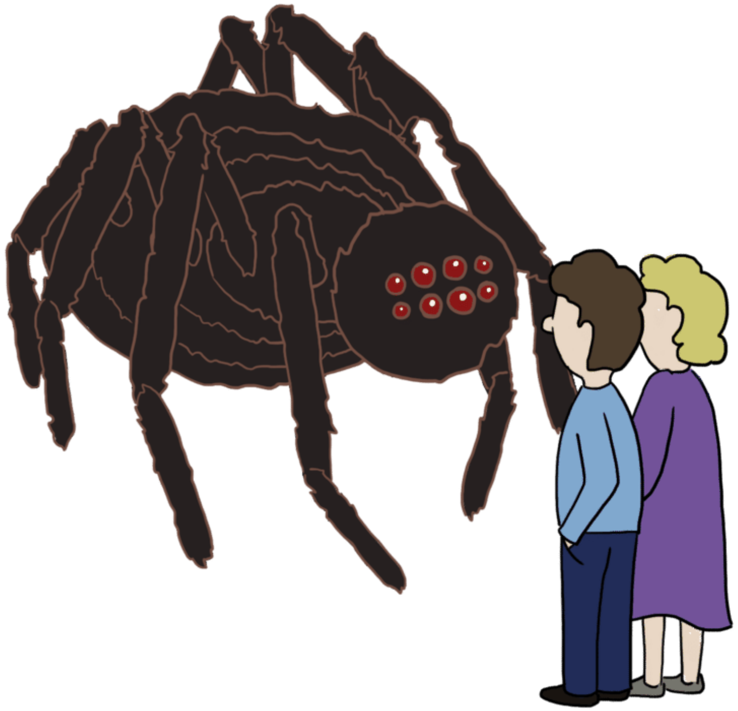
Welcome to the ‘Finale’ of DS Foundation Course!

TIME FOR TRIVIA

New York Times Puzzle

https://www.nytimes.com/interactive/2015/07/03/upshot/a-quick-puzzle-to-test-your-problem-solving.html

Hypothesis Testing



“I’ve narrowed it down to two hypothesis: it grew, or we shrunk.”

Test your hypothesis?

* Assume truly random

$P(\text{Bill not picked on a night}) = \frac{3}{4}$

$P(\text{Bill not picked 3 nights in a row}) = \frac{3}{4}\cdot\frac{3}{4}\cdot\frac{3}{4} = \frac{27}{64} = 0.42$

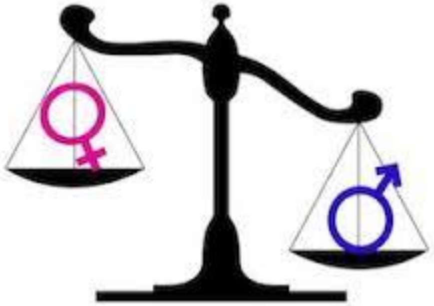
$P(\text{'' '' '' 12 '' '' '' ''}) = \left(\frac{3}{4}\right)^{12} \approx 0.032 = 3.2\%$

Some Real World Examples

You go to a petrol bunk to fill 5 liters of Petrol/Gasoline

How are you sure that fuel quantity is correct?

How can a inspector validate this assumption?



Is sex ratio in India 1:1?

How can you validate this hypothesis

Hypothesis

- **Statistical hypothesis** is an assumption about a population parameter. This assumption may or may not be true
- **Hypothesis testing** refers to the formal procedures used by statisticians to accept or reject statistical hypotheses

Types of Statistical Hypothesis

- **Null Hypothesis.** The null hypothesis, denoted by H_0, is usually the hypothesis that sample observations result purely from chance
- **Alternative hypothesis**. The alternative hypothesis, denoted by $H1$or Ha, is the hypothesis that sample observations are influenced by some non-random cause.

Is average Experience between DSS & BD different

Current Batch


```
dss_exp = np.array([12,  15,  13,  20,  19,  20,  11,  19,  11,  12,  19,  13,
                    12,  10,  6,  19,  3,  1,  1,  0,  4,  4,  6,  5,  3,  7,
                    12,  7,  9,  8,  12,  11,  11,  18,  19,  18,  19,  3,  6,
                     5,  6,  9,  11,  10,  14,  14,  16,  17,  17,  19,  0,  2,
                     0,  3,  1,  4,  6,  6,  8,  7,  7,  6,  7,  11,  11,  10,
                    11,  10,  13,  13,  15,  18,  20,  19,  1,  10,  8,  16,
                    19,  19,  17,  16,  11,  1,  10,  13,  15,  3,  8,  6,  9,
                    10,  15,  19,  2,  4,  5,  6,  9,  11,  10,  9,  10,  9,
                    15,  16,  18,  13,])
```

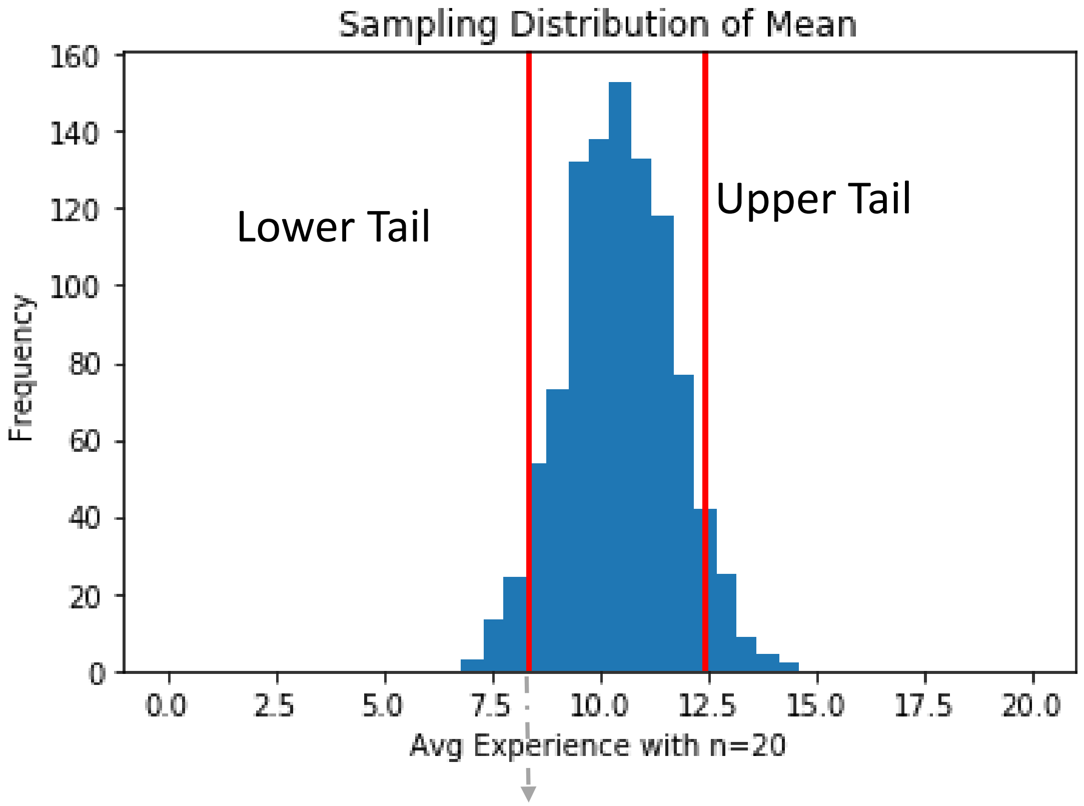
$n = 108$
$\mu = 10.43$

Previous Batch


```
dss_exp_prev = np.array([1, 14,  6,  7, 10, 10, 19, 15, 19, 15,
                         2,  2, 14, 14, 14,  3,  0,  4, 11,  7,
                         1,  2,  0,  1,  2,  2,  2,  1,  1,  2,
                         4,  4,  3,  3,  3,  3,  4,  3,  3,  7,
                         8,  6,  6,  6,  7, 8, 8, 8, 8, 7,
                         8, 0, 0, 7, 6, 9, 10, 9, 9, 11,
                         11, 9, 10, 10, 11, 10, 11, 9, 9, 9,
                         12, 14, 13, 14, 18, 14, 11, 10, 17, 20,
                         18, 5, 13, 4, 2, 4, 3, 12, 12, 14,
                         12, 12, 10, 14, 4, 11, 9])
```

$n = 97$
$\mu = 8.04$

Hypothesis Testing : n = 20, 90% Confidence Interval



Previous Batch Sample Mean = 8.25

Reject Null Hypothesis !!!

N = 20
Confidence = 90%
Significance = 0.1
T-Statistic = -1.725
P-Value = 0.10

P-value

- **P-value**. The strength of evidence in support of a null hypothesis is measured by the **P-value**.
- Suppose the test statistic is equal to *S*. The P-value is the probability of observing a test statistic as extreme as *S*, assuming the null hypothesis is true.
- If p-value is less than the level of significance we reject the null hypothesis

Calculating P-Value

Variable	Obs	Mean	Std. Dev.	Min	Max
y	90	-5.032836	3.566609	-13.51062	4.489684

The p value of a test is the probability of seeing a result *at least as extreme* as the one that you actually saw, assuming the null hypothesis is true.

In your example the null hypothesis is that $\mu = -4$. The standard test here is a two-sided t test, where we first compute the t-statistic:

$$t = \frac{\bar{x} - \mu}{s/\sqrt{n}}$$

where $\bar{x}$ is the sample mean, s is the sample standard deviation and n is the number of observations in your sample. In your data $\bar{x} = -5.033$, $s = 3.567$ and $n = 90$, so

$$t = \frac{-5.033 + 4}{3.567/\sqrt{90}} = -2.747$$

This is then compared to a t distribution with $n - 1$ degrees of freedom to calculate a p value. We want the probability that the result is *at least as extreme* as the one we saw, so we use a two-sided t test, since $t < -2.747$ and $t > 2.747$ are both considered equally extreme.

Let T_{89} be a t-distributed random variable with 89 degrees of freedom. We have

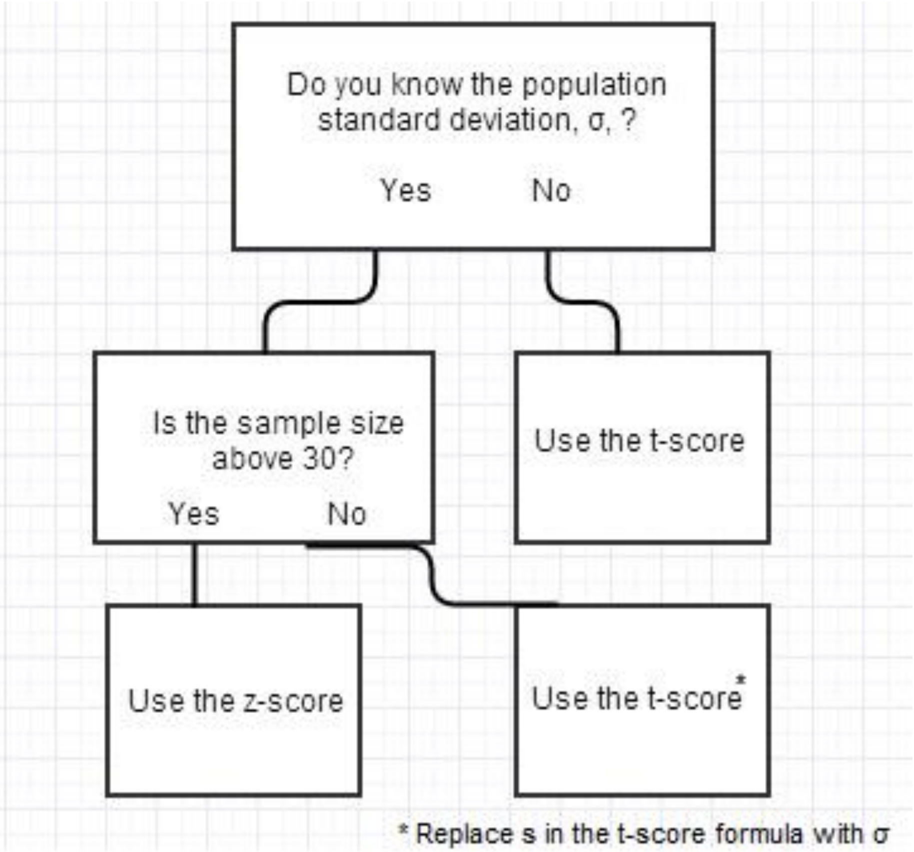
$$P(T_{89} \leq -2.747) = 0.00364$$

and $P(T_{89} \geq 2.747)$ will be the same since the t distribution is symmetric, which means that your p-value is

$$p = 2 \times 0.00364 = 0.00727$$

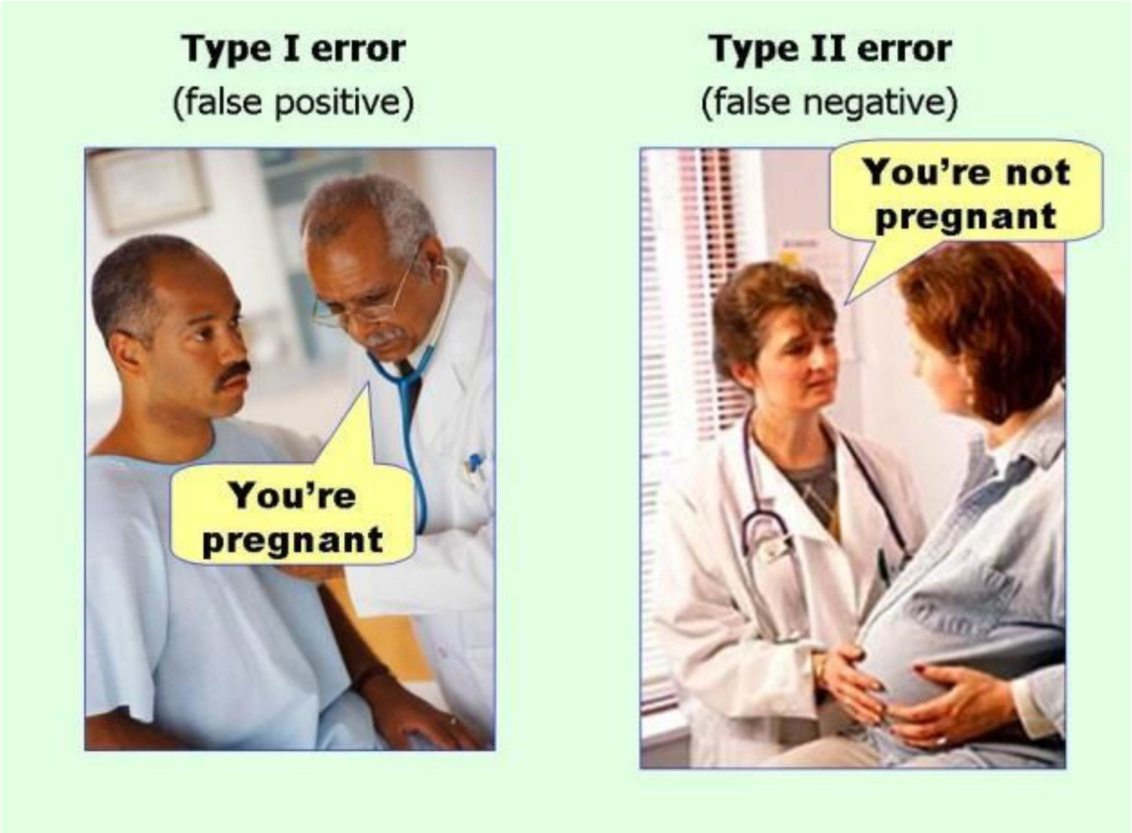
so your null hypothesis is rejected at the 1% significance level.

When to use Z-score vs T-score?



Decision Errors

- **Type I error**: Rejecting a null hypothesis when it is true.
- **Type II error**: “accept” or fails to reject the null hypothesis when it is false.



Putting it all in perspective

	H_0 True	H_0 False
Reject H_0	Type I Error	Correct Rejection
Fail to Reject H_0	Correct Decision	Type II Error

Recap

Python Basics

upx Move up in life

Variables and Data Types

Variable Assignment

```
>>> x=5
>>> x
5
```

Calculations With Variables

`>>> x+2` → 7	Sum of two variables
`>>> x-2` → 3	Subtraction of two variables
`>>> x*2` → 10	Multiplication of two variables
`>>> x**2` → 25	Exponentiation of a variable
`>>> x%2` → 1	Remainder of a variable
`>>> x/float(2)` → 2.5	Division of a variable

Types and Type Conversion

str()	'5', '3.45', 'True'	Variables to strings
int()	5, 3, 1	Variables to integers
float()	5.0, 1.0	Variables to floats
bool()	True, True, True	Variables to booleans

String Operations

Index starts at 0

```
>>> my_string[3]
>>> my_string[4:9]
```

String Methods

`>>> my_string.upper()`	String to uppercase
`>>> my_string.lower()`	String to lowercase
`>>> my_string.count('w')`	Count String elements
`>>> my_string.replace('e', 'i')`	Replace String elements
`>>> my_string.strip()`	Strip whitespaces

Selecting List Elements

Index starts at 0

Subset

`>>> my_list[1]`	Select item at index 1
`>>> my_list[-3]`	Select 3rd last item

Slice

`>>> my_list[1:3]`	Select items at index 1 and 2
`>>> my_list[1:]`	Select items after index 0
`>>> my_list[:3]`	Select items before index 3
`>>> my_list[:]`	Copy my_list

List Operations

```
>>> my_list + my_list
['my', 'list', 'is', 'nice', 'my', 'list', 'is', 'nice']
>>> my_list * 2
['my', 'list', 'is', 'nice', 'my', 'list', 'is', 'nice']
>>> my_list2 > 4
True
```

List Methods

`>>> my_list.index(a)`	Get the index of an item
`>>> my_list.count(a)`	Count an item
`>>> my_list.append('!')`	Append an item at a time
`>>> my_list.remove('!')`	Remove an item
`>>> del(my_list[0:1])`	Remove an item
`>>> my_list.reverse()`	Reverse the list
`>>> my_list.extend('!')`	Append an item
`>>> my_list.pop(-1)`	Remove an item
`>>> my_list.insert(0,'!')`	Insert an item
`>>> my_list.sort()`	Sort the list

NumPy

NumPy

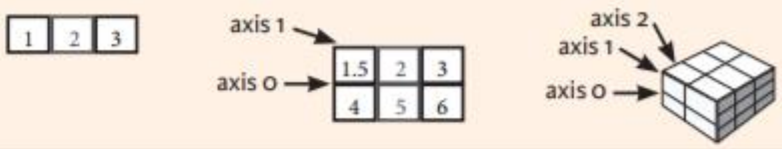
The **NumPy** library is the core library for scientific computing in Python. It provides a high-performance multidimensional array object, and tools for working with these arrays.

Use the following import convention:

```
>>> import numpy as np
```

NumPy Arrays

1D array 2D array 3D array



Creating Arrays

```
>>> a = np.array([1,2,3])
>>> b = np.array([(1.5,2,3), (4,5,6)], dtype = float)
>>> c = np.array([[(1.5,2,3), (4,5,6)], [(3,2,1), (4,5,6)]],
                 dtype = float)
```

Initial Placeholders

`>>> np.zeros((3,4))`	Create an array of zeros
`>>> np.ones((2,3,4),dtype=np.int16)`	Create an array of ones
`>>> d = np.arange(10,25,5)`	Create an array of evenly spaced values (step value)
`>>> np.linspace(0,2,9)`	Create an array of evenly spaced values (number of samples)
`>>> e = np.full((2,2),7)`	Create a constant array
`>>> f = np.eye(2)`	Create a 2X2 identity matrix
`>>> np.random.random((2,2))`	Create an array with random values
`>>> np.empty((3,2))`	Create an empty array

Numpy Arrays

```
>>> my_list = [1, 2, 3, 4]
>>> my_array = np.array(my_list)
>>> my_2darray = np.array([[1,2,3],[4,5,6]])
```

Selecting Numpy Array Elements

Index starts at 0

Subset
`>>> my_array[1]`
`2`
	Select item at index 1

Slice
`>>> my_array[0:2]`
`array([1, 2])`
	Select items at index 0 and 1

Subset 2D Numpy arrays
`>>> my_2darray[:,0]`
`array([1, 4])`
	my_2darray[rows, columns]

Numpy Array Operations

```
>>> my_array > 3
  array([False, False, False,  True], dtype=bool)
>>> my_array * 2
  array([2, 4, 6, 8])
>>> my_array + np.array([5, 6, 7, 8])
  array([6, 8, 10, 12])
```

Numpy Array Functions

`>>> my_array.shape`	Get the dimensions of the array
`>>> np.append(other_array)`	Append items to an array
`>>> np.insert(my_array, 1, 5)`	Insert items in an array
`>>> np.delete(my_array,[1])`	Delete items in an array
`>>> np.mean(my_array)`	Mean of the array
`>>> np.median(my_array)`	Median of the array
`>>> my_array.corrcoef()`	Correlation coefficient
`>>> np.std(my_array)`	Standard deviation

Acing the ‘Axes’

Axis = 0 (Columns)
Axis=1(Rows)

- Mean()
- Sum()
- Describe()
- Count()
- Sort_index(by=)
- Fillna(method = ffill)

Axis = 0 (Rows)
Axis=1(Columns)

- dropna()
- Apply()
- Cumsum()
- Drop()
- Concat()

Pandas

1. Reading and Writing Data

a. Reading a CSV file

```
>>>df=pd.read_csv('AnalyticsVidhya.csv')
```

b. Writing content of data frame to CSV file

```
>>>df.to_csv('AV.csv')
```

c. Reading an Excel file

```
>>>df=pd.read_excel('AV.xlsx','sheet1')
```

d. Writing content of data frame to Excel file

```
>>>df.to_excel('AV2.xlsx',sheet_name='sheet2')
```

Pandas

2.Getting Preview of Dataframe

a. Looking at top n records

```
>>>df.head(5)
```

b. Looking at bottom n records

```
>>>df.tail(5)
```

c. View columns name

```
>>>df.columns
```

Pandas

3. Rename Columns of Data Frame

a. Rename method helps to rename column of data frame.

```
>>>df2=df.rename(columns={'old_columnname':'new_columnname'})
```

This statement will create a new data frame with new column name.

b. To rename the column of existing data frame, set inplace=True.

```
>>>df.rename(columns={'old_columnname':'new_columnname'}, inplace=True)
```

Pandas

4. Selecting Columns or Rows

a. Accessing sub data frames

```
>>>df[['column1','column2']]
```

b. Filtering Records

```
>>>df[ df['column1']>10]
>>>df[ (df['column1']>10) & df['column2']==30]
>>>df[ (df['column1']>10) | df['column2']==30]
```

Pandas

5. Handling Missing Values

This is an inevitale part of dealing with data . To overcome this hurdle, use dropna or fillna function.

a. dropna: It is used to drop rows or columns having missing data

```
>>>df1.dropna()
```

b. fillna: It is used to fill missing values

```
>>>df2.fillna(value=5) #It replaces all missing values with 5
>>>mean=df2['column1'].mean()
>>>df2['column1'].fillna(mean) #It replaces all missing values of column1 with mean of available values
```

Pandas

6. Creating New Columns

New column is a function of existing columns

```
>>>df['NewColumn1']=df['column2']
```
#Create a copy of existing column2

```
>>>df['NewColumn2']=df['column2']+10
```
#Add 10 to existing column2 then create a new one

```
>>>df['NewColumn3']= df['column1'] + df['column2']
```
#Add elements of column1 and column2 then create new column

Pandas

7. Aggregate

a. Groupby: Groupby helps to perform three operations

i. **Splitting the data into groups**
ii. **Applying a function to each group individually**
iii. **Combining the result into a data structure**

```
>>>df.groupby('column1').sum()
>>>df.groupby(['column1','column2']).count()
```

b. Pivot Table: It helps to generate data structure. It has three components index, columns and values (similar to excel)

```
>>>pd.pivot_table(df, values='column1', index=['column2','column3'], columns=['column4'])
```

By default, it shows the sum of values column but you can change it using argument aggfunc

```
>>>pd.pivot_table(df, values='column1', index=['column2','column3'], columns=['column4'], aggfunc=len)
```

#it shows count

Pandas

8. Merging/ Concatenating DataFrames

It performs similar operation like we do in SQL.

a. Concatenating: It concatenate two or more data frames based on their columns.

```
>>>pd.concat([df1,df2])
```

b. Merging: We can perform left, right and inner join also.

```
>>>pd.merge(df1, df2, on='column1', how='inner')
>>>pd.merge(df1, df2, on='column1', how='left')
>>>pd.merge(df1, df2, on='column1', how='right')
>>>pd.merge(df1, df2, on='column1', how='outer')
```

Pandas

9. Applying function to element, column or dataframe

a. **Map: It iterates over each element of a series.**

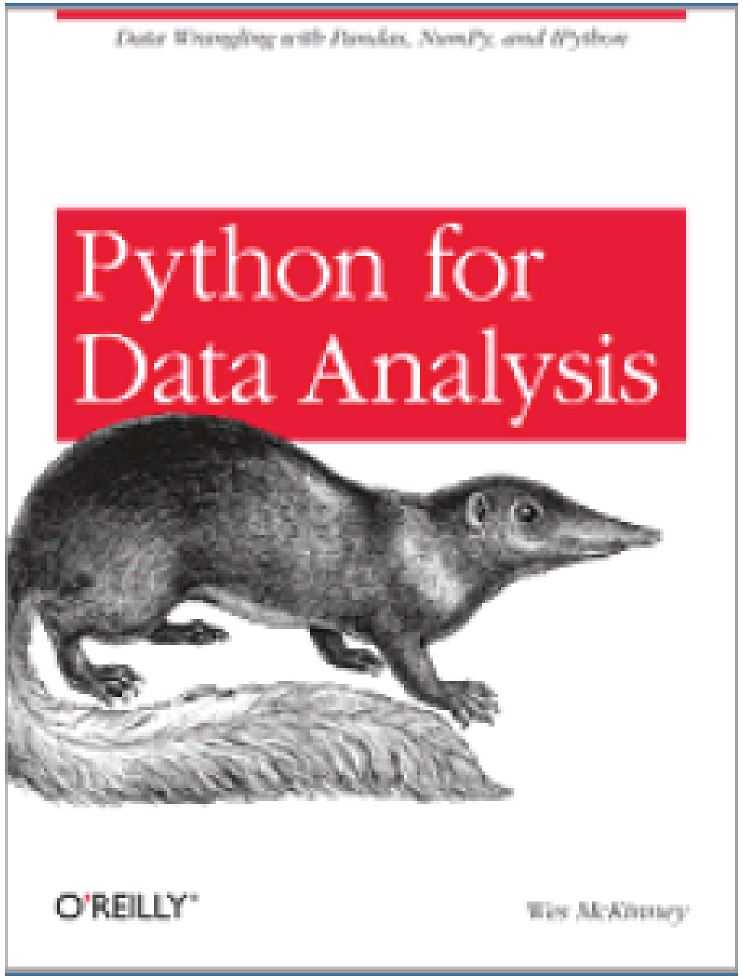
```
>>>df['column1'].map(lambda x: 10+x)
```
#this will add 10 to each element of column1

```
>>>df['column2'].map(lambda x: 'AV'+x)
```
#this will concatenate "AV" at the beginning of each element of column2 (column format is string)

b. **Apply: As the name suggests, applies a function along any axis of the DataFrame.**

```
>>>df[['column1','column2']].apply(sum)
```
#it will returns the sum of all the values of column1 and column2.

Pandas



Matplotlib vs Seaborn

- Matplotlib to be used for basic plotting – bar charts, line graphs, scatter plots
- Matplotlib visualizations useful for quick prototyping
- Seaborn usually used for statistical visualizations – heatmaps, pairplots, box plots
- Seaborn visualizations are aesthetically more appealing

Statistics

DEFINITIONS

- **STATISTICS** - A set of tools for collecting, organizing, presenting, and analyzing numerical facts or observations.
  1. **Descriptive Statistics** - procedures used to organize and present data in a convenient, useable, and communicable form.
  2. **Inferential Statistics** - procedures employed to arrive at broader generalizations or inferences from sample data to populations.
- **STATISTIC** - A number describing a sample characteristic. Results from the manipulation of sample data according to certain specified procedures.
- **DATA** - Characteristics or numbers that are collected by observation.
- **POPULATION** - A complete set of actual or potential observations.
- **PARAMETER** - A number describing a population characteristic; typically, inferred from sample statistic.
- **SAMPLE** - A subset of the population selected according to some scheme.
- **RANDOM SAMPLE** - A subset selected in such a way that each member of the population has an equal opportunity to be selected. **Ex.** ***lottery numbers in a fair lottery***
- **VARIABLE** - A phenomenon that may take on different values.

- **MEAN** - The point in a distribution of measurements about which the summed deviations are equal to zero. ***Average value of a sample or population.***

POPULATION MEAN **SAMPLE MEAN**

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i \qquad \bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Note: The mean is very sensitive to extreme measurements that are not balanced on both sides.

- **WEIGHTED MEAN** - Sum of a set of observations multiplied by their respective weights, divided by the sum of the weights:

$$\text{WEIGHTED MEAN} = \frac{\sum_{i=1}^{G} w_i x_i}{\sum_{i=1}^{G} w_i}$$

where w_i = weight; x_i = observation; G = number of observation groups. Calculated from a population, sample, or groupings in a frequency distribution.

Ex. ***In the FrequencyDistribution below, the mean is 80.3; calculated by using frequencies for the w_i's. When grouped, use class midpoints for x_i's.***

- **MEDIAN** - Observation or potential observation in a set that divides the set so that the same number of observations lie on each side of it. For an odd number of values, it is the middle value; for an even number it is the average of the middle two.

Ex. ***In the Frequency Distribution table below, the median is 79.5.***

- **MODE** - Observation that occurs with the greatest frequency. **Ex.** ***In the Frequency Distribution table below, the mode is 88.***

MEASURES OF DISPERSION

- **SUM OF SQUARES *(SS)*** - Deviations from the mean, squared and summed:

Population SS $= \Sigma(x_i - \mu_x)^2$ or $\Sigma x_i^2 - \frac{(\Sigma x_i)^2}{N}$

Sample SS $= \Sigma(x_i - \bar{x})^2$ or $\Sigma x_i^2 - \frac{(\Sigma x_i)^2}{n}$

- **VARIANCE** - The average of square differences between observations and their mean.

POPULATION VARIANCE **SAMPLE VARIANCE**

$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i-\mu)^2 \qquad s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i-\bar{x})^2$$

VARIANCES FOR GROUPED DATA

POPULATION **SAMPLE**

$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{G} f_i(m_i-\mu)^2 \qquad s^2 = \frac{1}{n-1}\sum_{i=1}^{G} f_i(m_i-\bar{x})^2$$

- **STANDARD DEVIATION** - Square root of the variance:

Ex. ***Pop. S.D.*** $\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i-\mu)^2}$

Statistics

- Population and Sample
- Central Limit Theorem
- Z-scores
- Normal Distribution
- Hypothesis Testing

Parting Thoughts

The one attribute that will define your success as a Data Scientist

Do you know what is so special about this guy?

Leonardo Da Vinci

Invention

Painting

Sculpting

Architecture

Science

Music

Specialty : Curiosity

Mathematics

Literature

Anatomy

Geology

Astronomy

Cartography

You too need to be curious…….

Programming

Machine Learning

Statistics

Big Data

Deep Learning

Design Thinking

Your Name

Communication Skills

IoT

Blockchain

Augmented Reality

Empirical Research

Domain Expertise

The Way Forward

- Master all the concepts we have covered in our 12 sessions
- Complete the project
- Explore the two books listed under ‘Books to Refer’ section in LMS
- Do not stop questioning ‘Why’ while learning anything new
- Most important of them all – stay in touch with each other and always share your learnings!

Wish You All the Best For Your Future