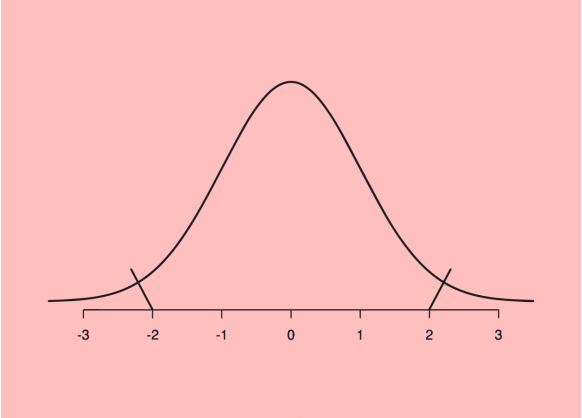
Single sample z-test



What is the GOAL for this MODULE?

The goal for this module is to explore the aspects of the data visualization.

What did we ACHIEVE in the class TODAY?

- We learned about the sampling distribution.
- We learned about the z -test of a single sample.

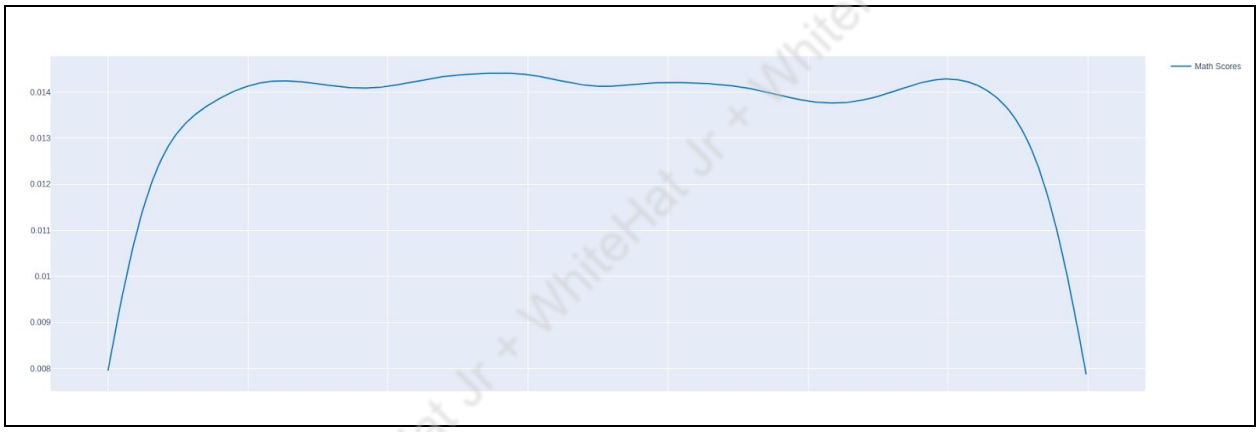
Which CONCEPTS/CODING BLOCKS did we cover today?

- Finding mean of the population
- Finding mean of the population
- Finding all 3 deviations
- Performing the z-test

How did we DO the activities?

1. We got data of marks of 1000 students in a math test and then we plotted it.

```
main.py > [e] mean
1  import plotly.figure_factory as ff
2  import plotly.graph_objects as go
3  import statistics
4  import random
5  import pandas as pd
6  import csv
7
8  df = pd.read_csv("studentMarks.csv")
9  data = df["Math_score"].tolist()
10
11 #plotting the graph
12 fig = ff.create_distplot([data],["Math Scores"], show_hist= False)
13 fig.show()
14
```



2. We found the standard deviation and sampling mean of 100 students 1000 times and plotted it.

```
titled-1    main.py
ain.py > ...
import plotly.figure_factory as ff
import plotly.graph_objects as go
import statistics
import random
import pandas as pd
import csv

df = pd.read_csv("studentMarks.csv")
data = df["Math_score"].tolist()

#plotting the graph
fig = ff.create_distplot([data],["Math Scores"], show_hist= False)
fig.show()

#calculating the mean and standard deviation of the population data
mean = statistics.mean(data)
std_deviation = statistics.stdev(data)
print("mean of popultion:- ",mean)
print("Standard deviation of popultion:- ",std_deviation)
```

```
mean of popultion:-  64.908
Standard deviation of popultion:-  20.418311064891586
```

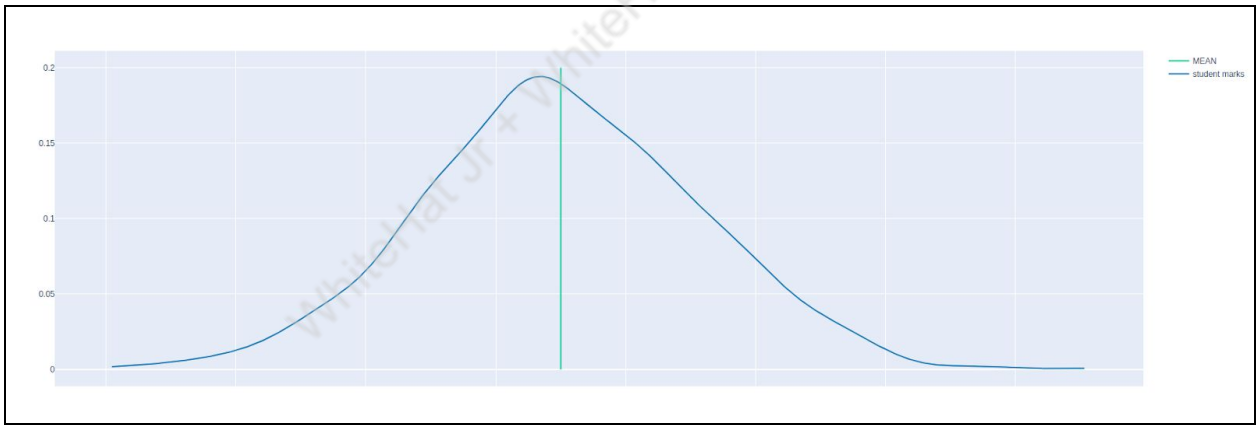
3. We plotted the trace of sampling mean on the graph.

```python
main.py > ...
##  code to find the mean of 100 data points 1000 times
#function to get the mean of the given data samples
# pass the number of data points you want  as counter
def random_set_of_mean(counter):
    dataset = []
    for i in range(0, counter):
        random_index= random.randint(0,len(data)-1)
        value = data[random_index]
        dataset.append(value)
    mean = statistics.mean(dataset)
    return mean


# Pass the number of time you want the mean of the data points as a parameter in range function in for loop
mean_list = []
for i in range(0,1000):
    set_of_means= random_set_of_mean(100)
    mean_list.append(set_of_means)

## calculating mean and standard_deviation of the sampling distribution.
std_deviation = statistics.stdev(mean_list)
mean = statistics.mean(mean_list)
print("mean of sampling distribution:- ",mean)

#plotting the mean of the sampling
fig = ff.create_distplot([mean_list], ["student marks"], show_hist=False)
fig.add_trace(go.Scatter(x=[mean, mean], y=[0, 0.20], mode="lines", name="MEAN"))
fig.show()
```



4. Then compared the population mean and sampling mean and standard deviation deviation and found that followed the properties of sampling distribution.

```python
#calculating the mean and standard deviation of the population data
mean = statistics.mean(data)
std_deviation = statistics.stdev(data)
print("mean of popultion:- ",mean)
print("Standard deviation of popultion:- ",std_deviation)
```

```
## calculating mean and standard_deviation of the sampling distribution.
std_deviation = statistics.stdev(mean_list)
mean = statistics.mean(mean_list)
print("mean of sampling distribution:- ",mean)
print("Standard deviation of sampling distribution:- ", std_deviation)
```

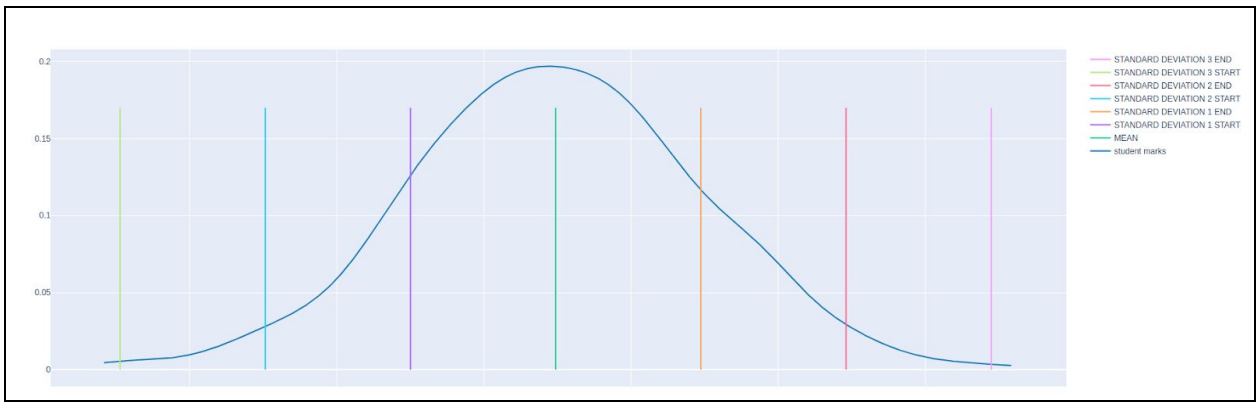
```
mean of poultion:-  64.908
Standard deviation of poultion:-  20.418311064891586
mean of sampling distribution:-  64.88579
Standard deviation of sampling distribution:-  2.013784920393835
```

5. Then we found the 1 standard deviation,2nd standard deviation and 3rd standard deviation and added it's traces on the plot.

```
## findig the standard deviation starting and ending values
first_std_deviation_start, first_std_deviation_end = mean-std_deviation, mean+std_deviation
second_std_deviation_start, second_std_deviation_end = mean-(2*std_deviation), mean+(2*std_deviation)
third_std_deviation_start, third_std_deviation_end = mean-(3*std_deviation), mean+(3*std_deviation)
print("std1",first_std_deviation_start, first_std_deviation_end)
print("std2",second_std_deviation_start, second_std_deviation_end)
print("std3",third_std_deviation_start,third_std_deviation_end)
```

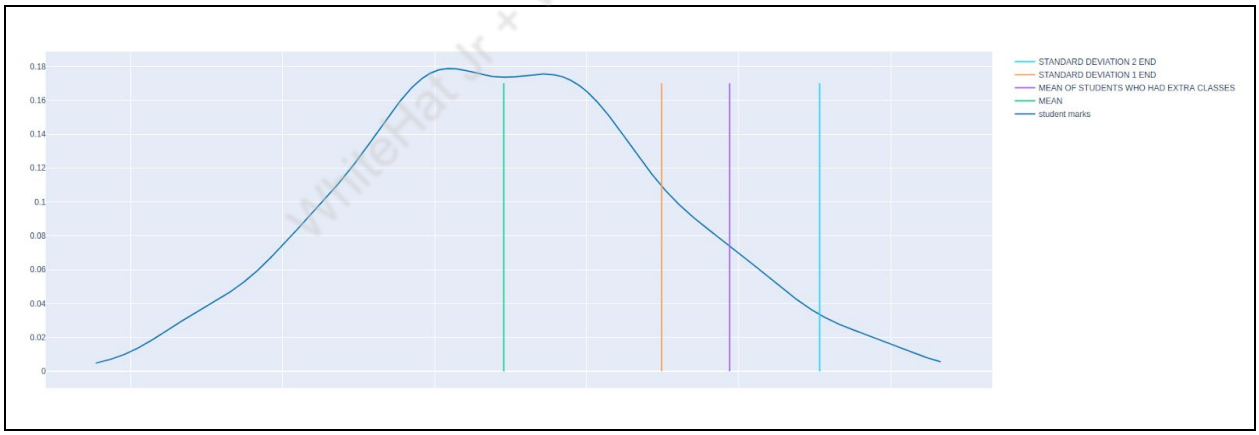
```
## plotting the graph with traces
fig = ff.create_distplot([mean_list], ["student marks"], show_hist=False)
fig.add_trace(go.Scatter(x=[mean, mean], y=[0, 0.17], mode="lines", name="MEAN"))
fig.add_trace(go.Scatter(x=[first_std_deviation_start, first_std_deviation_start], y=[0, 0.17], mode="lines
fig.add_trace(go.Scatter(x=[first_std_deviation_end, first_std_deviation_end], y=[0, 0.17], mode="lines", r
fig.add_trace(go.Scatter(x=[second_std_deviation_start, second_std_deviation_start], y=[0, 0.17], mode="lir
fig.add_trace(go.Scatter(x=[second_std_deviation_end, second_std_deviation_end], y=[0, 0.17], mode="lines",
fig.add_trace(go.Scatter(x=[third_std_deviation_start,third_std_deviation_start], y=[0,0.17], mode="lines",
fig.add_trace(go.Scatter(x=[third_std_deviation_end,third_std_deviation_end], y=[0,0.17], mode="lines", nam
fig.show()
```

```
t marks"], show_hist=False)
 0.17], mode="lines", name="MEAN"))
n_start, first_std_deviation_start], y=[0, 0.17], mode="lines", name="STANDARD DEVIATION 1 START"))
n_end, first_std_deviation_end], y=[0, 0.17], mode="lines", name="STANDARD DEVIATION 1 END"))
on_start, second_std_deviation_start], y=[0, 0.17], mode="lines", name="STANDARD DEVIATION 2 START"))
on_end, second_std_deviation_end], y=[0, 0.17], mode="lines", name="STANDARD DEVIATION 2 END"))
n_start,third_std_deviation_start], y=[0,0.17], mode="lines", name="STANDARD DEVIATION 3 START"))
n_end,third_std_deviation_end], y=[0,0.17], mode="lines", name="STANDARD DEVIATION 3 END"))
```

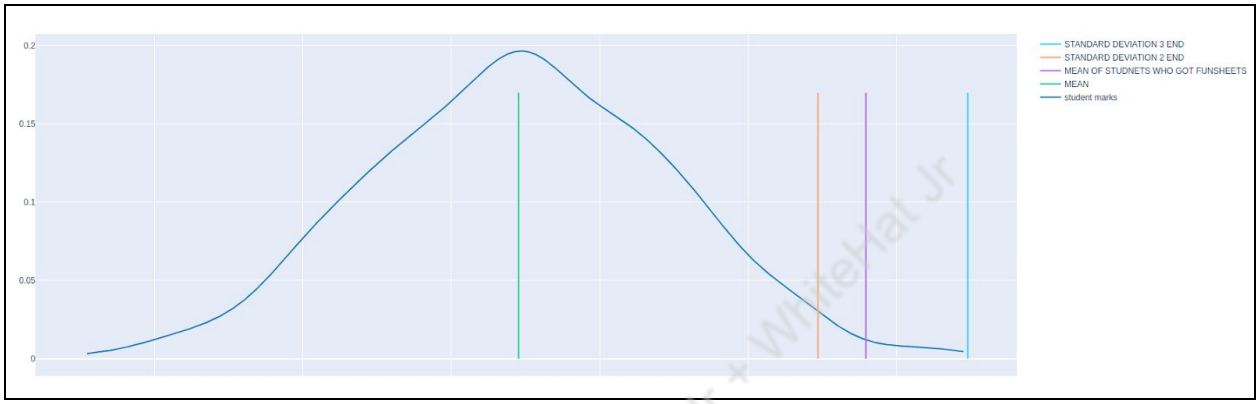


6. We plotted the sample mean and saw in which deviation it lies.

```
# finding the mean of the SECOND data (STUDENTS WHO HAD EXTRA CLASSES ) and plotting it on the plot.
df = pd.read_csv("data2.csv")
data = df["Math_score"].tolist()
mean_of_sample2 = statistics.mean(data)
print("mean of sample 2:- ",mean_of_sample2)
fig = ff.create_distplot([mean_list], ["student marks"], show_hist=False)
fig.add_trace(go.Scatter(x=[mean, mean], y=[0, 0.17], mode="lines", name="MEAN"))
fig.add_trace(go.Scatter(x=[mean_of_sample2, mean_of_sample2], y=[0, 0.17], mode="lines", name="MEAN OF SA
fig.add_trace(go.Scatter(x=[first_std_deviation_end, first_std_deviation_end], y=[0, 0.17], mode="lines", r
fig.add_trace(go.Scatter(x=[second_std_deviation_end, second_std_deviation_end], y=[0, 0.17], mode="lines",
fig.show()
```



```
# finding the mean of the THIRD data (STUDENTS WHO GOT FUNSHEET) and plotting it on the plot.
df = pd.read_csv("data3.csv")
data = df["Math_score"].tolist()
mean_of_sample3 = statistics.mean(data)
print("mean of sample3:- ",mean_of_sample3)
fig = ff.create_distplot([mean_list], ["student marks"], show_hist=False)
fig.add_trace(go.Scatter(x=[mean, mean], y=[0, 0.17], mode="lines", name="MEAN"))
fig.add_trace(go.Scatter(x=[mean_of_sample3, mean_of_sample3], y=[0, 0.17], mode="lines", name="MEAN OF STU
fig.add_trace(go.Scatter(x=[second_std_deviation_end, second_std_deviation_end], y=[0, 0.17], mode="lines",
fig.add_trace(go.Scatter(x=[third_std_deviation_end, third_std_deviation_end], y=[0, 0.17], mode="lines", r
fig.show()
```



7. Then we took the math scores of the 3 different schools, calculated the mean and standard deviation of the data and drew the trace lines at the mean for all 3 deviations.

z-score.py > ...

```
import plotly.figure_factory as ff
import plotly.graph_objects as go
import statistics
import random
import pandas as pd
import csv

#Change the School data here
df = pd.read_csv("School3.csv")
data = df["Math_score"].tolist()


##  code to find the mean of 100 data points 1000 times
#function to get the mean of the given data samples
# pass the number of data points you want  as counter
def random_set_of_mean(counter):
    dataset = []
    for i in range(0, counter):
        random_index= random.randint(0,len(data)-1)
        value = data[random_index]
        dataset.append(value)
    mean = statistics.mean(dataset)
    return mean


# Function to get the mean of 100 data sets
mean_list = []
for i in range(0,1000):
    set_of_means= random_set_of_mean(100)
    mean_list.append(set_of_means)

```

```
## findig the standard deviation starting and ending values
first_std_deviation_start, first_std_deviation_end = mean-std_deviation, mean+std_deviation
second_std_deviation_start, second_std_deviation_end = mean-(2*std_deviation), mean+(2*std_deviation)
third_std_deviation_start, third_std_deviation_end = mean-(3*std_deviation), mean+(3*std_deviation)
# print("std1",first_std_deviation_start, first_std_deviation_end)
# print("std2",second_std_deviation_start, second_std_deviation_end)
# print("std3",third_std_deviation_start,third_std_deviation_end)




# finding the mean of THE STUDENTS WHO GAVE EXTRA TIME TO MATH LAB and plotting on graph
df = pd.read_csv("School_1_Sample.csv")
data = df["Math_score"].tolist()
mean_of_sample1 = statistics.mean(data)
print("Mean of sample1:- ",mean_of_sample1)
fig = ff.create_distplot([mean_list], ["student marks"], show_hist=False)
fig.add_trace(go.Scatter(x=[mean, mean], y=[0, 0.17], mode="lines", name="MEAN"))
fig.add_trace(go.Scatter(x=[mean_of_sample1, mean_of_sample1], y=[0, 0.17], mode="lines", name="MEAN OF STU
fig.add_trace(go.Scatter(x=[first_std_deviation_end, first_std_deviation_end], y=[0, 0.17], mode="lines", r
fig.add_trace(go.Scatter(x=[second_std_deviation_end, second_std_deviation_end], y=[0, 0.17], mode="lines",
fig.add_trace(go.Scatter(x=[third_std_deviation_end, third_std_deviation_end], y=[0, 0.17], mode="lines", r
fig.show()



#finding the mean of the STUDENTS WHO USED MATH PRACTISE APP and plotting it on the plot.
# df = pd.read_csv("School_2_Sample.csv")
# data = df["Math_score"].tolist()
# mean_of_sample2 = statistics.mean(data)
# print("mean of sample 2:- ",mean_of_sample2)
```

8. We calculated the z score using a formula.

```
#finding the z score using the formula
z_score = (mean_of_sample1 - mean)/std_deviation
print("The z score is = ",z_score)
```

```
mean of sampling distribution:-  50.69924
Standard deviation of sampling distribution:-  2.879529182125215
Mean of sample1:-  50.41
The z score is =  -0.10044697646944323
```

```
#finding the z score using the formula
z_score = (mean_of_sample2 - mean)/std_deviation
print("The z score is = ",z_score)
```

```
mean of sampling distribution:-  49.75977
Standard deviation of sampling distribution:-  2.8632207529598865
mean of sample 2:-  55.33
The z score is =  1.9454420321037795
```

```
#finding the z score using the formula
z_score = (mean_of_sample3 - mean)/std_deviation
print("The z score is = ",z_score)
```

```
mean of sampling distribution:-  50.02569
Standard deviation of sampling distribution:-  2.9773998780704503
mean of sample3:-  57.29
The z score is =  2.4398167184408397
```

What's NEXT?

In the next class, we will learn more about the z test of multiple samples. Our next class will be the capstone class so don't forget to bring your parents to the class.

Extend your knowledge:

You can watch the video to learn about z tests hypothesis in depth
https://youtu.be/HoqzIR8xj4s