

probability distribution let we have a data 10000 customers - Age(min=12, max=96) 0% = 12 5% = 14 10% = 18 20% = 22 25% = 24.6 50% = 36 75% = 58 90% = 66 95% = 72 99% = 88 100 = 96 $p(\text{age} < 66) : 90\%$ $p(\text{age} > 72) : 100\%$ - $p(\text{age} < 72) = 100 - 95 = 5\%$ $p(\text{age} > 22 \text{ and } \text{age} < 66) = 70\%$ what if the $p(\text{age} > 25)$ $p(\text{age} > 30 \text{ and } \text{age} < 60)$ bins freq ----- 12-14 (smaller interval binning) let 5 14-16 3 16-18 8 18-20 9 20-22 11 92-94 2 94-96 1 we draw a bar graph binning intervals in x axis and frequency in y axis we draw a curve which roughly connects all points This is how Histogram is generated we get $f(x)$ curve the integration of the $f(x)$ will give us area under the curve ----- $f(x)$:probability Distribution function - There are ~20 commonly occurring probability functions -Almost 99% of the data follows either one of those 20 This is an emirical rule we have those $f(x)$ for the PDcurves, so we can apply that to the suitable data we have also defined standard scales and corresponding probability % whatever is the data points and its distribution , we can convert data into its standard form E.g., If age follows exponential curve can be converted to standard exp curve ----- Normal Distrubtion -displot gives bell curve let say we have data points whose size is > 30 Tendence ~68% of the data lies between mean \pm sd LC = mean -sd UC = mean +sd data[data > LC | data < UC] mean \pm sd : 68% mean \pm 2*sd : 95% mean \pm 3*sd : 99.7% - curve is perfectly symmtric - skew close to zero - kurtosis : -2 to +2 - mean==median==mode - Hypothesis testing to prove normality - Shapiro wilk's Test - Anderson-Darling Test -Any distribution can be converted to normal distribution using suitable transformation Standard normal distrubtions - a normal distribution so converted that its new mean =0 sd=1 -this transformation : Z transform $Z_value = x_value - \text{mean}$ ----- std if age follows a normal distrubtion $p(\text{weight} > 141)$ $1 - p(\text{weight} < 141)$ $p(\text{weight} < 141) = 141 - 109$ ----- = 2.46 =0.99266% 13 $p(\text{weight} > 141) = 0.00744$ $p(\text{weight} < 120) = 120 - 109$ ----- = 0.8461 =79.955% 13

Practice problem

[Details](#)

If **birth weights** in a population are normally distributed with a mean of 109 oz and a standard deviation of 13 oz

- What is the chance of obtaining a birth weight of 141 oz *or heavier* when sampling birth records at random?
- What is the chance of obtaining a birth weight of 120 *or lighter*?

$p(\text{weight} > 141)$ $1 - p(\text{weight} < 141)$ $p(\text{weight} < 141) = 141 - 109$ ----- = 2.46 =0.99266% 13 $p(\text{weight} > 141) = 0.00744$ $p(\text{weight} < 120) = 120 - 109$ ----- = 0.8461 =79.955% Central Limit Theorm ----- A data follows any distribution , if we take enough samples E.g. whats the price of i kg of rice in Maxico Sample1 : 5 variants of rice : mean_s1 Sample2 : 10 variants of rice : mean_s2 Sample3 : 1 variants of rice : mean_s3 s50: :mean_s50 ----- mean(all samples) 1. the mean of all samples follow normal distribution 2. the mean of all sample means is equal to population Mean 3. the SD of samples mean is error factor = Standard error Standard error = $SD(\text{population}) / \sqrt{\text{sample_size}}$ z-Test and T-test- -z-test is a test to prove if a sample mean is significantly different from population mean or not - It is applicable if data is normally distributed - sample size is atleast 30 - we know the population mean and std div Student's t test is a test to prove if a sample mean is significantly different from population mean or not - It is applicable if data is not normally distributed- t distribution - sample size is < 30 - we dont know the population mean and std div when to use : - to compare a sample estimation with a population - to compare 2 estimators from a sample E.g. score_test1 vs score_test2 jan_spend vs mar_spend for 200 customers -python -ttest - one sample t test -two sample t test - independ sample t test E.g jan_spend(male) vs jan_spend(female)

Example 1


[Details](#)


Suppose that we have been told that the price of petrol in Melbourne is normally distributed with a mean of 92 cents per litre, and a standard deviation of 3.1 cents/litre. To test whether this price is in fact true, we sample 50 service stations and obtain a mean of 93.6 cents/litre

Solution:

- **Step 1:** State the null and alternative hypotheses
 - $H_0: \mu = 92$
 - $H_a: \mu \neq 92$
- **Step2:** Determine the appropriate test statistic and it's distribution

Because we know the population standard deviation, we can use the z distribution
- **Step3:** Specify the significance level, Say $\alpha = 0.05$





Hypothesis Testing

- Step1 : Define the null and alt hypothesis Null/H)/Ho : mostly checks equality Alt/H1/Ha : mostly checks for different/ inequality check for > check for < -Step2 : see which stats method can we apply? Z-Test T test F test X2 test
- step3 : what is the test statistics and p value? and level of confidence Z-Test T test F test X2 test
- Step4 : what to do with the Null Hypothesis based on the above values??

- do we reject null hypothesis
- Do we fail to reject the null hypothesis (accept)

If p is high we fail to reject the null
If p is low we reject the null

-step 5 : Conclusion

Ans: 1 H0: pop mean is exactly 92($\mu=92$) Ha: pop mean is not equal to 92, its different $\mu \neq 92$

2 - Normally dis ?? Yes

- Do we have pop info?? yes
- is the same size > 30? yes
- So which test? Z test

3. Test Statics? Z Score = sample mean-pop_mean

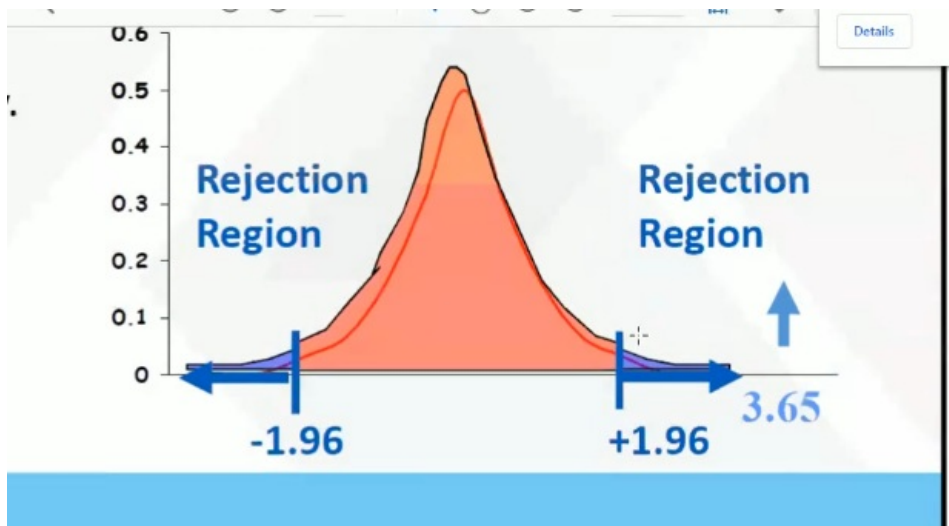
SD/\sqrt{n}

$$\frac{93.6-92}{3.1/\sqrt{50}} = 3.65$$

for the decided CI of 95% the upper and lower cutoff

UC=mean+1.96*SE

LC=mean-1.96*SE



conclusion : we reject the Null Hypothesis

The petrol price in melbourn is diff from 92

```
In [1]: import numpy as np
import pandas as pd
import seaborn as sns
from matplotlib import pyplot as plt
```

A module that was compiled using NumPy 1.x cannot be run in NumPy 2.0.2 as it may crash. To support both 1.x and 2.x versions of NumPy, modules must be compiled with NumPy 2.0. Some module may need to rebuild instead e.g. with 'pybind11>=2.12'.

If you are a user of the module, the easiest solution will be to downgrade to 'numpy<2' or try to upgrade the affected module. We expect that some modules will need time to support NumPy 2.

```
Traceback (most recent call last): File "C:\Users\sival\anaconda3\lib\runpy.py", line 197, in _run_module_as_main
    return _run_code(code, main_globals, None,
File "C:\Users\sival\anaconda3\lib\runpy.py", line 87, in _run_code
    exec(code, run_globals)
File "C:\Users\sival\anaconda3\lib\site-packages\ipykernel_launcher.py", line 16, in <module>
    app.launch_new_instance()
File "C:\Users\sival\anaconda3\lib\site-packages\traitlets\config\application.py", line 846, in launch_instance
    app.start()
File "C:\Users\sival\anaconda3\lib\site-packages\ipykernel\kernelapp.py", line 677, in start
    self.io_loop.start()
File "C:\Users\sival\anaconda3\lib\site-packages\tornado\platform\asyncio.py", line 199, in start
    self.asyncio_loop.run_forever()
File "C:\Users\sival\anaconda3\lib\asyncio\base_events.py", line 601, in run_forever
    self._run_once()
File "C:\Users\sival\anaconda3\lib\asyncio\base_events.py", line 1905, in _run_once
    handle._run()
File "C:\Users\sival\anaconda3\lib\asyncio\events.py", line 80, in _run
    self._context.run(self._callback, *self._args)
File "C:\Users\sival\anaconda3\lib\site-packages\ipykernel\kernelbase.py", line 471, in dispatch_queue
    await self.process_one()
File "C:\Users\sival\anaconda3\lib\site-packages\ipykernel\kernelbase.py", line 460, in process_one
    await dispatch(*args)
File "C:\Users\sival\anaconda3\lib\site-packages\ipykernel\kernelbase.py", line 367, in dispatch_shell
    await result
File "C:\Users\sival\anaconda3\lib\site-packages\ipykernel\kernelbase.py", line 662, in execute_request
    reply_content = await reply_content
File "C:\Users\sival\anaconda3\lib\site-packages\ipykernel\ipkernel.py", line 360, in do_execute
    res = shell.run_cell(code, store_history=store_history, silent=silent)
File "C:\Users\sival\anaconda3\lib\site-packages\ipykernel\zmqshell.py", line 532, in run_cell
    return super().run_cell(*args, **kwargs)
File "C:\Users\sival\anaconda3\lib\site-packages\IPython\core\interactiveshell.py", line 2863, in run_cell
    result = self._run_cell(
File "C:\Users\sival\anaconda3\lib\site-packages\IPython\core\interactiveshell.py", line 2909, in _run_cell
    return runner(coro)
File "C:\Users\sival\anaconda3\lib\site-packages\IPython\core\async_helpers.py", line 129, in _pseudo_sync_runner
    coro.send(None)
File "C:\Users\sival\anaconda3\lib\site-packages\IPython\core\interactiveshell.py", line 3106, in run_cell_async
    has_raised = await self.run_ast_nodes(code_ast.body, cell_name,
File "C:\Users\sival\anaconda3\lib\site-packages\IPython\core\interactiveshell.py", line 3309, in run_ast_nodes
    if await self.run_code(code, result, async_=asy):
File "C:\Users\sival\anaconda3\lib\site-packages\IPython\core\interactiveshell.py", line 3369, in run_code
    exec(code_obj, self.user_global_ns, self.user_ns)
File "C:\Users\sival\AppData\Local\Temp\ipykernel_30892\2193719592.py", line 2, in <cell line: 2>
    import pandas as pd
File "C:\Users\sival\anaconda3\lib\site-packages\pandas\__init__.py", line 80, in <module>
    from pandas.core.api import (
File "C:\Users\sival\anaconda3\lib\site-packages\pandas\core\api.py", line 28, in <module>
    from pandas.core.arrays import Categorical
File "C:\Users\sival\anaconda3\lib\site-packages\pandas\core\arrays\__init__.py", line 1, in <module>
    from pandas.core.arrays.arrow import ArrowExtensionArray
File "C:\Users\sival\anaconda3\lib\site-packages\pandas\core\arrays\arrow\__init__.py", line 5, in <module>
    from pandas.core.arrays.arrow.array import ArrowExtensionArray
File "C:\Users\sival\anaconda3\lib\site-packages\pandas\core\arrays\arrow\array.py", line 50, in <module>
    from pandas.core import (
File "C:\Users\sival\anaconda3\lib\site-packages\pandas\core\ops\__init__.py", line 8, in <module>
    from pandas.core.ops.array_ops import (
File "C:\Users\sival\anaconda3\lib\site-packages\pandas\core\ops\array_ops.py", line 56, in <module>
    from pandas.core.computation import expressions
File "C:\Users\sival\anaconda3\lib\site-packages\pandas\core\computation\expressions.py", line 21, in <module>
    from pandas.core.computation.check import NUMEXPR_INSTALLED
File "C:\Users\sival\anaconda3\lib\site-packages\pandas\core\computation\check.py", line 5, in <module>
    ne = import_optional_dependency("numexpr", errors="warn")
File "C:\Users\sival\anaconda3\lib\site-packages\pandas\compat\_optional.py", line 135, in import_optional_dependency
    module = importlib.import_module(name)
File "C:\Users\sival\anaconda3\lib\importlib\__init__.py", line 127, in import_module
    return _bootstrap.gcd_import(name[level:], package, level)
File "C:\Users\sival\anaconda3\lib\site-packages\numexpr\__init__.py", line 24, in <module>
    from numexpr.interpreter import MAX_THREADS, use_vml, _BLOCK_SIZE1
```

AttributeError Traceback (most recent call last)
AttributeError: _ARRAY_API not found

A module that was compiled using NumPy 1.x cannot be run in NumPy 2.0.2 as it may crash. To support both 1.x and 2.x versions of NumPy, modules must be compiled with NumPy 2.0. Some module may need to rebuild instead e.g. with 'pybind11>=2.12'.

If you are a user of the module, the easiest solution will be to downgrade to 'numpy<2' or try to upgrade the affected module. We expect that some modules will need time to support NumPy 2.

```
Traceback (most recent call last): File "C:\Users\sival\anaconda3\lib\runpy.py", line 197, in _run_module_as_main
    return _run_code(code, main_globals, None,
File "C:\Users\sival\anaconda3\lib\runpy.py", line 87, in _run_code
    exec(code, run_globals)
File "C:\Users\sival\anaconda3\lib\site-packages\ipykernel_launcher.py", line 16, in <module>
    app.launch_new_instance()
File "C:\Users\sival\anaconda3\lib\site-packages\traitlets\config\application.py", line 846, in launch_instance
    app.start()
File "C:\Users\sival\anaconda3\lib\site-packages\ipykernel\kernelapp.py", line 677, in start
    self.io_loop.start()
File "C:\Users\sival\anaconda3\lib\site-packages\tornado\platform\asyncio.py", line 199, in start
    self.asyncio_loop.run_forever()
File "C:\Users\sival\anaconda3\lib\asyncio\base_events.py", line 601, in run_forever
    self._run_once()
File "C:\Users\sival\anaconda3\lib\asyncio\base_events.py", line 1905, in _run_once
    handle._run()
File "C:\Users\sival\anaconda3\lib\asyncio\events.py", line 80, in _run
    self._context.run(self._callback, *self._args)
File "C:\Users\sival\anaconda3\lib\site-packages\ipykernel\kernelbase.py", line 471, in dispatch_queue
    await self.process_one()
File "C:\Users\sival\anaconda3\lib\site-packages\ipykernel\kernelbase.py", line 460, in process_one
    await dispatch(*args)
File "C:\Users\sival\anaconda3\lib\site-packages\ipykernel\kernelbase.py", line 367, in dispatch_shell
    await result
File "C:\Users\sival\anaconda3\lib\site-packages\ipykernel\kernelbase.py", line 662, in execute_request
    reply_content = await reply_content
File "C:\Users\sival\anaconda3\lib\site-packages\ipykernel\ipkernel.py", line 360, in do_execute
    res = shell.run_cell(code, store_history=store_history, silent=silent)
File "C:\Users\sival\anaconda3\lib\site-packages\ipykernel\zmqshell.py", line 532, in run_cell
    return super().run_cell(*args, **kwargs)
File "C:\Users\sival\anaconda3\lib\site-packages\IPython\core\interactiveshell.py", line 2863, in run_cell
    result = self._run_cell()
File "C:\Users\sival\anaconda3\lib\site-packages\IPython\core\interactiveshell.py", line 2909, in _run_cell
    return runner(coro)
File "C:\Users\sival\anaconda3\lib\site-packages\IPython\core\async_helpers.py", line 129, in _pseudo_sync_runner
    coro.send(None)
File "C:\Users\sival\anaconda3\lib\site-packages\IPython\core\interactiveshell.py", line 3106, in run_cell_async
    has_raised = await self.run_ast_nodes(code_ast.body, cell_name,
File "C:\Users\sival\anaconda3\lib\site-packages\IPython\core\interactiveshell.py", line 3309, in run_ast_nodes
    if await self.run_code(code, result, async_=asy):
File "C:\Users\sival\anaconda3\lib\site-packages\IPython\core\interactiveshell.py", line 3369, in run_code
    exec(code_obj, self.user_global_ns, self.user_ns)
File "C:\Users\sival\AppData\Local\Temp\ipykernel_30892\2193719592.py", line 2, in <cell line: 2>
    import pandas as pd
File "C:\Users\sival\anaconda3\lib\site-packages\pandas\__init__.py", line 80, in <module>
    from pandas.core.api import (
File "C:\Users\sival\anaconda3\lib\site-packages\pandas\core\api.py", line 28, in <module>
    from pandas.core.arrays import Categorical
File "C:\Users\sival\anaconda3\lib\site-packages\pandas\core\arrays\__init__.py", line 1, in <module>
    from pandas.core.arrays.arrow import ArrowExtensionArray
File "C:\Users\sival\anaconda3\lib\site-packages\pandas\core\arrays\arrow\__init__.py", line 5, in <module>
    from pandas.core.arrays.arrow.array import ArrowExtensionArray
File "C:\Users\sival\anaconda3\lib\site-packages\pandas\core\arrays\arrow\array.py", line 64, in <module>
    from pandas.core.arrays.masked import BaseMaskedArray
File "C:\Users\sival\anaconda3\lib\site-packages\pandas\core\arrays\masked.py", line 60, in <module>
    from pandas.core import (
File "C:\Users\sival\anaconda3\lib\site-packages\pandas\core\nanops.py", line 52, in <module>
    bn = import_optional_dependency("bottleneck", errors="warn")
File "C:\Users\sival\anaconda3\lib\site-packages\pandas\compat\_optional.py", line 135, in import_optional_dependency
    module = importlib.import_module(name)
File "C:\Users\sival\anaconda3\lib\importlib\__init__.py", line 127, in import_module
    return _bootstrap.gcd_import(name[level:], package, level)
File "C:\Users\sival\anaconda3\lib\site-packages\bottleneck\__init__.py", line 7, in <module>
    from .move import (move_argmax, move_argmin, move_max, move_mean, move_median,
```

```
-----
AttributeError                                Traceback (most recent call last)
AttributeError: _ARRAY_API not found
```

```
In [2]: data=pd.Series(np.random.normal(loc=110 ,scale = 10, size=10000))
# loc means central tendency
# scale means std deviation
```

```
In [3]: data
```

```
Out[3]: 0      106.815679
        1      112.765677
        2      102.912704
        3      122.161152
        4      113.942089
        ...
        9995    100.888295
        9996    117.934763
        9997    106.499901
        9998    109.883819
        9999    107.993178
Length: 10000, dtype: float64
```

```
In [24]: sns.distplot(data)
plt.show()
```

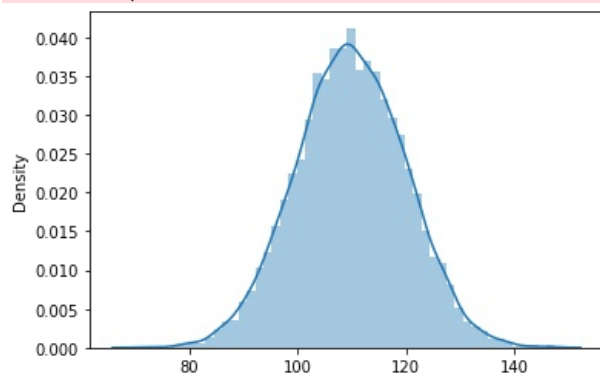
C:\Users\sival\AppData\Local\Temp\ipykernel_30892\2883516215.py:1: UserWarning:

`'distplot'` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `'displot'` (a figure-level function with similar flexibility) or `'histplot'` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see <https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751>

```
sns.distplot(data)
```



```
In [10]: data.mean()
```

```
Out[10]: np.float64(109.81953872823456)
```

```
In [11]: data.std()
```

```
Out[11]: np.float64(10.098677297465043)
```

```
In [12]: LC=data.mean()-data.std()
        UC=data.mean()+data.std()
```

```
In [13]: len(data[(data>LC)&(data<UC)]) / len(data) * 100
```

```
Out[13]: 68.05
```

```
In [15]: LC=data.mean()-2*data.std()
        UC=data.mean()+2*data.std()
        len(data[(data>LC)&(data<UC)]) / len(data) * 100
```

```
Out[15]: 95.65
```

```
In [16]: LC=data.mean()-3*data.std()
        UC=data.mean()+3*data.std()
        len(data[(data>LC)&(data<UC)]) / len(data) * 100
```

```
Out[16]: 99.72999999999999
```

```
In [18]: data.skew()
```

```
Out[18]: np.float64(-0.009189262514076894)
```

```
In [19]: data.kurtosis()
```

```
Out[19]: np.float64(-0.002579663020549905)
```

```
In [20]: data.median()
```

```
Out[20]: np.float64(109.79850151241288)
```

```
In [23]: #Standard Gaussian/Normal
```

```
z_data = (data-data.mean())/data.std()
```

```
In [27]: sns.distplot(data)
plt.show()
```

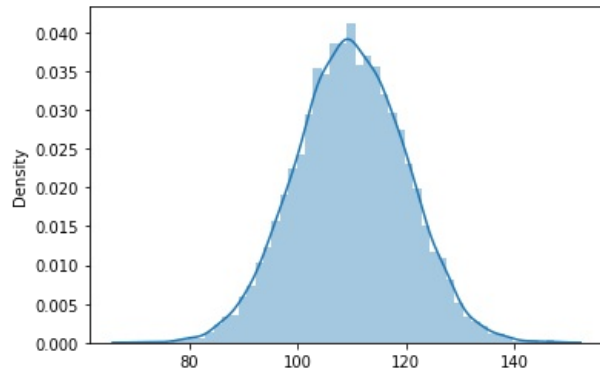
C:\Users\sival\AppData\Local\Temp\ipykernel_30892\275652812.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see <https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751>

```
sns.distplot(data)
```



```
In [26]: sns.distplot(z_data)
plt.show()
```

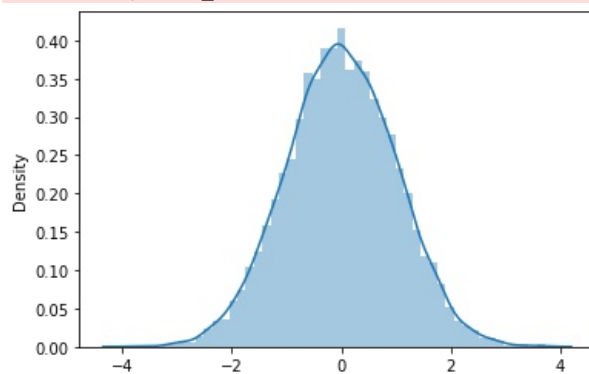
C:\Users\sival\AppData\Local\Temp\ipykernel_30892\2040845632.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see <https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751>

```
sns.distplot(z_data)
```



```
In [28]: z_data.mean()
```

```
Out[28]: np.float64(-2.3110402480597257e-16)
```

```
In [29]: z_data.std()
```

```
Out[29]: np.float64(1.0)
```

```
In [36]: data2=pd.Series(np.random.randint(40,high=100,size=1000, dtype=int))
```

```
In [37]: sns.distplot(data2)
```

C:\Users\sival\AppData\Local\Temp\ipykernel_30892\4141362572.py:1: UserWarning:

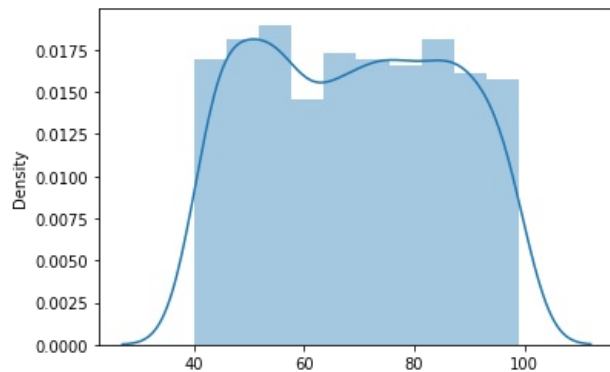
`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see <https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751>

```
sns.distplot(data2)
<Axes: ylabel='Density'>
```

Out[37]:



```
In [38]: s1_mean = data2.sample(n=10).mean()
```

```
In [42]: all_sample_means=[]
for i in range(0,150):
    all_sample_means.append(data2.sample(n=10).mean())
all_sample_means = pd.Series(all_sample_means)
len(all_sample_means)
```

Out[42]: 150

```
In [43]: sns.distplot(all_sample_means)
```

C:\Users\sival\AppData\Local\Temp\ipykernel_30892\2369821419.py:1: UserWarning:

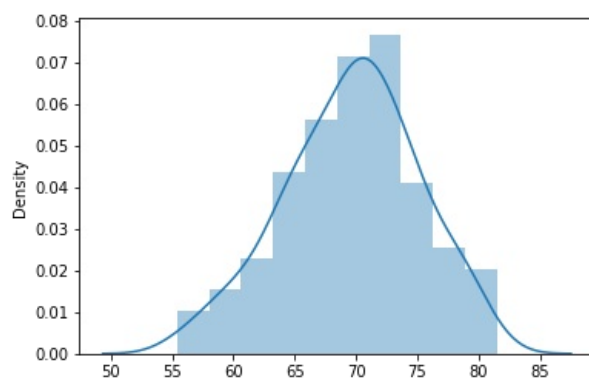
`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see <https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751>

```
sns.distplot(all_sample_means)
<Axes: ylabel='Density'>
```

Out[43]:



```
In [44]: data2.mean()
```

Out[44]: np.float64(69.037)

```
In [45]: all_sample_means.mean()
```

Out[45]: np.float64(69.64266666666666)

```
In [46]: data2.mean()-all_sample_means.mean()
```

Out[46]: np.float64(-0.60566666666666501)

```
In [49]: data2.std()/np.sqrt(150)
```

```
Out[49]: np.float64(1.4102196966878102)
```

```
In [ ]:
```

```
Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js
```