Name : Cheruvathoor Abin Anto

Roll No : TIT2425008

Practical 5

Perform the following to apply Frequentist Approach under Statistical Inference on Data Set ACCIDENTS\_GU\_BCN\_2010 and ACCIDENTS\_GU\_BCN\_2013: Sampling Dsitribution of Point Estimates, Traditional Approach, The Computationally Intensive Approach, Confidence Intervals, Hypothesis Testing - Testing Hypotheses Using Confidence Intervals

**Sampling Distribution of Point Estimates**

import pandas as pd

data2 = pd.read\_csv('D:/Aby/IDS/ACCIDENTS\_GU\_BCN\_2013.csv', encoding='latin-1')

data2['Date'] =data2['NK Any']. apply ( lambda x : str(x))+ '-' + data2['Mes de any']. apply ( lambda x : str(x)) + '-' + data2['Dia de mes']. apply ( lambda x : str(x))

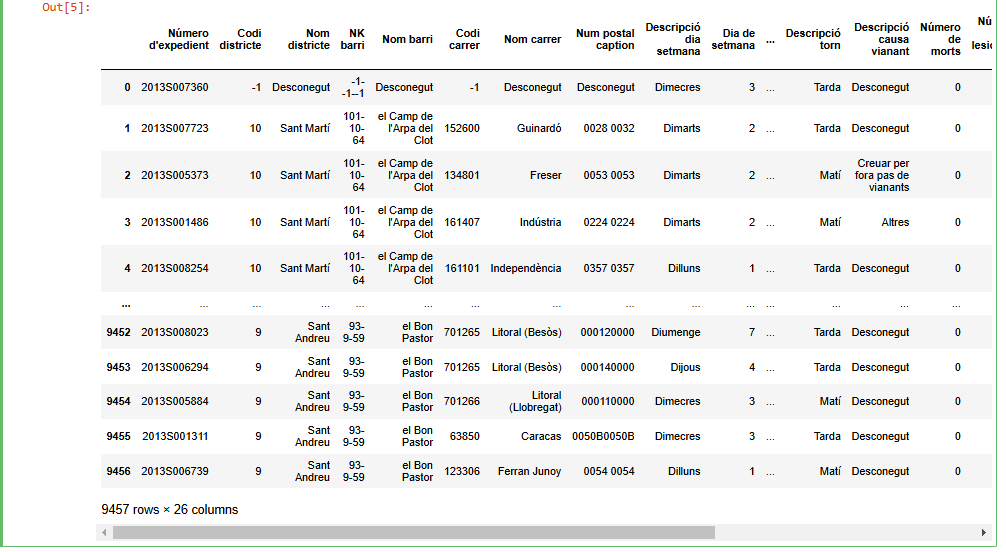
data2['Date']

data2['Date'] = pd.to\_datetime( data2['Date'], format='%Y-%m-%d')

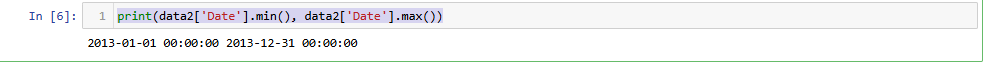
#converts the resulting column to a datetime object.

#NaT (Not a Time) values are added if not able to convert the date to datetime object

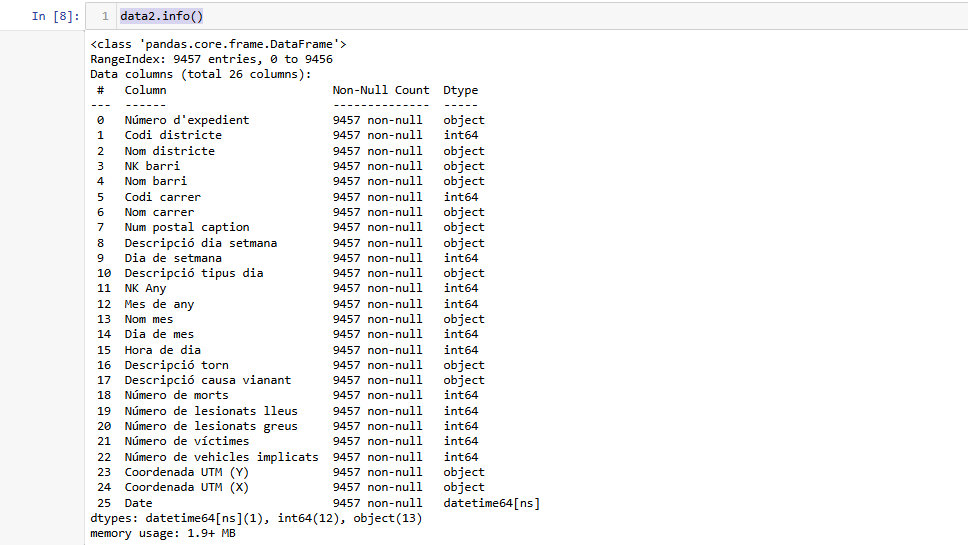
data2



print(data2['Date'].min(), data2['Date'].max())



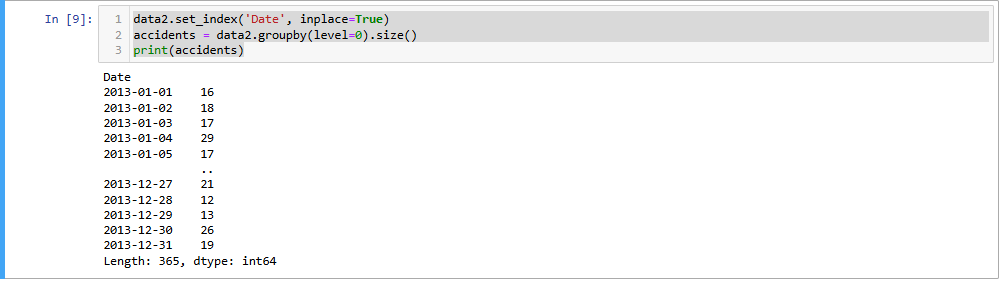
data2.info()



data2.set\_index('Date', inplace=True)

accidents = data2.groupby(level=0).size()

print(accidents)



print( accidents. mean())



import numpy as np

df = accidents.to\_frame() #converting a pandas Series into a DataFrame

N\_test = 10000

elements = 200

# mean array of samples

means = [0] \* N\_test

# sample generation

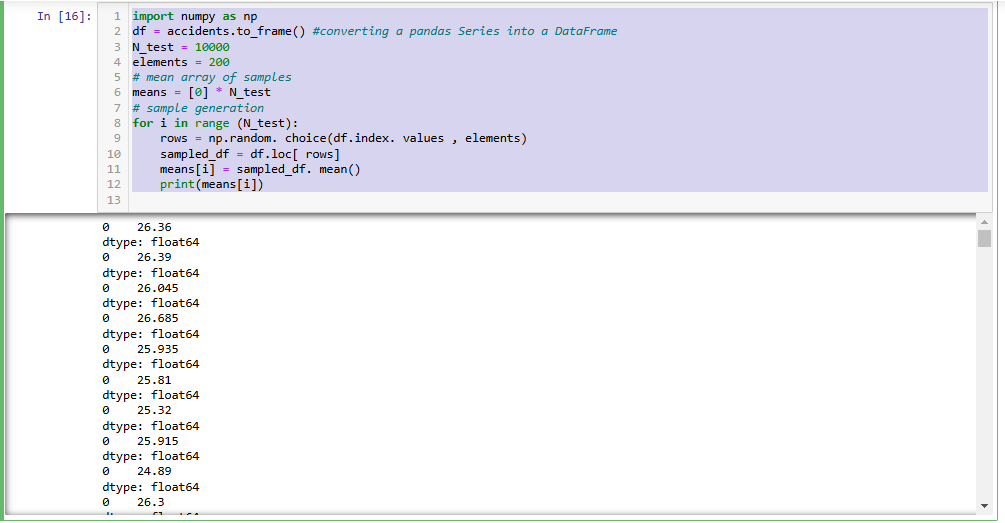
for i in range (N\_test):

rows = np.random. choice(df.index. values , elements)

sampled\_df = df.loc[ rows]

means[i] = sampled\_df. mean()

print(means[i])



**The Traditional Approach**

import math

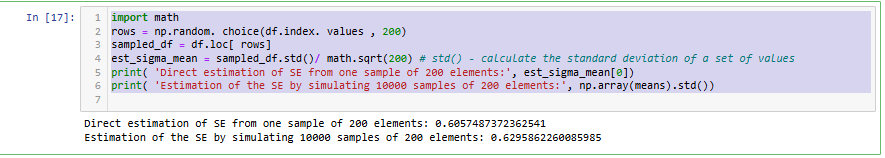
rows = np.random. choice(df.index. values , 200)

sampled\_df = df.loc[ rows]

est\_sigma\_mean = sampled\_df.std()/ math.sqrt(200) # std() - calculate the standard deviation of a set of values

print( 'Direct estimation of SE from one sample of 200 elements:', est\_sigma\_mean[0])

print( 'Estimation of the SE by simulating 10000 samples of 200 elements:', np.array(means).std())

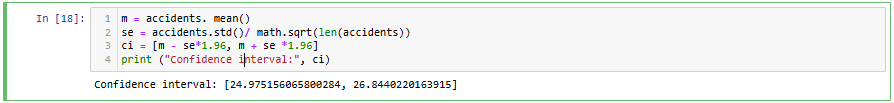


m = accidents. mean()

se = accidents.std()/ math.sqrt(len(accidents))

ci = [m - se\*1.96, m + se \*1.96]

print ("Confidence interval:", ci)



import numpy as np

def meanBootstrap(data, num\_samples):

"""

Performs bootstrapping to estimate the mean of a dataset.

Parameters:

data (array-like): The dataset from which bootstrap samples will be drawn.

num\_samples (int): The number of bootstrap samples to generate.

Returns:

bootstrap\_means (list): List of means of bootstrap samples.

"""

bootstrap\_means = []

n = len(data)

for \_ in range(num\_samples):

# Create a bootstrap sample by sampling with replacement

sample = np.random.choice(data, size=n, replace=True)

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

return bootstrap\_means

import numpy as np

m = meanBootstrap( accidents , 10000)

sample\_mean = np. mean(m)

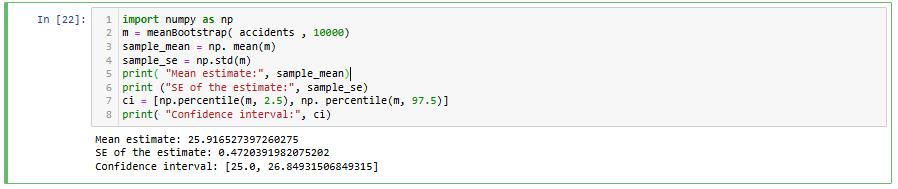
sample\_se = np.std(m)

print( "Mean estimate:", sample\_mean)

print ("SE of the estimate:", sample\_se)

ci = [np.percentile(m, 2.5), np. percentile(m, 97.5)]

print( "Confidence interval:", ci)



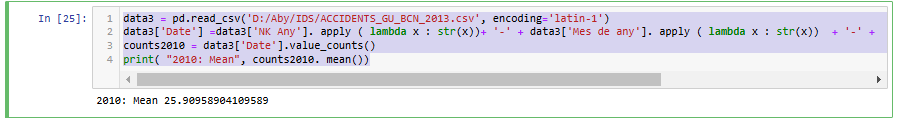
**Testing Hypotheses Using Confidence Intervals**

data3 = pd.read\_csv('D:/Aby/IDS/ACCIDENTS\_GU\_BCN\_2013.csv', encoding='latin-1')

data3['Date'] =data3['NK Any']. apply ( lambda x : str(x))+ '-' + data3['Mes de any']. apply ( lambda x : str(x)) + '-' + data3['Dia de mes']. apply ( lambda x : str(x))

counts2010 = data3['Date'].value\_counts()

print( "2010: Mean", counts2010. mean())

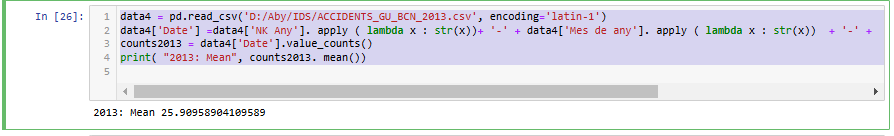


data4 = pd.read\_csv('D:/Aby/IDS/ACCIDENTS\_GU\_BCN\_2013.csv', encoding='latin-1')

data4['Date'] =data4['NK Any']. apply ( lambda x : str(x))+ '-' + data4['Mes de any']. apply ( lambda x : str(x)) + '-' + data4['Dia de mes']. apply ( lambda x : str(x))

counts2013 = data4['Date'].value\_counts()

print( "2013: Mean", counts2013. mean())



n = len(counts2013)

mean = counts2013. mean()

s = counts2013.std()

ci = [mean - s\*1.96/np.sqrt(n), mean + s\*1.96/np.sqrt (n)]

print("2010 accident rate estimate:", counts2010. mean())

print ("2013 accident rate estimate:", counts2013. mean())

print("CI for 2013:",ci)

