Then we will use the function **`random.shuffle`** to shuffle the list **`[0, 1, ...,n-1]`**, where **`n`** is the length of the data set. The **`shuffle`** function randomly rearranges the values of its argument in place, much like shuffling a deck of cards. We will call the list of indices **`ind`**, both before and after shuffling.

```
import random
n=len(data)
ind=list(range(n))
random.shuffle(ind)
```

At this point, the list **`ind`** still contains each of the integers **`[0, 1, ..., n-1]`**, but they have been rearranged in a random order. Since there are **`n`** total indices, then 75% of the indices (rounded to the nearest integer) is given by **`ntrain=round(0.75*n)`**. The idea is to extract the data pairs with indices given by the first **`ntrain`** numbers in the shuffled **`ind`** as the training set, and use the remaining data as the test set. Python lets us do this as follows:

```
ntrain=round(.75*n)
xtrain=xvals[ind[:ntrain]]
ytrain=yvals[ind[:ntrain]]
xtest=xvals[ind[ntrain:]]
ytest=yvals[ind[ntrain:]]
```

In this code, **`ind[:ntrain]`** gives the first 75% of the indices in the array **`ntrain`**, while **`ind[ntrain:]`** gives the remaining 25%. There is no-overlap, and all indices are included. By referring to **`xvals[list_of_indices]`** Python picks out those values in **`xvals`**, and only those values in **`xvals`**, that are expressed in **`list_of_indices`**. Here we have replaced **`list_of_indices`** with **`ind[:ntrain]`** to get a training set, and **`ind[ntrain:]`** to get a test set.

The two code blocks above can be replaced by the function in the **`sklearn`** package **`model_selection train_test_split`**.[15]

```
from sklearn.model_selection import train_test_split
xtrain,xtest,ytrain,ytest=\
     train_test_split(xvals.reshape(-1,1),yvals)
```

First, we'll use our old friend **`scipy.stats.linregress`** on the training set:

```
from scipy.stats import linregress as LR
b,a,r,p,stderr=LR(xtrain,ytrain)
print(a,b,r)
```

```
-0.2168130767260834 0.9246615087756963 0.9653789783330483
```

This tells us that the regression on the training set gives (rounded to 2 decimals)

$$ERP = 0.92 - 0.22 \times PRP \tag{1.67}$$

[15] For details see https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html. The **reshape** method ensure the that x-array is a column vector.