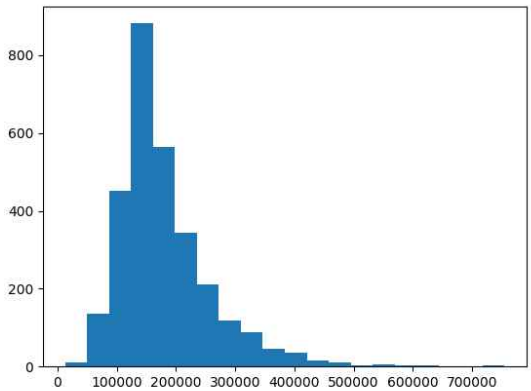
통계학실험(005)
과제 3 (4장)
2022-18608 문현호

1
예제 1. 현재 주어진 자료는 일정 기간동안 지역 내의 모든 부동산 거래를 기록한 자료이므로 일종의 모집단이라고 생각할 수 있다. SalePrice 변수에 대해 히스토그램을 그려보고 수치적 요약값을 구해보자. 모집단의 분포는 어떠한가?

```
ames = pd.read_csv("data/ames.csv")
sale_price = ames.loc[:, "SalePrice"]
df_sp = pd.DataFrame(sale_price)
plt.hist(sale_price, bins=20)
plt.show()
print(sale_price.describe())

'''
count         2930.000000
mean        180796.060068
std          79886.692357
min          12789.000000
25%         129500.000000
50%         160000.000000
75%         213500.000000
max         755000.000000
Name: SalePrice, dtype: float64
```




```
'''
```

- 평균이 180000 정도인, 주로 낮은 값에 쏠려있는 분포를 보인다.

2

예제 2. 이 지역에서 발생한 전체 부동산 거래 가격의 평균값( )을 추정해보려고 한다. 지금처럼 모집단 전체를 알게 되는 경우는 매우 드물기 때문에 대부분의 경우에는 모집단의 부분집합인 표본을 선택하여 모수를 추정하게 된다. SalePrice에서 50개의 랜덤 표본을 선택해보자. 이 때, 모평균의 추정값은 무엇인가?

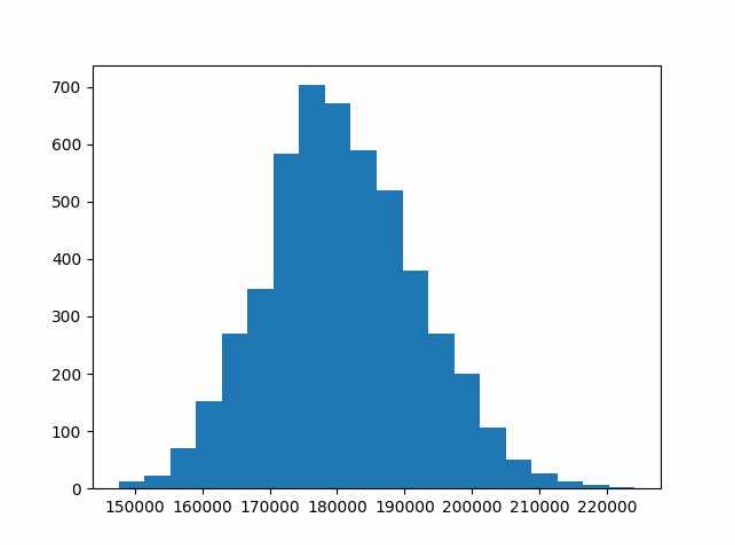
```
sp50 = np.random.choice(sale_price, size=50)
mean50 = np.mean(sp50)
print(mean50)

'''
[156000 161000 109008 149900 210000  60000 148000 139500 277000 119900
 115000 105900 248500 170000 161000 308030 248000 119500 136900 168165
 201000 135000 355000 175000 137000 591587 149000 213000 155000 356383
 183000 105000 111500 255900 136000 148000 112000 186000 194000 144000
 160000 205950 191000 165000 172785 197000  95000 164000 175000 143000]
180468.16

'''
```

- 모평균의 추정값은 무작위로 선출된 표본의 평균, 즉 180468.16이다.
- 실제 모평균과는 약 300 정도의 차이를 보인다.

3
예제 2의 과정을 5000번 반복하여 표본 평균의 표본 분포를 구해보자. 즉, 크기가 50인 랜덤 표본을 선택하여 표본평균을 구하는 과정을 5000번 반복하고 이 결과를 sample_mean50이라는 이름의 벡터에 저장한다. sample_mean50을 이용하여 히스토그램을 sample_mean50 그려보자. 표본 평균의 분포는 어떠한가?

```
sp50s = [np.random.choice(sale_price, size=50) for _ in range(5000)]
mean50s = pd.DataFrame([np.mean(sp50) for sp50 in sp50s])
plt.hist(mean50s, bins=20)
plt.show()
```

'''



'''

- 표본 평균의 분포는 180000 전후로, 어느 정도 정규분포와 유사한 형태를 보인다.

4
예제 3의 sample_mean50의 평균과 분산을 계산해보자. sample_mean50의 평균값은 모집단의 평균과 어떠한 관계가 있는가? sample_mean50의 분산값은 모분산과 어떠한 관계가 있는가?

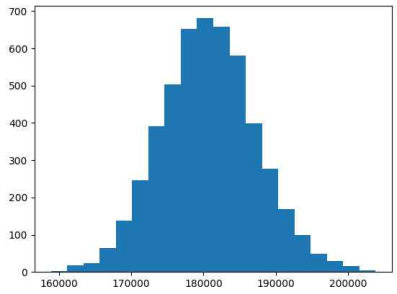
```
print(mean50s.describe())
print("total std : ", df_sp.std())
print("sample mean(50) std : ", mean50s.std())

'''
                          0
count       5000.000000
mean      180780.665844
std        11323.130501
min       147555.280000
25%       173129.820000
50%       180310.510000
75%       188235.105000
max       224174.700000
total std :  SalePrice      79886.692357
dtype: float64
sample mean(50) std :  0       11323.130501
dtype: float64

'''
```

- 표본 평균의 표준편차는 모집단의 표준편차보다 7.0552배 정도 작았다.
- 이 값은 50의 제곱근인 7.0711과 상당히 유사한 수치이다.

5
예제 3의 과정을 표본의 크기를 150으로 증가시켜 반복해보자. 이 결과는 sample_mean150에 저장한다. 표본의 크기에 따른 표본 평균의 분포는 어떠한가?

```
sp150s = [np.random.choice(sale_price, size=150) for _ in range(5000)]
mean150s = pd.DataFrame([np.mean(sp150) for sp150 in sp150s])
plt.hist(mean150s, bins=20)
plt.show()
print(mean150s.describe())
print("total std : ", df_sp.std())
print("sample mean(150) std : ", mean150s.std())
```

'''




```
                0
count    5000.000000
mean   180703.436719
std      6555.407927
min    158851.906667
25%    176239.723333
50%    180590.800000
75%    185022.968333
max    203858.833333
total std :  SalePrice    79886.692357
dtype: float64
sample mean(150) std :  0    6555.407927
dtype: float64
```

'''

- 표본이 커지자, 그 표본 평균의 분포는 평균이 예제 3처럼 180000 정도이지만 예제 3에서보다 더 좁은, 정규분포의 모습을 보였다.
- 이때, 표본 평균의 표준편차는 모집단의 표준편차보다 12.186배 정도 작았다.
- 이 값은 150의 제곱근인 12.247과 상당히 유사한 수치이다.

appendix 1
1) sex 변수에서 0을 Female, 1을 Male로 매핑하여라.
2) 양적자료에 대해서 요약통계량 값을 구하여라.
3) sex 변수에 대해서 빈도를 구하여라.

```
body_dims = pd.read_csv("data/bodydims.csv")
label = {
    0: "Female",
    1: "Male"
}
body_dims["sex"] = body_dims["sex"].map(label)
print(body_dims.describe(include="object"))
print("\n")
bd_sex = body_dims["sex"]
print(bd_sex.value_counts())
```

```
'''
              sex
count         507
unique          2
top        Female
freq          260


Female      260
Male        247
Name: sex, dtype: int64

'''
```

- 매핑, 요약통계량, 빈도는 위와 같다. 여성은 260, 남성은 247명 있었다.

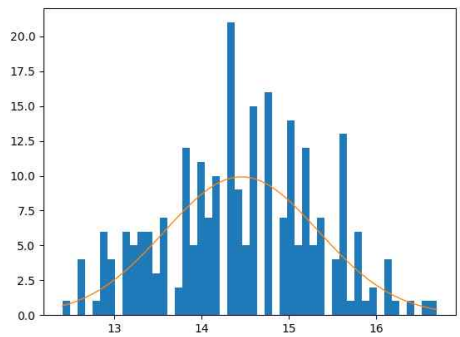
appendix 2
sex 별로 변수 wgt와 hgt의 평균, 최솟값, 최댓값을 출력하여라. 그리고 이에 대한 간단한 해석을 제시하여라.

```
label_show = {
    "wgt": ["mean", "min", "max"],
    "hgt": ["mean", "min", "max"]
}
print(body_dims.groupby("sex").agg(label_show))

'''
              wgt                          hgt
             mean    min    max          mean    min    max
sex
Female   60.600385   42.0   105.2   164.872308   147.2   182.9
Male     78.144534   53.9   116.4   177.745344   157.2   198.1

'''
```

- 체중 평균은 여성이 60.6, 남성이 78.1 정도였고, 최솟값은 여성이 더 낮고 최댓값은 남성이 더 높았다. 최대 최소 모두 평균의 2배를 넘어가지는 않았다.
- 키 평균은 여성이 164.9, 남성이 177.7 정도였다. 최솟값은 여성이 더 낮고 최댓값은 남성이 더 높았다. 최대 최소 모두 평균에서 30, 평균의 20%를 이상 벗어나지는 않았다.
- 전체적으로 남성이 여성보다 체중과 키가 더 커, 체격이 크다는 사실을 확인할 수 있다.

appendix 3
1) 주어진 자료를 성별에 따라 나누어서 각각 bodydims_M과 bodydims_F로 저장하여라
2) 남성의 데이터셋에서 elb.di 변수에 대해서 상대도수 히스토그램을 그리고, 그 히스토그램 위에 eld.di의 평균과 분산을 가지는 정규분포를 그려라.

```
body_dims_M = body_dims[body_dims["sex"] == "Male"]
body_dims_F = body_dims[body_dims["sex"] == "Female"]

bdm_elb_di = body_dims_M["elb.di"]
print(bdm_elb_di.describe())
plt.hist(body_dims_M.loc[:, "elb.di"], bins=50)

y = np.linspace(bdm_elb_di.min(), bdm_elb_di.max(), 50, endpoint=True)
fy_ = norm.pdf(y, loc=[bdm_elb_di.mean()], scale=[bdm_elb_di.std()])
fy = [i*len(bdm_elb_di)/sum(fy_) for i in fy_]
plt.plot(y, fy, linewidth=1.0, linestyle="-")

plt.show()

'''
count    247.000000
mean      14.457085
std        0.882543
min       12.400000
25%       13.800000
50%       14.400000
75%       15.100000
max       16.700000
Name: elb.di, dtype: float64
```




```
'''
```

appendix 4

어떤 타자의 평균타율이 0.383이라고 하자. 이 타자가 시즌동안 96번 타석에 나간다 할 때 이 타자가 안타를 36번이상 70번 미만으로 칠 확률을 구해라. (이 타자에 대해서 안타의 횟수에 대한 분포는 이항분포를 따른다 가정한다.)

1) 정확한 확률을 구하여라.
2) 정규분포로 근사시켜 확률을 구하여라.
3) 연속성 수정을 사용하여 확률을 구하여라.

```
rate = 0.383
n = 96
p = np.zeros((100, 100), dtype="longdouble")
p[0, 0] = 1
for i in range(n):
    for j in range(i+1):
        p[i+1][j] += p[i][j] * (1-rate)
        p[i+1][j+1] += p[i][j] * rate
print("From calculation : ", sum(p[n, 36:70]))

print("From ND approximation : ",
      norm.cdf(69, n*rate, math.sqrt(n*rate*(1-rate))) - norm.cdf(36, n*rate, math.sqrt(n*rate*(1-rate))))

print("From continuity correction : ",
      norm.cdf(69.5, n*rate, math.sqrt(n*rate*(1-rate))) - norm.cdf(35.5, n*rate, math.sqrt(n*rate*(1-rate))))

'''
From calculation :  0.6019265761771341
From ND approximation :  0.5640493966657784
From continuity correction :  0.6049653429802357

'''
```

- 실제 프로그래밍을 통해 구한 확률은 60.19% 정도였다.
- 정규분포 근사를 통해 구한 값은 56.40%, 연속성 수정을 사용해서 구한 값은 60.50%였다. 정규분포 근사보다는 연속성 수정이 충분히 가까운 값을 보였다.