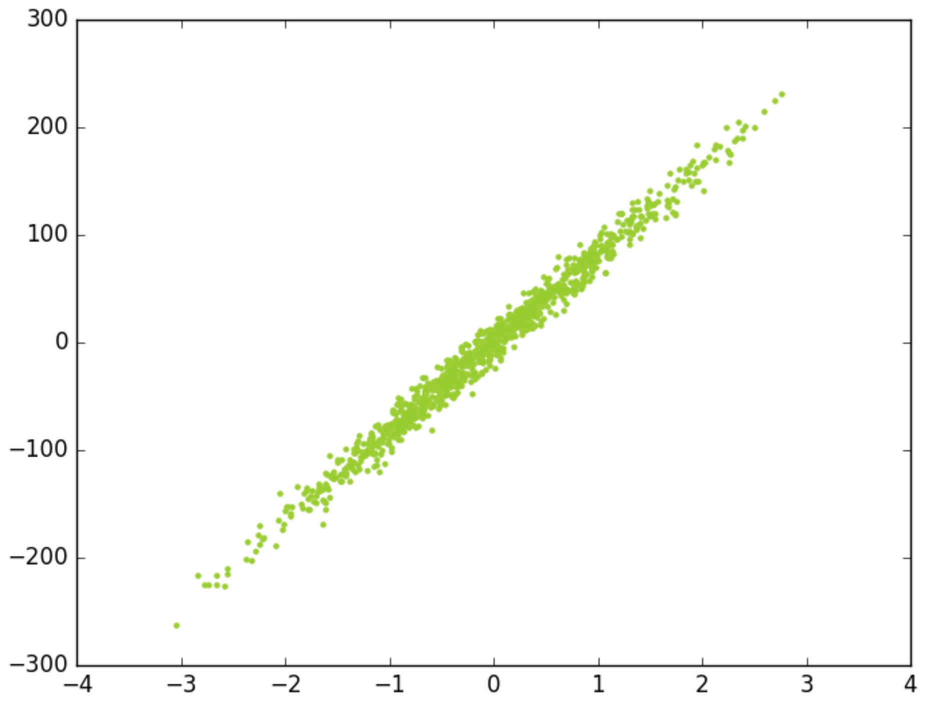

Figure 1: Scatter plot of simulated data for linear model with pytho sklearn.

```
set.seed(32) #to make your code reproducible, remember to set seed when you are
    random sampling something in R
x=runif(1000)
y=10.8*x+0.6
plot(x,y)

x=rnorm(1000)
y=10.8*x+0.6
plot(x,y)
```

(2) How does the scatterplot look like? Is it too artificial? Try to add 50% of normal distributed noise to the y and then plot the scatterplot and see whether this looks more realistic?
Hint: random sample 50% of your y use sample() function in R. Then add some random noise generated with rnorm() to the those y.

```
#add some noise
idx=sample(1:1000,500)
y[idx]=y[idx]+rnorm(length(idx))
plot(x,y)
```

Generate synthetic DNA sequence in python

In the lecture, the synthetic DNA sequence is generated with python. In this case, I assumed the percentage for A,C,G,T in the genome are 20%, 30%, 30%, 20%. The python code and results are listed below:

```
import random
>>> def DNA(length):
>>>     return ''.join(random.choice("A"*2+"C"*3+"G"*3+"T"*2) for _ in xrange(length
    ))
>>> print DNA(10)
CGGCCCCGCG
>>> print DNA(50)
GTCCCCTCCGCACTTCGTACATTGGTCTTCACGTTCGGATCCCTCTACTG
>>> print DNA(100)
CGGGCCCTATTTGGGGTAAGGTGTGGAATCCGGCCGGAGTGTCCGCCCTAACTTCTGCTTTCGATGCCTCGACGTCCCGCCGGGACTTGGGCACT
```

Now please try it yourself, see whether you can reproduce my code.

Generate synthetic DNA sequence in R

(1) Now it's your turn to try to generate some DNA sequence in R. In this step, please don't consider the frequency difference for A,C,G,T. (Assuming they are equally distributed, 25% for each nucleotide). Hint: Use sample() function in R. An alternative is to try randDNA() function from RBioinf R pacakge.

```
DNA=paste0(sample(c("A","C","G","T"),1000,rep=TRUE),collapse = "")
#sample() function sample A,T,C,G equally, paste0(collapse="") concatenate strings
    together.
```

(2) Now try to generate some DNA sequence considering different probability for each A,C,G,T nucleotide.

$$\begin{aligned} P(A) &= 0.25 \\ P(G) &= 0.3 \\ P(C) &= 0.25 \\ P(T) &= 0.2 \end{aligned}$$

```
DNA=paste0(sample(c("A","C","G","T"),1000,rep=TRUE,prob=c(0.25,0.25,0.3,0.2)),
    collapse = "")
#sample() function sample A,T,C,G equally, paste0(collapse="") concatenate strings
    together.
```