```
from scipy import stats
import seaborn as sns
import  numpy as np
import matplotlib.pyplot as plt
```

Uber Data

```
id = "1NokZy4YzavFdTZlWcIUs47WW5M2A4ElE"
print("https://drive.google.com/uc?export=download&id=" + id)
```

```
    https://drive.google.com/uc?export=download&id=1NokZy4YzavFdTZlWcIUs47WW5M2A4E
```

```
/drive.google.com/uc?export=download&id=1NokZy4YzavFdTZlWcIUs47WW5M2A4ElE" -O Uber_
```

```
    --2022-07-01 13:34:47--  https://drive.google.com/uc?export=download&id=1NokZy
    Resolving drive.google.com (drive.google.com)... 74.125.142.100, 74.125.142.10
    Connecting to drive.google.com (drive.google.com)|74.125.142.100|:443... conne
    HTTP request sent, awaiting response... 303 See Other
    Location: https://doc-0c-ag-docs.googleusercontent.com/docs/securesc/ha0ro937g
    Warning: wildcards not supported in HTTP.
    --2022-07-01 13:34:48--  https://doc-0c-ag-docs.googleusercontent.com/docs/sec
    Resolving doc-0c-ag-docs.googleusercontent.com (doc-0c-ag-docs.googleuserconte
    Connecting to doc-0c-ag-docs.googleusercontent.com (doc-0c-ag-docs.googleuserc
    HTTP request sent, awaiting response... 200 OK
    Length: 18251707 (17M) [application/zip]
    Saving to: ‘Uber_dataset.zip’

    Uber_dataset.zip    100%[===================>]  17.41M  48.6MB/s    in 0.4s

    2022-07-01 13:34:49 (48.6 MB/s) - ‘Uber_dataset.zip’ saved [18251707/18251707]
```

```
!unzip Uber_dataset.zip
```

```
    Archive:  Uber_dataset.zip
      inflating: uber_travel_data.csv
      inflating: __MACOSX/._uber_travel_data.csv
```

```
!ls -lrt
```

```
    total 525784
    -rw-r--r-- 1 root root 520141836 May 12 14:30 uber_travel_data.csv
    drwxr-xr-x 1 root root      4096 Jun 29 13:44 sample_data
    -rw-r--r-- 1 root root  18251707 Jul  1 13:34 Uber_dataset.zip
    drwxr-xr-x 2 root root      4096 Jul  1 13:34 __MACOSX
```

```
import pandas as pd
```

```
df = pd.read_csv("./uber_travel_data.csv")
df.sample(100).head()
```

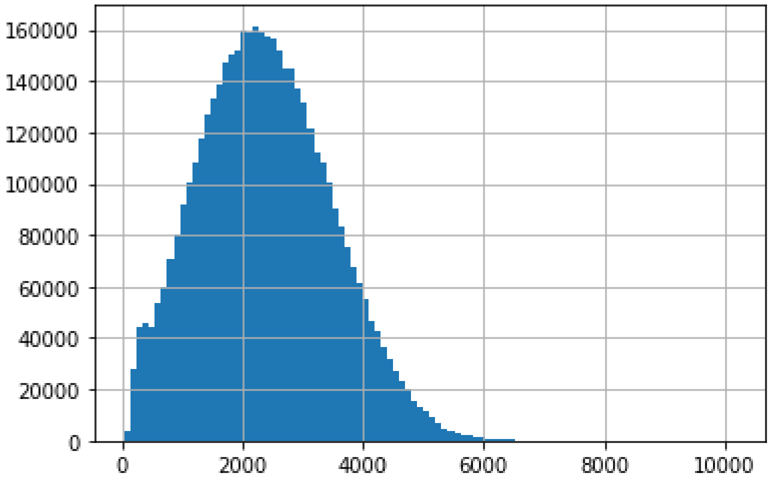
	sourceid	source	dstid	
3699703	234	113, Press Colony, Press Colony, Mayapuri, New...	76	124, SPG Quai
2441504	156	Doctor Satpal Sachdeva Marg, Keshav Puram, Tri...	230	N494, Block N, F
1824456	119	81, Zulfe Bengal, Dilshad Garden, Delhi	58	Pushta Road,
198463	11	Mother Teresa Crescent, Talkatora Garden, Cent...	283	
488666	29	Street Number 14, Block C, Sitapuri Part 1, Ja...	60	

```
df.shape
```

```
(4542026, 5)
```

```
# histogram of travel_times
df["travel_time"].hist(bins = 100)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fde4ea25dd0>
```



```
df.value_counts(['sourceid', 'dstid']).sort_values()
```

```
sourceid  dstid
69        4        50
167       107      50
          101      50
264       14       50
167       100      50
                   ..
83        88       79
244       32       79
202       201      79
```

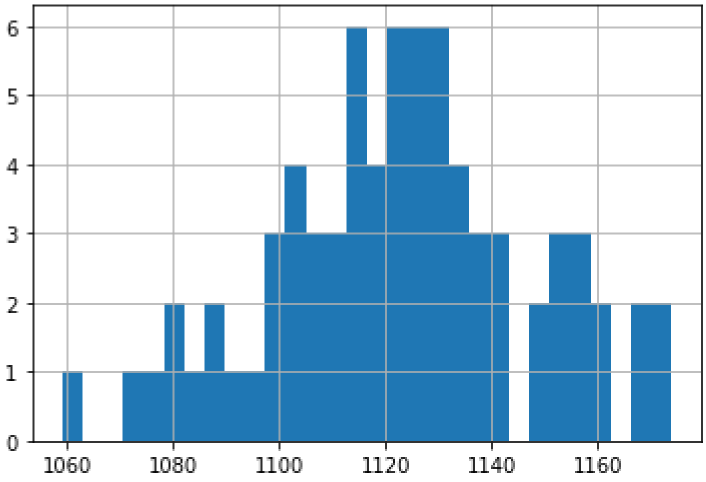
```
                135       79
     45         170       79
     Length: 70429, dtype: int64
```

```
data = df[(df["sourceid"] == 1) & (df["dstid"] == 5)] ["travel_time"]
data.shape
```

```
     (75,)
```

```
data.hist(bins=30)
```

```
     <matplotlib.axes._subplots.AxesSubplot at 0x7fde4e9b9f50>
```

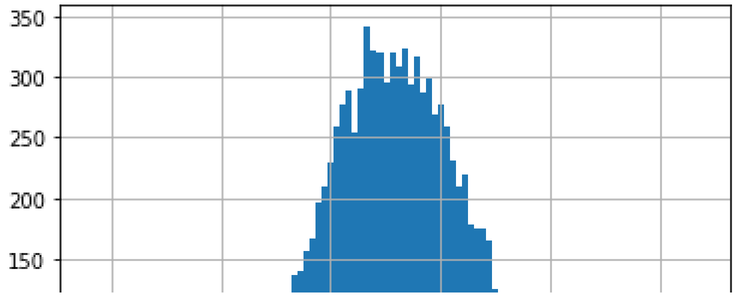


CLT for C.I on mean of travel_time

```
# Let's create r=10000 bootstrap samples, and let each bootstrap sample be of size=
# bs_means is a list of 'r' bootstrap sample means
r = 10000
data = df[(df["sourceid"] == 1) & (df["dstid"] == 5)] ["travel_time"]
size = 50
bs_means = np.empty(r)

for i in range(r):
    bs_sample = np.random.choice(data, size=size)
    bs_means[i] = np.mean(bs_sample)
```

```
import matplotlib.pyplot as plt
plt.figure()
plt.hist(bs_means, bins=100)
plt.grid()
plt.show()
```



```
# QQ-plot with normal distribution
```

```
# compute C.I on the mean given that bs_means follows Gaussian distribution: CLT
print(np.mean(bs_means))
print(np.std(bs_means))
```

```
1122.85326
3.4374772628193493
```

```
print(np.mean(bs_means)-2*np.std(bs_means))
print(np.mean(bs_means)+2*np.std(bs_means))
```

```
1115.9783054743614
1129.7282145256388
```

```
# could we just use the 2.5th percentile and 97.5th percentile value
print(np.percentile(bs_means,2.5))
print(np.percentile(bs_means,97.5))

# what if r is say 100 and not 10,000?
```

```
1116.02
1129.52
```

95% C.I on 99th percentile value for travel_time via bootsrapping

```
# What if we want a C.I on the 99th precentile?
#Let's create r=10000 bootstrap samples, and let each bootstrap sample be of size=1
# bs_99p is a list of 'r' bootstrap sample's 99th percentiles
r = 10000
data = df[(df["sourceid"] == 1) & (df["dstid"] == 5)] ["travel_time"]
size = 75
bs_99p = np.empty(r)

for i in range(r):
    bs_sample = np.random.choice(data, size=size)
    bs_99p[i] = np.percentile(bs_sample,99)
```

```
len(bs_99p)
```

```
    10000
```

```
bs_99p
```

```
    array([1167., 1167., 1174., ..., 1174., 1174., 1174.])
```

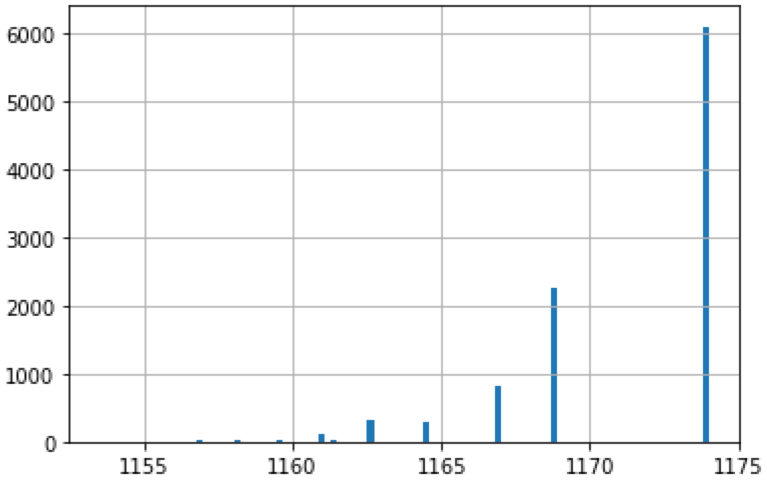
```
#bs_99p may or maynot be normally distributed.
print(np.percentile(bs_99p,2.5))
print(np.percentile(bs_99p,97.5))
```

```
    1162.56
    1174.0
```

```
# Point estimate of the 99th percenitle of the 75 observed samples
print(np.percentile(data,99))
```

```
    1174.0
```

```
# plot the pdf of bs_99p
import matplotlib.pyplot as plt
plt.figure()
plt.hist(bs_99p, bins=100)
plt.grid()
plt.show()
```



CLT as 'n' and 'r' changes

```
data = df[(df["sourceid"] == 1) & (df["dstid"] == 5)] ["travel_time"]
data.shape
```

```
    (75,)
```

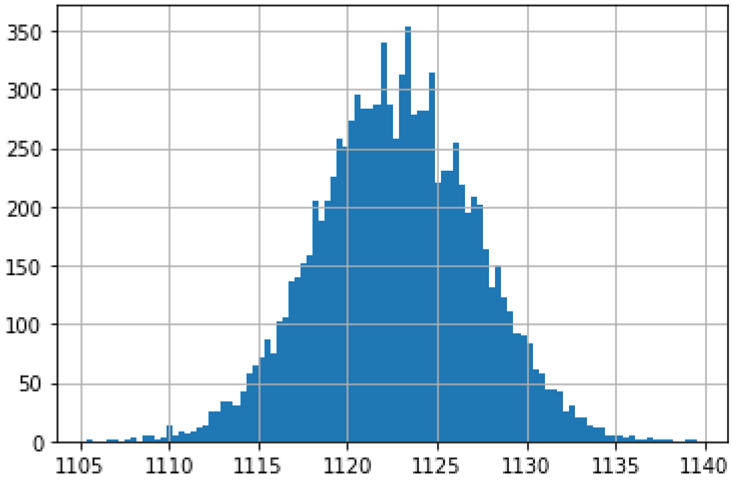
Change "r"

```
# n=30, r=10000
# Let's create r=10000 bootstrap samples, and let each bootstrap sample be of size=
# bs_means is a list of 'r' bootstrap sample means
r = 10000
data = df[(df["sourceid"] == 1) & (df["dstid"] == 5)] ["travel_time"]
size = 30
bs_means = np.empty(r)

for i in range(r):
    bs_sample = np.random.choice(data, size=size)
    bs_means[i] = np.mean(bs_sample)

plt.figure()
plt.hist(bs_means, bins=100)
plt.grid()
plt.show()

print(np.mean(bs_means), np.std(bs_means))
```



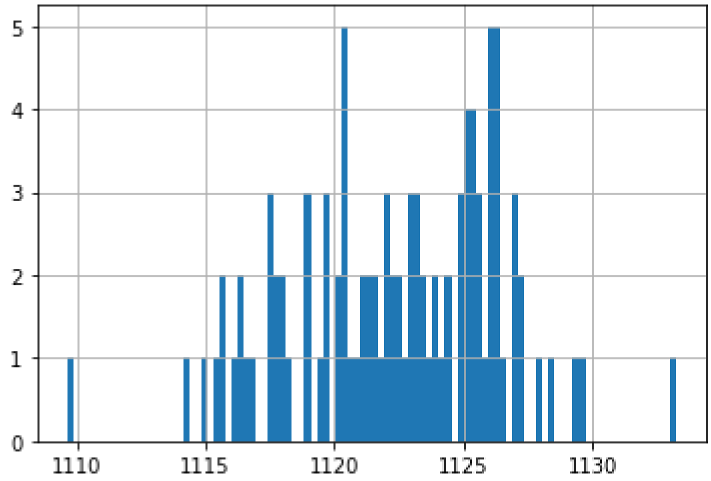
```
1122.7299666666668 4.429857032932577
```

```
# n=30, r=100
# Let's create r=10000 bootstrap samples, and let each bootstrap sample be of size=
# bs_means is a list of 'r' bootstrap sample means
r = 100
data = df[(df["sourceid"] == 1) & (df["dstid"] == 5)] ["travel_time"]
size = 30
bs_means = np.empty(r)

for i in range(r):
    bs_sample = np.random.choice(data, size=size)
    bs_means[i] = np.mean(bs_sample)

plt.figure()
plt.hist(bs_means, bins=100)
plt.grid()
plt.show()

print(np.mean(bs_means), np.std(bs_means))
```



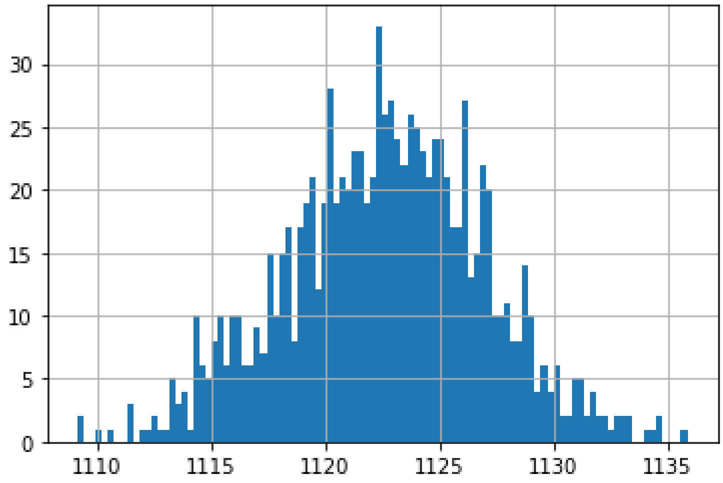
```
1122.3886666666665 3.983455284046753
```

```
# n=30, r=1000
# Let's create r=10000 bootstrap samples, and let each bootstrap sample be of size=
# bs_means is a list of 'r' bootstrap sample means
r = 1000
data = df[(df["sourceid"] == 1) & (df["dstid"] == 5)] ["travel_time"]
size = 30
bs_means = np.empty(r)

for i in range(r):
    bs_sample = np.random.choice(data, size=size)
    bs_means[i] = np.mean(bs_sample)

plt.figure()
plt.hist(bs_means, bins=100)
plt.grid()
plt.show()

print(np.mean(bs_means), np.std(bs_means))
```



```
1122.5838999999999 4.3441068268786065
```

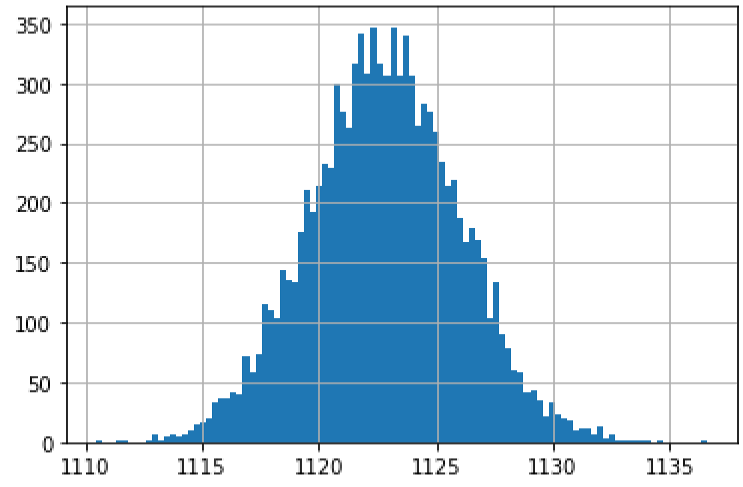
Change "n"

```
# n=60, r=10000
# Let's create r=10000 bootstrap samples, and let each bootstrap sample be of size=
# bs_means is a list of 'r' bootstrap sample means
r = 10000
data = df[(df["sourceid"] == 1) & (df["dstid"] == 5)] ["travel_time"]
size = 60
bs_means = np.empty(r)

for i in range(r):
    bs_sample = np.random.choice(data, size=size)
    bs_means[i] = np.mean(bs_sample)

plt.figure()
plt.hist(bs_means, bins=100)
plt.grid()
plt.show()

print(np.mean(bs_means), np.std(bs_means))
```



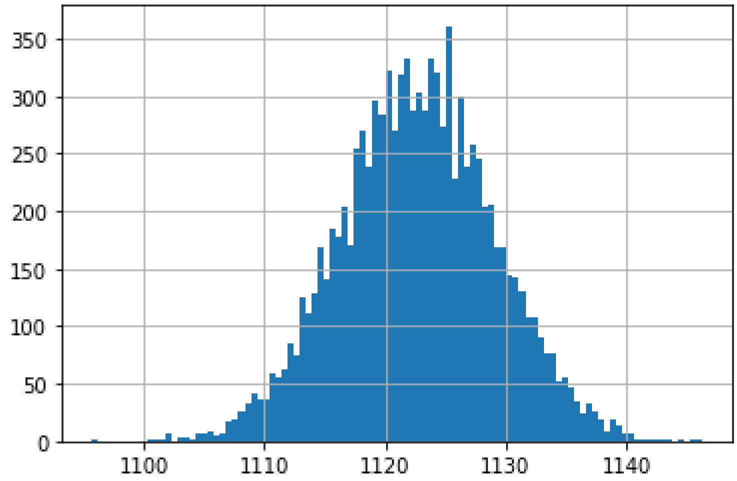
```
1122.8224 3.2077886318563236
```

```
# n=15, r=10000
# Let's create r=10000 bootstrap samples, and let each bootstrap sample be of size=
# bs_means is a list of 'r' bootstrap sample means
r = 10000
data = df[(df["sourceid"] == 1) & (df["dstid"] == 5)] ["travel_time"]
size = 15
bs_means = np.empty(r)

for i in range(r):
    bs_sample = np.random.choice(data, size=size)
    bs_means[i] = np.mean(bs_sample)

plt.figure()
plt.hist(bs_means, bins=100)
plt.grid()
plt.show()

print(np.mean(bs_means), np.std(bs_means))
```



```
1122.7105866666666 6.337015344627504
```