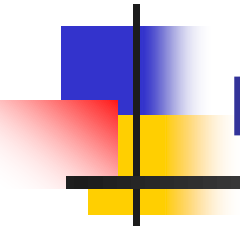
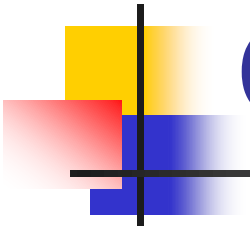
Introduction to Data Science
CS61
June 12 - July 12, 2018



Dr. Ash Pahwa

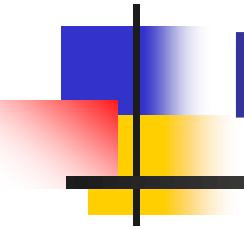
Lesson 3: Data Exploration-2

Lesson 3.2: Random Numbers and Sampling



Outline

- Random Library
- Random Functions in Numpy
- Sampling



Random Library

Random Number Generation
Import Random Library

```
import numpy as np
import matplotlib.pyplot as plt
import string

import random
```

1. Random Number Between 0 and 1

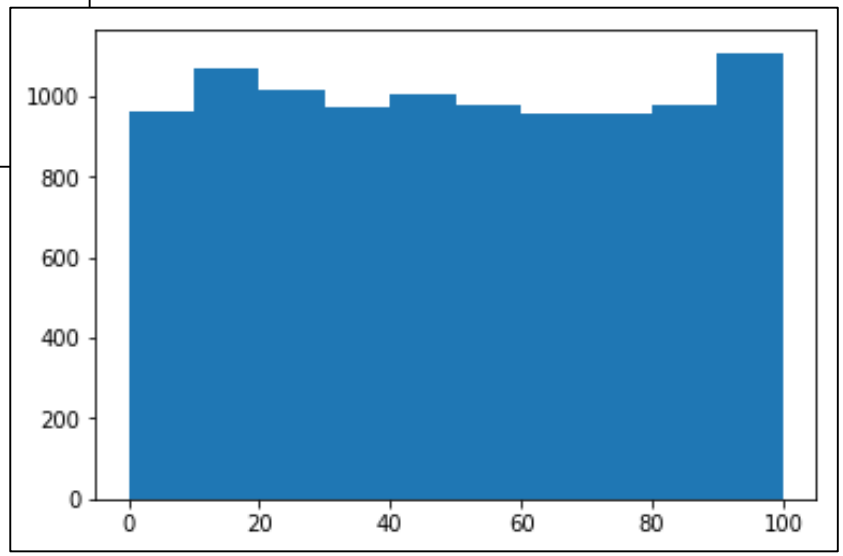
```
##########################################
# 1. Random number between 0 - 1
#
value = random.random()

print (value)
0.6064850062581212
```

2. Uniform Distribution Single Number

```
#########################################################
# 2. Single number from Uniform distribution
#
print(random.uniform(2000,2015))
2011.9341777520733

print(random.randint(0,100))
63
```

3. Uniform Distribution randint

```
#####################################
# 3. Uniform Distribution
#
ran_sample = []
for i in range(0,10000):
    sample = random.randint(0,100)
    ran_sample.append(sample)

plt.hist(ran_sample,bins=10)
```



3.1 Simulation of a Die + Toss

```
##############################################
# 2. Simulation of a Die + Toss
#
value = random.randint(1,6)
print (value)
5
#
value = random.randint(0,1)
print (value)
0
```

4. Sorting the Random Numbers

```
##############################################
# 4. Sorting the random numbers
#
ran_sample = []

for i in range(0,10):
    sample = random.randint(0,100)
    ran_sample.append(sample)

ran_sample
[96, 1, 15, 93, 83, 94, 37, 28, 97, 68]

ran_sample.sort()

ran_sample
[1, 15, 28, 37, 68, 83, 93, 94, 96, 97]
```

5. Random number with replacement
A series of random numbers with replacement

```
################################################
# 5. Random number with replacement
# A series of random numbers with replacement
#
ran_sample = []

for i in range(0,10):
    sample = int(random.uniform(2000,2015))
    ran_sample.append(sample)



print(ran_sample)
[2006, 2005, 2007, 2000, 2010, 2001, 2013, 2003, 2005, 2004]

ran_sample.sort()

print(ran_sample)
[2000, 2001, 2003, 2004, 2005, 2005, 2006, 2007, 2010, 2013]
```

6. String Set Sorting

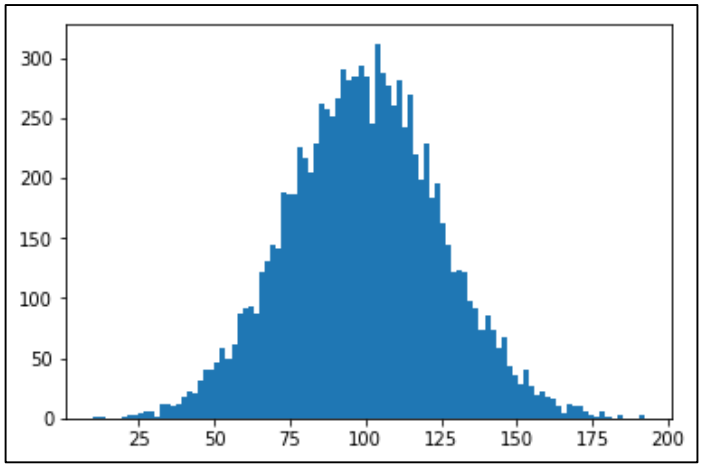
```
##################################################
# 6. String set sorting
#
my_cards = ['club','spade','heart','diamond']
random.shuffle(my_cards)
my_cards
Out[44]: ['diamond', 'spade', 'club', 'heart']

my_card = random.choice(my_cards)
print(my_card)
Heart

############################
greetings = ['Hello','Hi','Hey','Howdy','Hola']
value = random.choice(greetings)
print(value + ', Ash')
Howdy, Ash
```

7. Random Selection

```
##############################################
# 7. Random selection
#

print(random.choice(['yes','no','maybe']))
maybe
```

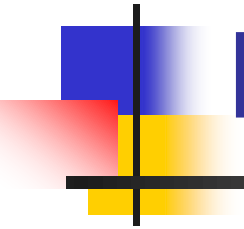
8. Gaussian Distribution Mean=100, Std=25

```
#####################################################
# 8. Gaussian Distribution
#
test_list = []

for i in range (0,10000):
    test_list.append(random.gauss(100,25))

len(test_list)
10000

sum(test_list)/10000
99.53962614911326

print(np.std(test_list))
25.1475344404

plt.hist(test_list,bins=100)
```



9. Random Password Generation

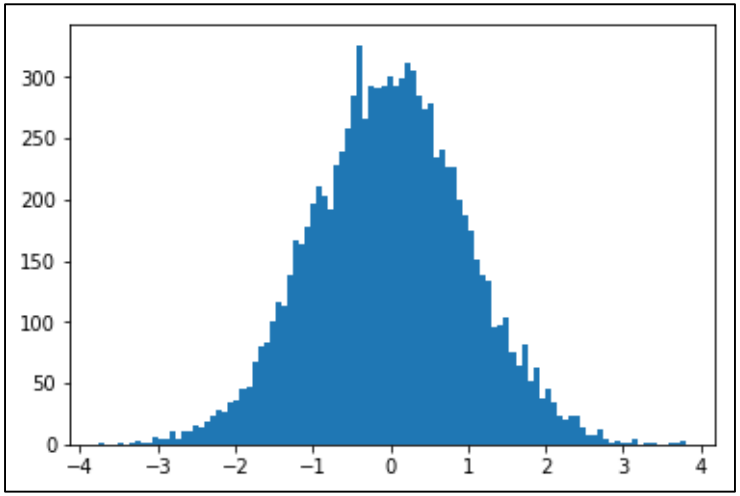
```
################################################
# 9. Random Password generation
#
print (string.digits)
0123456789

print(string.ascii_letters)
abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ

pw = ''
for i in range(0,16):
    ran_char = random.choice(string.ascii_letters + string.digits)
    pw = pw + ran_char

print(pw)
6TyFOxp1RNzyMhIX
```

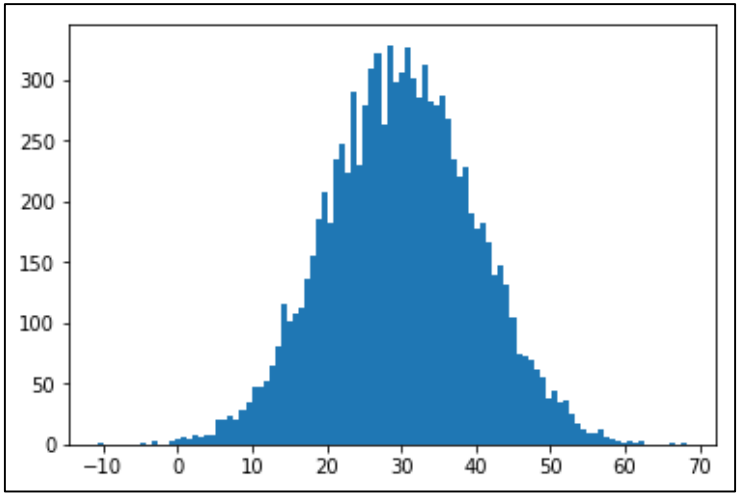


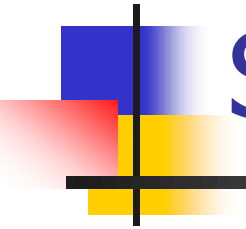
Random Functions in Numpy

10. Random Number Generation Normal Distribution

```
#####################################################
# 1. Random numbers - Normal Distribution
#
r = np.random.randn(10000)
plt.hist(r,bins=100)

r = 10*np.random.randn(10000) + 30
plt.hist(r,bins=100)
```

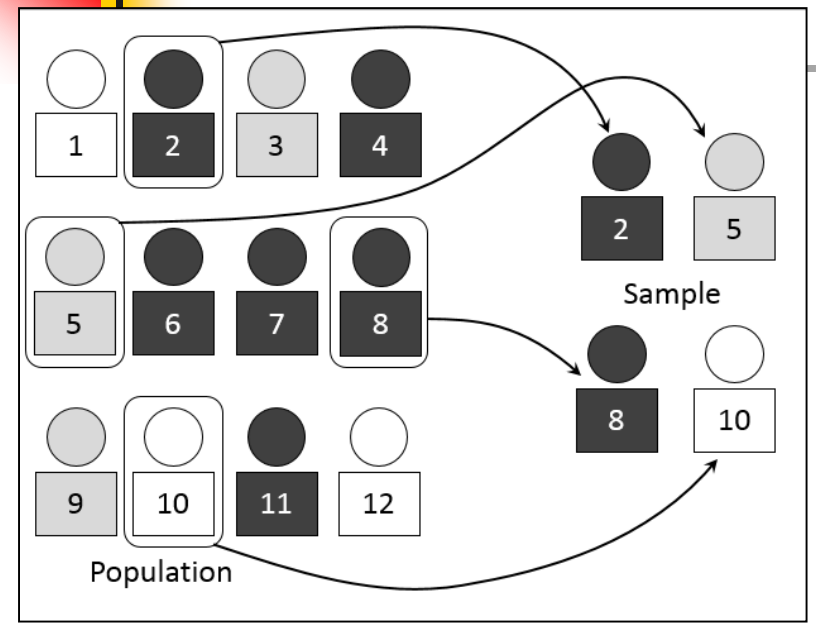


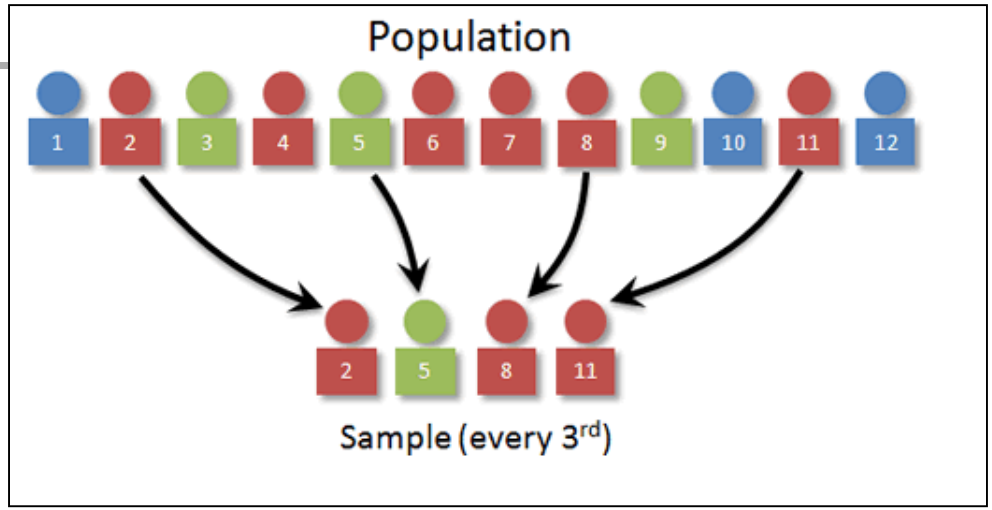


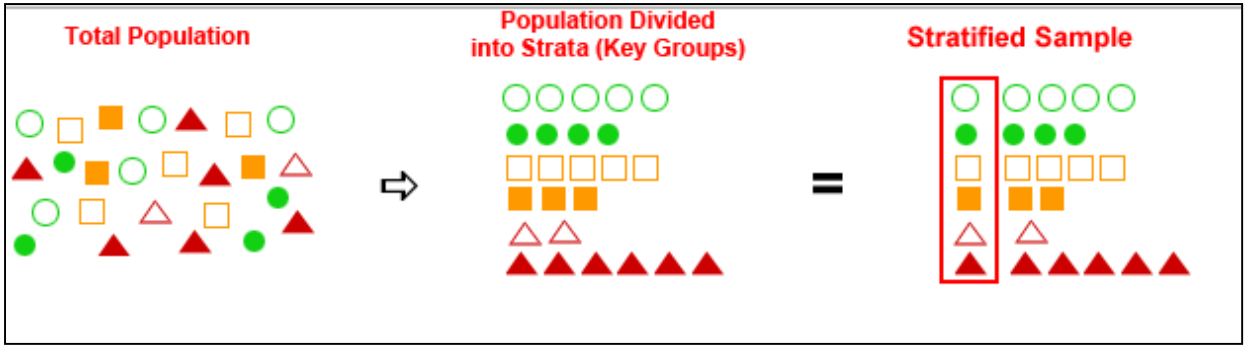


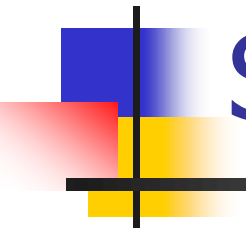
Sampling

Sampling


1
2
3
4
5
6
7
8
9
10
11
12
Population
2
5
8
10
Sample


Population
1
2
3
4
5
6
7
8
9
10
11
12
2
5
8
11
Sample (every 3rd)


Total Population
Population Divided into Strata (Key Groups)
Stratified Sample



Sampling in R

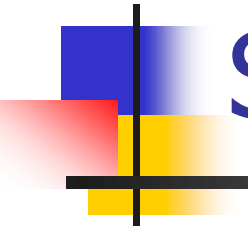
R Commands for Sampling

```
> set.seed(0)
> (x = 1:10)
 [1]  1  2  3  4  5  6  7  8  9 10
> sample(x,5)
[1]  9  3 10  5  6
>
> sample(1:10,5)
[1] 3 9 8 5 4
>
> sample(10,5)
[1] 1 2 9 5 3
>
> sample(10,5,replace=T)
[1]  8  5  8 10  4
>
```

Sampling in R

```
> DataFile = read.csv("DataFile.csv")
> length(DataFile)
[1] 2
> (nData = nrow(DataFile))
[1] 10
> DataFile
   X. Name
1   1    A
2   2    B
3   3    C
4   4    D
5   5    E
6   6    F
7   7    G
8   8    H
9   9    I
10 10    J
> set.seed(0)
> (trainIdx <- sample(seq(1, nrow(DataFile)), floor(nrow(DataFile) * 0.70)))
[1]  9  3 10  5  6  2  4
> (nTrain = length(trainIdx))
[1] 7
> (nTest = nData - nTrain)
[1] 3
>
> (yTrain <- DataFile$Name[trainIdx])
[1] I C J E F B D
Levels: A B C D E F G H I J
> (yTest <- DataFile$Name[-trainIdx])
[1] A G H
Levels: A B C D E F G H I J
```

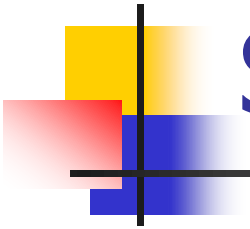
	A	B
1	#	Name
2	1	A
3	2	B
4	3	C
5	4	D
6	5	E
7	6	F
8	7	G
9	8	H
10	9	I
11	10	J



Sampling in Python

11. Sampling Without Replacement

```
################################################
# 11. Simulation of deck of cards + Sampling
#
deck = list(range(1,53))

print(deck)
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21,
22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40,
41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52]

random.shuffle(deck)

print(deck)
[28, 30, 31, 44, 47, 38, 42, 48, 16, 39, 49, 35, 9, 29, 27, 37, 19, 8, 20,
25, 23, 33, 6, 22, 51, 12, 32, 34, 21, 45, 7, 1, 3, 15, 13, 11, 10, 52, 40,
26, 24, 46, 41, 18, 36, 4, 43, 14, 50, 17, 5, 2]

hand = random.sample(deck,k=5)

print(hand)
[33, 6, 25, 42, 38]
```



Summary

- Random Library
- Random Functions in Numpy
- Sampling