МІНІСТЕРСТВО ОСВІТИ І НАУК УКРАЇНИ

ЗАХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ФАКУЛЬТЕТ КОМП’ЮТЕРНИХ ІНФОРМАЦІНИХ ТЕХНОЛОГІЙ

Лабораторна робота №2

з дисципліни «Засоби аналізу даних на Python»

на тему:

Методи та алгоритми статистичної та попередньої обробки даних

Виконав:

Студент групи ІПЗ-22

Строгий Є. В.

Перевірив:

Тимчишин В. С.

Тернопіль 2024

**Мета роботи**: Надбання практичних навичок із застосування основних методів математичної статистики, реалізованих в бібліотеках NumPy, SciPy для ефективного розв’язання широкого кола задач аналітичного аналізу даних в програмному середовищі Python.

**МОДЕЛЬНІ ПРИКЛАДИ З ЛАБОРАТОРНОЇ РОБОТИ**

'''

ПРИКЛАД 1

'''

**import** numpy **as** np

**from** scipy **import** stats

**import** matplotlib.pyplot **as** plt

**%matplotlib** inline

plt**.**style**.**use('ggplot')

In [56]:

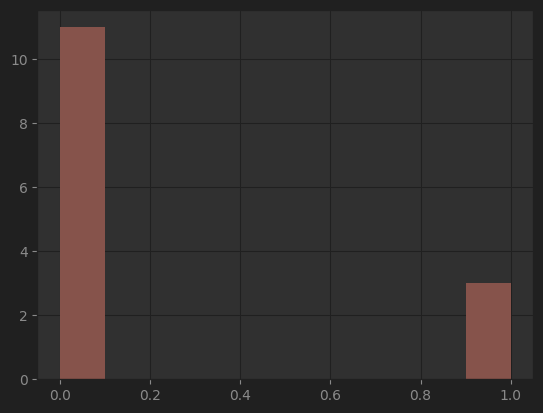
rv\_bernoulli **=** stats**.**bernoulli(p**=**0.3)

In [7]:

rv\_bernoulli**.**rvs(14)

plt**.**hist(rv\_bernoulli**.**rvs(14),bins**=**10)

plt**.**show()



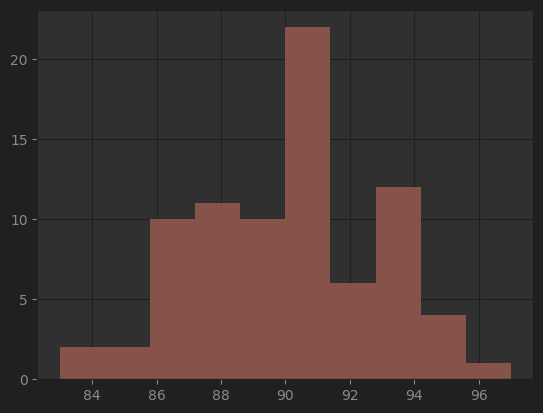
In [8]:

rv\_binom **=** stats**.**binom(100, p**=**0.9)

rv\_binom**.**rvs(8)

plt**.**hist(rv\_binom**.**rvs(80),bins**=**10)

plt**.**show()



In [12]:

a **=** 5

b **=** 10

rv\_uniform **=** stats**.**uniform(a, b **-** a)

rv\_uniform**.**cdf(5.5)

Out[12]:

0.1

In [13]:

rv\_uniform**.**pdf(7)

Out[13]:

0.2

In [16]:

X **=** np**.**linspace(a **-** 2, b **+** 2, 100)

cdf **=** rv\_uniform**.**cdf(X)

plt**.**plot(X, cdf)

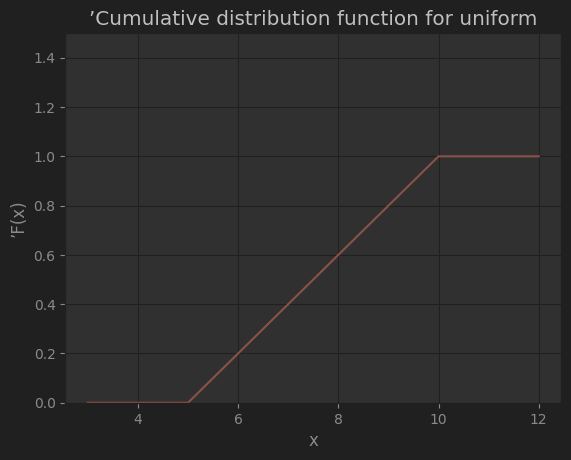
plt**.**ylabel('’F(x)')

plt**.**xlabel('x')

plt**.**ylim([0,1.5])

plt**.**title('’Cumulative distribution function for uniform')

plt**.**show()



In [17]:

X **=** np**.**linspace(a**-**2 , b **+** 2, 1000)

pdf **=** rv\_uniform**.**pdf(X)

plt**.**plot(X, pdf)

plt**.**ylabel('f(x)')

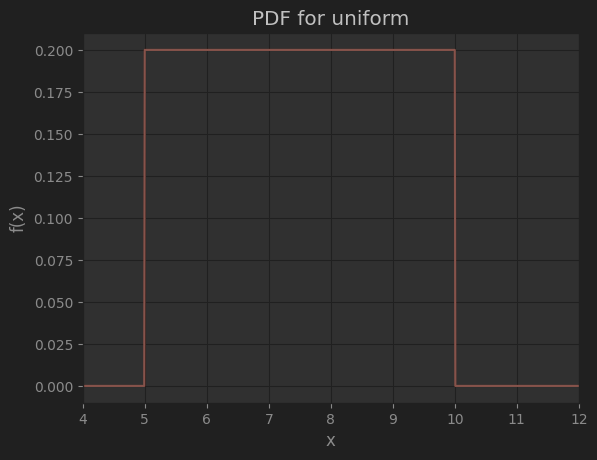
plt**.**xlabel('x')

plt**.**xlim([4, 12])

plt**.**title('PDF for uniform')

Out[17]:

Text(0.5, 1.0, 'PDF for uniform')



In [19]:

mu **=** 2

sigma **=** 0.5

rv\_norm **=** stats**.**norm(loc**=**mu, scale**=**sigma)

rv\_norm**.**rvs(17)

Out[19]:

array([2.16873021, 2.57209665, 2.1182045 , 1.96869305, 1.91169415,

1.67329246, 1.099242 , 1.81440531, 0.83583473, 1.60102193,

0.94222477, 1.3303094 , 2.13034596, 1.95660272, 1.46260657,

2.20370413, 1.63927468])

In [20]:

x **=** np**.**linspace(0, 4, 100)

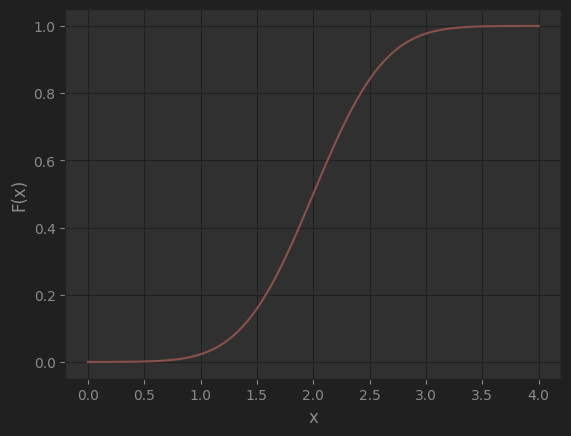
cdf **=** rv\_norm**.**cdf(x)

plt**.**plot(x, cdf)

plt**.**ylabel('F(x)')

plt**.**xlabel('x')

plt**.**show()



In [22]:

x **=** np**.**linspace(0,4,100)

pdf **=** rv\_norm**.**pdf(x)

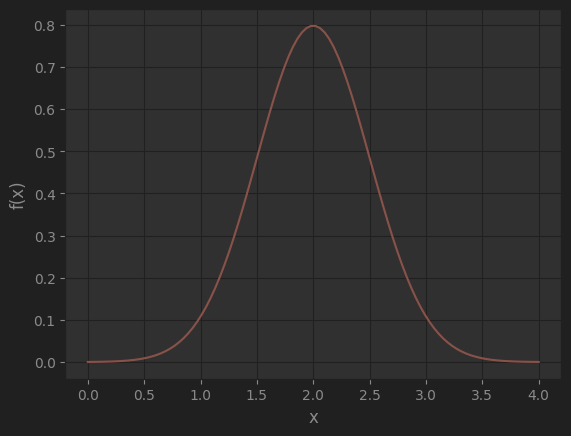
plt**.**plot(x, pdf)

plt**.**ylabel('f(x)')

plt**.**xlabel('x')

Out[22]:

Text(0.5, 0, 'x')



In [23]:

means **=** [1,5,10]

mu **=** 10

sigmas **=** [1, 2, 4, 6]

**for** sigma **in** sigmas:

rv\_norm **=** stats**.**norm(loc**=**mu, scale**=**sigma)

x **=** np**.**linspace(0,20,200)

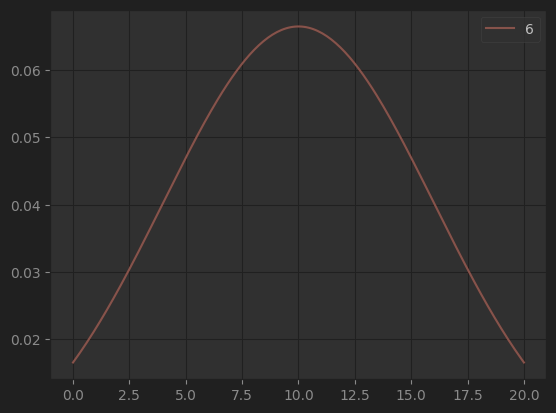
cdf **=** rv\_norm**.**pdf(x)

plt**.**plot(x, cdf, label**=**sigma)

plt**.**legend()

Out[23]:

<matplotlib.legend.Legend at 0x21a1328f460>



In [25]:

mean, var, skew **=** rv\_norm**.**stats(moments**=**'mvs')

print(mean, var, skew)

10.0 36.0 0.0

In [54]:

'''

ПРИКЛАД 2

'''

In [27]:

**from** scipy **import** stats

population **=** stats**.**norm**.**rvs(loc**=**2, scale**=**5, size**=**100000)

In [29]:

**import** numpy **as** np

sample\_size **=** 100

sample **=** np**.**random**.**choice(a **=** population, size **=** sample\_size)

In [30]:

sample\_mean **=** sample**.**mean()

st\_dev **=** population**.**std()

In [33]:

z\_value **=** stats**.**norm**.**ppf(q **=** 0.975)

print('z-value:right', z\_value)

z-value:right 1.959963984540054

In [34]:

z\_value **=** stats**.**norm**.**ppf(q **=** 0.025)

print("z-value:left", z\_value)

z-value:left -1.9599639845400545

In [35]:

interval **=** z\_value **\*** (st\_dev**/**np**.**sqrt(sample\_size))

conf\_inv **=** (sample\_mean **-** interval, sample\_mean **+** interval)

print("Confidence interval:", conf\_inv)

Confidence interval: (3.039251367897685, 1.0777684747517144)

In [36]:

**def** compute\_ci(sample, st\_dev):

z\_value **=** stats**.**norm**.**ppf(q **=** 0.975)

sample\_size **=** len(sample)

interval **=** z\_value **\*** (st\_dev**/**np**.**sqrt(sample\_size))

conf\_inv **=** (sample\_mean **-** interval, sample\_mean **+** interval)

**return** conf\_inv

In [37]:

np**.**random**.**seed(5)

sample\_size **=** 2000

sample **=** np**.**random**.**choice(a **=** population, size **=** sample\_size)

ci **=** compute\_ci(sample, st\_dev)

print("conf interval for 2000 sample size:", ci)

conf interval for 2000 sample size: (1.8392094670358323, 2.2778103756135666)

In [40]:

**def** compute\_ci\_t(sample, alpha**=**0.95):

n **=** sample**.**shape[0]

mu, se **=** np**.**mean(sample), stats**.**sem(sample)

bound **=** se **\*** stats**.**t**.**ppf((1 **+** alpha) **/** 2., n**-**1)

**return** mu **-** bound, mu **+** bound

sample **=** np**.**random**.**choice(a **=** population, size **=** 30)

ci\_t **=** compute\_ci\_t(sample, alpha**=**0.95)

print("conf interval with t test for 2000 sample size:", ci\_t)

conf interval with t test for 2000 sample size: (1.0500565962890003, 3.7883864315619586)

In [50]:

**import** pandas **as** pd

**import** numpy **as** np

taxi\_mex **=** pd**.**read\_csv('taxi-route/mex\_clean.csv')

**def** generate\_distribution\_sample(data, sample\_size, dist\_size):

sample\_means **=** []

**for** i **in** range(dist\_size):

sample **=** np**.**random**.**choice(a **=** data, size **=** sample\_size)

sample\_means**.**append(np**.**mean(sample))

**return** sample\_means

sample\_size **=** 10000

dist\_size **=** 50000

sample\_means **=** generate\_distribution\_sample(taxi\_mex['dist\_meters']**/**1000, sample\_size, dist\_size)

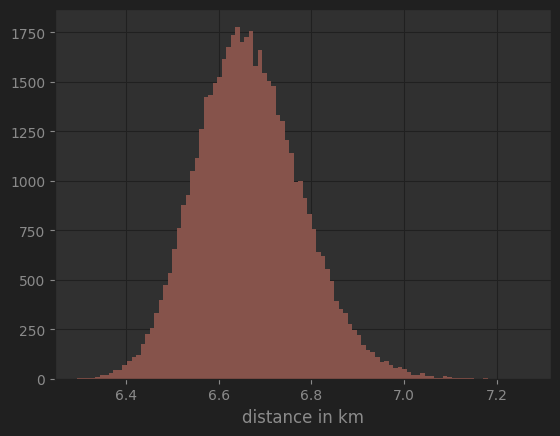
In [51]:

**import** matplotlib.pyplot **as** plt

plt**.**hist(sample\_means, bins**=**100)

plt**.**xlabel('distance in km')

plt**.**show()



In [49]:

np**.**sort(sample\_means)

lb **=** np**.**percentile(sample\_means, 2.5)

ub **=** np**.**percentile(sample\_means, 97.5)

print("conf interval for bootstrap:", (lb, ub))

conf interval for bootstrap: (6.6081902, 6.6081902)

In [53]:

'''

ПРИКЛАД 3

'''

In [58]:

**import** numpy **as** np

**from** scipy **import** stats

**import** matplotlib.pyplot **as** plt

**import** pandas **as** pd

**%matplotlib** inline

taxi\_bog **=** pd**.**read\_csv('taxi-route/bog\_clean.csv')

taxi\_mex **=** pd**.**read\_csv('taxi-route/mex\_clean.csv')

In [59]:

sample **=** taxi\_mex['wait\_sec']**.**sample(n**=**3000)**/**60

print(stats**.**ttest\_1samp(sample, 10))

TtestResult(statistic=1.453391892057237, pvalue=0.14621960531911962, df=2999)

In [64]:

taxi\_mex['pickup\_datetime'] **=** pd**.**to\_datetime(taxi\_mex**.**pickup\_datetime)

taxi\_mex['month'] **=** taxi\_mex['pickup\_datetime']**.**dt**.**month

taxi\_bog**.**shape

taxi\_mex**.**shape

Out[64]:

(12694, 13)

In [66]:

print(stats**.**ttest\_ind(taxi\_mex['trip\_duration']**.**sample(n**=**3000),

taxi\_bog['trip\_duration']**.**sample(n**=**3000)))

Ttest\_indResult(statistic=-6.228380497873815, pvalue=5.032560415892388e-10)

In [67]:

print(stats**.**ttest\_ind(taxi\_mex['wait\_sec']**.**sample(n**=**3000),

taxi\_bog['wait\_sec']**.**sample(n**=**3000)))

Ttest\_indResult(statistic=0.5801760387501473, pvalue=0.5618177041101722)

In [69]:

control**=**taxi\_mex[taxi\_mex**.**month **==** 11]['trip\_duration']**.**sample(n**=**1000)

treatment **=** taxi\_mex[taxi\_mex**.**month **==** 12]['trip\_duration']**.**sample(n**=**1000)

print(stats**.**ttest\_rel(control, treatment))

TtestResult(statistic=-0.8962780500959215, pvalue=0.37032019828439056, df=999)

**ОПРАЦЮВАННЯ САМОСТІЙНО ОБРАНОГО ДАТАСЕТУ**

<https://www.kaggle.com/datasets/sahirmaharajj/fuel-economy>

'''

ПРИКЛАД 1.2

https://www.kaggle.com/datasets/sahirmaharajj/fuel-economy

Цей набір даних пропонує всебічний погляд на економію палива в автомобільному світі, досліджуючи деталі різних транспортних засобів, що охоплюють різні роки, марки та моделі. Він містить інформацію про технічні характеристики двигуна, типи палива, системи трансмісії та навіть такі фактори, як економія палива, викиди CO2 та технологічні особливості.

'''

In [135]:

**import** pandas **as** pd

**import** numpy **as** np

taxi\_mex **=** pd**.**read\_csv('dataset/fuel.csv', low\_memory**=False**)

print(taxi\_mex)

**def** generate\_distribution\_sample(data, sample\_size, dist\_size):

sample\_means **=** []

**for** i **in** range(dist\_size):

sample **=** np**.**random**.**choice(a **=** data, size **=** sample\_size)

sample\_means**.**append(np**.**mean(sample))

**return** sample\_means

sample\_size **=** 10000

dist\_size **=** 38000

sample\_means **=** generate\_distribution\_sample(taxi\_mex['city\_range\_ft1'], sample\_size, dist\_size)

vehicle\_id year make model \

0 26587 1984 Alfa Romeo GT V6 2.5

1 27705 1984 Alfa Romeo GT V6 2.5

2 26561 1984 Alfa Romeo Spider Veloce 2000

3 27681 1984 Alfa Romeo Spider Veloce 2000

4 27550 1984 AM General DJ Po Vehicle 2WD

... ... ... ... ...

38108 37564 2017 Volvo XC60 FWD

38109 37547 2017 Volvo XC90 AWD

38110 37548 2017 Volvo XC90 AWD

38111 37703 2017 Volvo XC90 AWD PHEV

38112 37687 2017 Volvo XC90 FWD

class drive \

0 Minicompact Cars NaN

1 Minicompact Cars NaN

2 Two Seaters NaN

3 Two Seaters NaN

4 Special Purpose Vehicle 2WD 2-Wheel Drive

... ... ...

38108 Small Sport Utility Vehicle 2WD Front-Wheel Drive

38109 Standard Sport Utility Vehicle 4WD All-Wheel Drive

38110 Standard Sport Utility Vehicle 4WD All-Wheel Drive

38111 Standard Sport Utility Vehicle 4WD All-Wheel Drive

38112 Standard Sport Utility Vehicle 2WD Front-Wheel Drive

transmission transmission\_type engine\_index engine\_descriptor \

0 Manual 5-Speed NaN 9001 (FFS)

1 Manual 5-Speed NaN 9005 (FFS) CA model

2 Manual 5-Speed NaN 9002 (FFS)

3 Manual 5-Speed NaN 9006 (FFS) CA model

4 Automatic 3-Speed NaN 1830 (FFS)

... ... ... ... ...

38108 Automatic (S8) NaN 90 SIDI

38109 Automatic (S8) NaN 52 SIDI

38110 Automatic (S8) NaN 53 SIDI

38111 Automatic (S8) NaN 54 SIDI; PHEV

38112 Automatic (S8) NaN 50 SIDI

... hours\_to\_charge\_ac\_240v composite\_city\_mpg composite\_highway\_mpg \

0 ... 0.0 0 0

1 ... 0.0 0 0

2 ... 0.0 0 0

3 ... 0.0 0 0

4 ... 0.0 0 0

... ... ... ... ...

38108 ... 0.0 0 0

38109 ... 0.0 0 0

38110 ... 0.0 0 0

38111 ... 0.0 29 32

38112 ... 0.0 0 0

composite\_combined\_mpg range\_ft1 city\_range\_ft1 highway\_range\_ft1 \

0 0 0 0.0 0.0

1 0 0 0.0 0.0

2 0 0 0.0 0.0

3 0 0 0.0 0.0

4 0 0 0.0 0.0

... ... ... ... ...

38108 0 0 0.0 0.0

38109 0 0 0.0 0.0

38110 0 0 0.0 0.0

38111 30 0 0.0 0.0

38112 0 0 0.0 0.0

range\_ft2 city\_range\_ft2 highway\_range\_ft2

0 NaN 0.00 0.0

1 NaN 0.00 0.0

2 NaN 0.00 0.0

3 NaN 0.00 0.0

4 NaN 0.00 0.0

... ... ... ...

38108 NaN 0.00 0.0

38109 NaN 0.00 0.0

38110 NaN 0.00 0.0

38111 NaN 13.84 13.3

38112 NaN 0.00 0.0

[38113 rows x 81 columns]

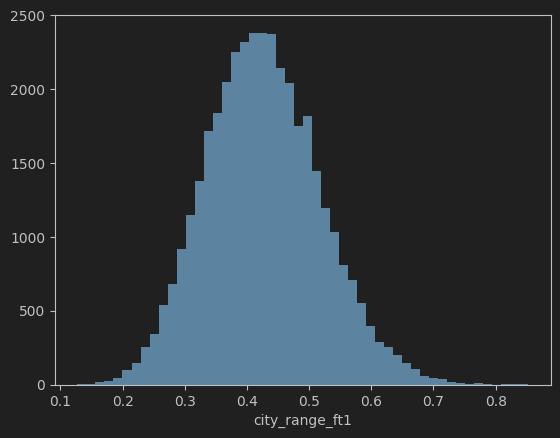
In [136]:

**import** matplotlib.pyplot **as** plt

plt**.**hist(sample\_means, bins**=**50)

plt**.**xlabel('city\_range\_ft1')

plt**.**show()



In [137]:

np**.**sort(sample\_means)

lb **=** np**.**percentile(sample\_means, 2.5)

ub **=** np**.**percentile(sample\_means, 97.5)

print("conf interval for bootstrap:", (lb, ub))

conf interval for bootstrap: (0.25818226474999995, 0.6170705455000001)

In [138]:

'''

ПРИКЛАД 1.3

https://www.kaggle.com/datasets/sahirmaharajj/fuel-economy

У статистиці t-критерій Стьюдента – це набір методів для перевірки гіпотез,

які часто його використовують для перевірки рівності середніх значень вибірки.

Для цього потрібно, щоб було обчислена спочатку t-статистика, після чого її

потрібно порівняти з деяким граничним значенням, що дозволить прийняти рішення

про прийняття або відхилення самої гіпотези.

T-статистика будується за таким принципом:

у чисельнику вказується величина з нульовим математичним очікуванням,

а в знаменнику – стандартне відхилення від цієї випадкової величини.

Алгоритм перевірки гіпотез побудований за таким принципом:

для початку потрібно, щоб була визначена деяка нульова гіпотеза.

У разі порівняння середніх потрібно прийняти припущення,

що середні значення за двома вибірками рівні.

Також визначається альтернативне їй припущення – нерівність середніх значень.

Після цього обчислюється t-статистика і порівнюється з показником p-value

(ймовірність прийняття нульової гіпотези).

P-value – це ймовірність прийняття нульової гіпотези за умови,

що справедлива альтернативна гіпотеза.

Тобто, якщо p-value дуже маленьке значення,

то, швидше за все, не відбуваються взагалі якісь випадкові процеси.

Але якщо ж p-value досить велике, то ймовірність того, що вибірки,

отримані випадковим чином, не мають ніяких зв’язків, дуже значна.

In [139]:

**import** numpy **as** np

**from** scipy **import** stats

**import** matplotlib.pyplot **as** plt

**import** pandas **as** pd

**%matplotlib** inline

set\_1 **=** pd**.**read\_csv('dataset/fuel.csv', low\_memory**=False**)

In [141]:

sample **=** set\_1['city\_range\_ft1']**.**sample(n**=**3000)

print(stats**.**ttest\_1samp(sample, 0.05))

TtestResult(statistic=2.020593215785822, pvalue=0.043410606004308745, df=2999)

In [142]:

set\_1**.**shape

Out[142]:

(38113, 81)

In [143]:

print(stats**.**ttest\_ind(set\_1['city\_range\_ft1']**.**sample(n**=**5000),

set\_1['highway\_range\_ft1']**.**sample(n**=**5000)))

Ttest\_indResult(statistic=1.323803206804343, pvalue=0.1855986770970235)

In [144]:

print(stats**.**ttest\_ind(set\_1['city\_range\_ft2']**.**sample(n**=**10000),

set\_1['highway\_range\_ft2']**.**sample(n**=**10000)))

Ttest\_indResult(statistic=2.1758017147358344, pvalue=0.029581758539972015)

In [145]:

control **=** set\_1['composite\_city\_mpg']**.**sample(n**=**7000)

treatment **=** set\_1['composite\_highway\_mpg']**.**sample(n**=**7000)

print(stats**.**ttest\_rel(control, treatment))

TtestResult(statistic=0.4893474754197052, pvalue=0.6246110502605983, df=6999)