SVEUČILIŠTE U SPLITU

SVEUČILIŠNI ODJEL ZA STRUČNE STUDIJE

Specijalistički diplomski stručni studij Primjenjeno računarstvo

Seminarski rad iz kolegija Statistika

Ilario Batistić

Split, 2024.

Sadržaj

Podaci korišteni za seminar

Podaci za seminar iz statistike mogu se pronaći u *Excel* datoteci "p16.xlsx". Ukupan broj uzoraka korištenih u svrhu testiranja i izrade seminara iznosi 50. Seminar je napisan u Python-u 3.12.1, pri čemu su korištene biblioteke poput *pandas*, *numpy*, *scipy*, *matplotlib*, *statistics* i *math*.

1. Zadatak

1.1 Aritmetička Sredina

Aritmetička sredina, također poznata kao srednja vrijednost (engl. mean ili arithmetic mean), definira se kao suma svih podataka podijeljena s ukupnim brojem podataka.

Formula:

$$X = \frac{\sum_{i=1}^{n} X_i}{n}$$

Slovo n predstavlja veličinu uzorka.

Računanje aritemtičke sredine u *Python-u* se može ovako izvesti:

```
Data. [1500, 2700, 1800, 3200, 3700, 2600, 2800, 1700, 2100, ...]

def arithmetic_mean(data: list):
    return np.mean(data)
```

Rezultat: 2842.0

1.2 Mod

Mod ili dominantna vrijednost uzorka predstavlja vrijednost statističkog obilježja koja se najčešće javlja u uzorku, odnosno ima najveću frekvenciju. Unimodalni nizovi imaju samo jedan mod. Uzorak može imati nula moda, jedan mod, ili može imati dva ili više modova, u kojem slučaju govorimo o bimodalnim nizovima ili multimodalnim nizovima.

Formula:

$$M_o = X_i$$

Računanje mod u *Python-u* se može ovako izvesti:

```
def modulo(data: list):
    return statistics.mode(data)
```

Rezultat: 2900

1.3 Median

Medijan (engl. median) predstavlja središnju vrijednost podataka poredanih po veličini, koja dijeli skup na dva jednako brojna dijela. Ako je broj podataka neparan, medijan je vrijednost središnjeg podatka. Ako je broj podataka paran, medijan predstavlja srednju vrijednost između dva središnja podatka.

Formula:

$$m = x_{(\frac{n+1}{2})}$$

Računanje median u *Python-u* se može ovako izvesti:

```
def median_of_an_sample(data: list):
    return np.median(data)
```

Rezultat: 2900.0

1.4 Karakteristična petorka

Karakterističnu petorku uzorka čini uređena petorka vrijednosti:

- ➔ X_{min} – najmanja vrijednost u uzorku
- ➔ Q_l – donji kvartila uzorka. vrijednost od koje je 25% podataka statističkog niza manje ili jednako, a 75% veće ili jednako.
- ➔ m – medijana
- ➔ Q_u – gornji kvartil uzorka. vrijednost od koje je 75% podataka statističkog niza manje ili jednako, a 25% podataka veće ili jednako.
- ➔ X_{max} – najveća vrijednost u uzorku

Računanje petorke u *Python-u* se može ovako izvesti:

```
def characteristic_five_of_an_sample(data: list):
    return {
        "Median": median_of_an_sample(data),
        "1st Quartile": np.percentile(data, 25),
        "3rd Quartile": np.percentile(data, 75),
        "Minimum": min(data),
        "Maximum": max(data)
    }
```

Rezultat:

- ➔ Median: 2900.0
- ➔ Q_l: 2425.0
- ➔ Q_u: 3275.0
- ➔ X_{min}: 1500
- ➔ X_{max}: 4000

1.5 Varijanca

Varijanca (s^2) se definira kao prosječno kvadratno odstupanje vrijednosti numeričkog obilježja od aritmetičke sredine uzorka.

Varijanca uzorka negrupiranog niza:

$$S_0^2 = \frac{\sum_{i=0}^{n} \left(X_i - \overline{X} \right)^2}{n}$$

Varijanca uzorka grupiranog niza:

$$S_0^2 = \frac{\sum_{i=0}^{n} f_i \left(X_i - \overline{X} \right)^2}{n}$$

Slovo n predstavlja ukupan broj podataka u uzorku, a $\overline{x}$ vrijednost aritmetičke sredine uzorka.

Računanje petorke u *Python-u* se može ovako izvesti:

```
def variance_of_the_data(data: list):
    return np.var(data)
```

Rezultat: 385636.0

1.6 Standardna devijacija

Standardna devijacija je statistička mjera raspršenosti ili varijabilnosti podataka u skupu. Ona pokazuje koliko su pojedinačne vrijednosti podataka udaljene od aritmetičke sredine skupa podataka.

Formula:

$$S_0 = \sqrt{S_0^2}$$

Računanje standarde devijacije u *Python-u* se može ovako izvesti:

```
def standard_deviation_of_data(data: list):
    return np.std(data)
```

Rezultat: 620.9959742220557

1.7 Interkvatil

Interkvartilni je statistička mjera raspršenosti koja predstavlja raspon između gornjeg i donjeg kvartila u skupu podataka.

Formula donjeg kvartila:

$$Q_l = X\left(\frac{n+1}{4}\right)$$

Formula gornjeg kvartila:

$$Q_u = X\left(\frac{3*(n+1)}{4}\right)$$

Formula interkvatila:

$$I_q = Q_u - Q_l$$

Računanje interkvatila u *Python-u* se može ovako izvesti:

```
def interquartile_range_of_data(data: list):
    return np.percentile(data, 75) - np.percentile(data, 25)
```

Rezultat: 850.0

1.8 Raspon uzorka

Raspon uzorka je jednostavna mjera varijabilnosti koja predstavlja razliku između najveće i najmanje vrijednosti u skupu podataka.

Formula:

$$R = X_{max} - X_{min}$$

Računanje raspon uzorka u *Python-u* se može ovako izvesti:

```
def range_of_data(data: list):
    return np.max(np.array(data)) - np.min(np.array(data))
```

Rezultat: 2500

2. Zadatak

2.1 Razdioba frekvencija

Svakom elementu $xi \in R(X)$ možemo pridružiti broj f_i koji predstavlja učestalost pojavljivanja elementa xi u nizu od n podataka. Broj f_i, i = 1,...,k nazivamo **frekvencijom elementa** x_i i vrijedi:

$$\sum_{i=1}^{k} f_i = n$$

Skup uređenih parova (x_i, f_i) naziva se **razdiobom frekvencija obilježja** X.

Računanje razdioba frekvencije u *Python-u* se može ovako izvesti:

```
def create_frequency_distribution(data, class_width=1000):
    bins = np.arange(np.min(data), np.max(data) + class_width, class_width)
    frequencies, bin_centers = np.histogram(data, bins=bins)
    return pd.DataFrame({
        'Interval': [(bin_centers[i], bin_centers[i+1]) for i in range(len(bin_centers)-1)],
        'Frequency': frequencies,
        'Class Width': class_width
    })
```

Rezultat:

Index	Interval	Frekvencija	Širina
0	(1500.0, 1812.5)	4	312.5
1	(1812.5, 2125.0)	4	312.5
2	(2125.0, 2437.5)	5	312.5
3	(2437.5, 2750.0)	6	312.5
4	(2750.0, 3062.5)	13	312.5
5	(3062.5, 3375.0)	6	312.5
6	(3375.0, 3687.5)	6	312.5
7	(3687.5, 4000.0)	6	312.5

2.2 Razdioba relativnih frekvencija

Broj $\frac{f_i}{n}$, i = 1,....,k zovemo **relativnom frekvencijom** elementa x_i. Skup uređenih parova (x_i, $\frac{f_i}{n}$), i = 1,....,k zovemo **razdioba relativnih frekvencija** obilježja X.

Računanje relativnih frekvencije u *Python-u* se može ovako izvesti:

```
def create_relative_frequency_distribution(frequency_distribution):
    total = frequency_distribution['Frequency'].sum()
    return pd.DataFrame({
        'Interval': frequency_distribution['Interval'],
        'Relative Frequency': frequency_distribution['Frequency'] / total
    })
```

Rezultat:

Index	Interval	Relativna frekvencija
0	(1500.0, 1812.5)	0.08
1	(1812.5, 2125.0)	0.08
2	(2125.0, 2437.5)	0.1
3	(2437.5, 2750.0)	0.12
4	(2750.0, 3062.5)	0.26
5	(3062.5, 3375.0)	0.12
6	(3375.0, 3687.5)	0.12
7	(3687.5, 4000.0)	0.12

2.3 Razdioba kumulativnih relativnih frekvencija

Kumulativne frekvencije su frekvencije dobivene postupnim redoslijednim zbrajanjem učestalosti kojima su se pojavili pojedinačni rezultati u uzorku, odnosno ukupan zbroj frekvencija kojima se pojavio neki rezultat i svi rezultati koji su mi po numeričkom ili nekom drugom redoslijedu prethodili.

Računanje kumulativnih frekvencije u *Python-u* se može ovako izvesti:

```
def create_cumulative_relative_frequency_distribution(relative_frequency_distribution):
    cumulative_relative_frequency = relative_frequency_distribution['Relative Frequency'].cumsum()
    return pd.DataFrame({'Interval': relative_frequency_distribution['Interval'],
                         'Cumulative Relative Frequency': cumulative_relative_frequency})
```

Rezultat:

Index	Interval	Kumulativna relativna frekvencija
0	(1500.0, 1812.5)	0.08
1	(1812.5, 2125.0)	0.16
2	(2125.0, 2437.5)	0.26
3	(2437.5, 2750.0)	0.38
4	(2750.0, 3062.5)	0.64
5	(3062.5, 3375.0)	0.76
6	(3375.0, 3687.5)	0.88
7	(3687.5, 4000.0)	1

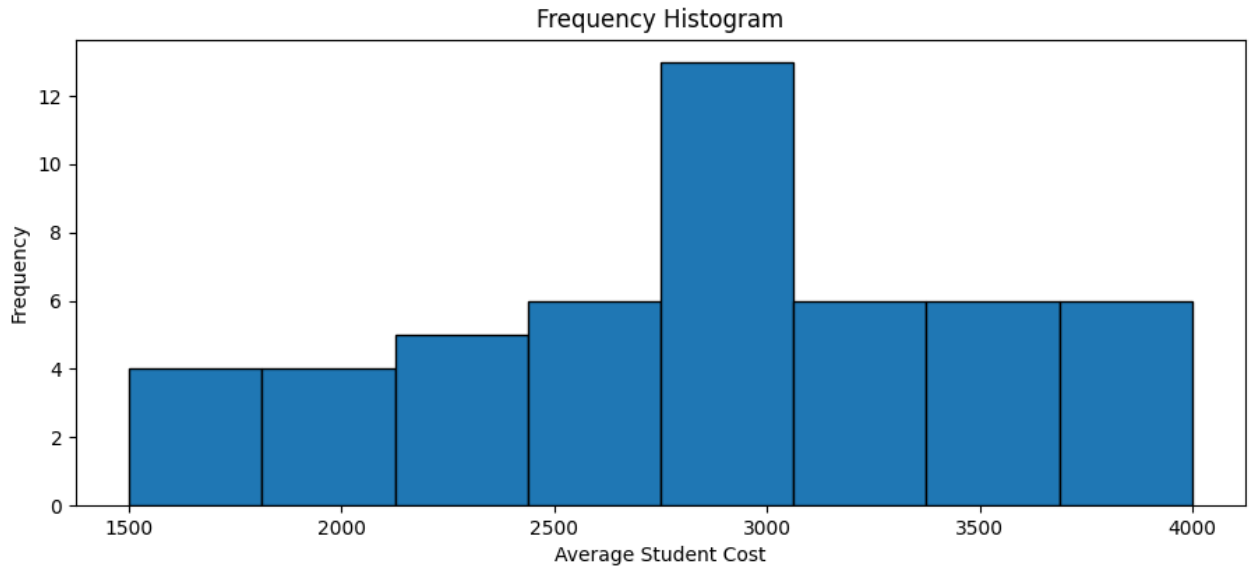
3. Zadatak

3. 1 Grafički prikaz histograma frekvencije

Prikaz histograma frekvencije u *Python*-u:

```
plt.figure(figsize=(12, 6))
plt.subplot(2, 2, 1)
plt.hist(data, bins=number_of_classes, edgecolor='black')
plt.title('Frequency Histogram')
plt.xlabel('Average Student Cost')
plt.ylabel('Frequency')
```

Rezultat:

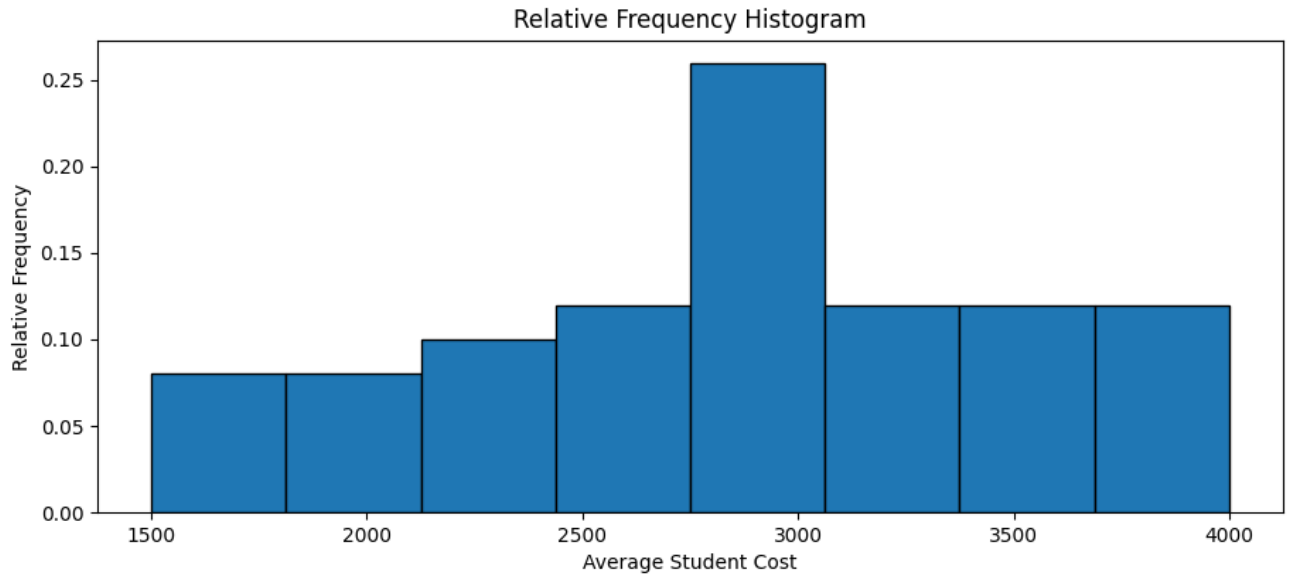


3.2 Grafički prikaz histograma relativnih frekvencija

Prikaz histograma relativne frekvencije u *Python*-u:

```
plt.subplot(2, 2, 2)
plt.hist(data, bins=number_of_classes, weights=np.zeros_like(data) + 1. / len(data), edgecolor='black')
plt.title('Relative Frequency Histogram')
plt.xlabel('Average Student Cost')
plt.ylabel('Relative Frequency')
```

Rezultat:



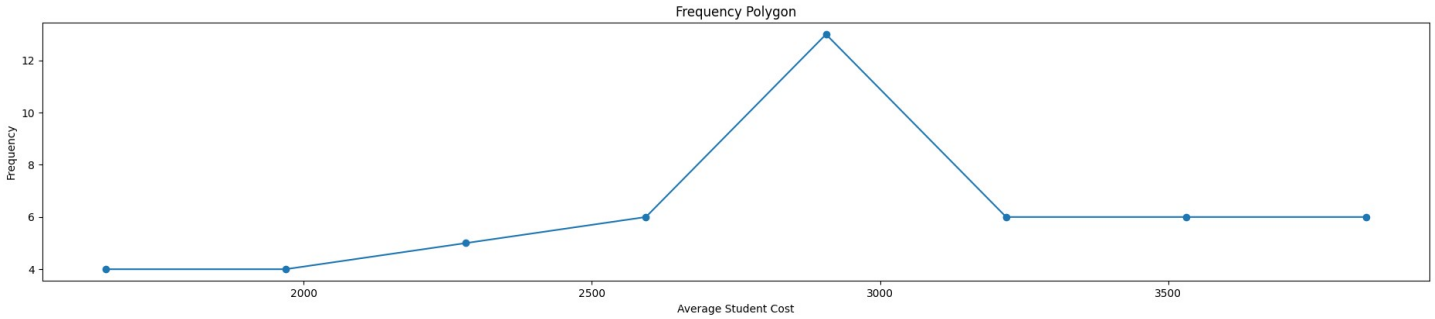
3.3 Grafički prikaz poligona frekvencija

Prikaz poligna frekvencije u *Python*-u:

```
plt.subplot(2, 1, 2)
plt.plot(bin_centers, frequencies, marker='o', linestyle='-')
plt.title('Frequency Polygon')
plt.xlabel('Average Student Cost')
plt.ylabel('Frequency')

plt.tight_layout()
plt.show()
```

Rezultat:



4. Zadatak

4.1 Interval povjerenja o očekivanju populacije s proizvoljnom pouzdanošću (90%, 95%, 99%)

Kvalitetniju procjenu nepoznatog parametra $\emptyset$ populacije daje nam intervalna procjena. Želimo odrediti interval za kojeg možemo s određenom vjerojatnošću tvrditi da sadrži nepoznati parametar $\emptyset$. Problem se može rješiti tako da odredimo sljedeće dvije statistike:

$$\emptyset_1 = f_1\left(X_1\ ,\ \dots\ ,\ X_n\right)$$

$$\emptyset_2 = f_2\left(X_1, X_2 \dots, X_n\right)$$

tako da je

$$P\left(\emptyset_1 \leq \emptyset_2\right) = 1$$

$$P\left(\emptyset_1 \leq \emptyset \leq \emptyset_2\right) = \beta$$

pri čemu je realni broj β(0 < β < 1) zadana vrijednost interval $[\emptyset_1, \emptyset_2]$ je slučajni interval jer su mu krajnje točke varijable. Nazivamo ga **interval povjerenja** (pouzdanosti) ta parametar $\emptyset$. Vjerojatnost β nazivamo **nivo povjerenja** (pouzdanosti).

Python kod za računanje interval povjerenja:

```
def calculate_confidence_interval(data: list, confidence_level=90):
    mean, std = np.mean(data), np.std(data, ddof=1)
    n, alpha = len(data) - 1, 1 - confidence_level
    margin_of_error = t.ppf(1 - alpha / 2, n) * (std / np.sqrt(n))
    lower_bound, upper_bound = mean - margin_of_error, mean + margin_of_error
    return (lower_bound, upper_bound)

confidence_90 = calculate_confidence_interval(data, 0.9)
confidence_95 = calculate_confidence_interval(data, 0.95)
confidence_99 = calculate_confidence_interval(data, 0.99)
```

Rezultat:

- 90%: (2691.756930440288, 2992.243069559712)
- 95%: (2661.9131602882017, 3022.0868397117983)
- 99%: (2601.837764106714, 3082.162235893286)

5. Zadatak

5.1 Hipoteza o očekivanju populacije

Svaku pretpostavku (tvrdnju) koja se odnosi na danu razdiobu obilježja nazivamo **statističkom hipotezom**. Provjeru istinitosti hipoteze pomoću uzorka nazivamo **statističkim testom** (testiranjem hipoteza).

Prilikom testiranja postavljaju se dvije hipoteze:

- **Nulta hipoteza** H_0
- **Alternativna hipoteza** H_1

Pri testiranju hipoteze H_0, na osnovi uzorka se donosi odluka o prihvaćanju ili odbacivanju hipoteze H_0. Postupak testiranja hipoteze provodi se u nekoliko koraka:

1) Definiraju se hipoteze H_0 i H_1
2) Zadaje se nivo značajnosti
3) Na osnovi uzorka $(X_1, X_2, ..., X_n)$ bira se procjenitelj zaodređeni parametar na osnovi kojeg donosimo odluke
4) Određuje se **kritično područje** – područje odbacivanja osnovne hipoteze H_0 u korist alternativne H_1
5) Odlučuje se o prihvaćanju ili odbacivanju hipoteze H_0

Testiranje na papiru:

n = 50

s = 620.996

$\bar{x} = 2842$

$\alpha = 0.05$

---

a.) $H_0 ... \mu_0 = 2642$

$H_0 ... \mu_1 \neq 2642$

b.) $\alpha = 0.05$

$\beta = 1 - 0.05 = 0.95$

$Z_{(B/2)} \approx 1.96$

c.) $t = \left(\frac{\bar{x} - \mu_0}{s}\right) * \sqrt{n}$

$t \approx 2.28$

d.) $C_{\propto} = <-\infty, -Z_{(B/2)}] \cup [Z_{(B/2)}, +\infty>$

$C_{\propto} = <-\infty, -2.28] \cup [2.28, +\infty>$

e.) $t \in C_{\alpha} \rightarrow H_0$ *prihvaćamo*

Testiranje u Python-u:

```
def hypothesis_test(data, hypothesis_mean, significance_level):
    sample_size, sample_mean, std_dev = len(data), np.mean(data), np.std(data)

    z_score = (sample_mean - hypothesis_mean) / (std_dev / np.sqrt(sample_size))
    critical_value = norm.ppf(1 - (significance_level / 2))

    return z_score, critical_value

z_score, critical_value = hypothesis_test(data, 2642, 0.05)
print("Z Score: ", z_score)
print("Critical Value: ", critical_value)

if np.abs(z_score) > critical_value:
    print("Fail to reject the null hypothesis")
else:
    print("Reject the null hypothesis")
```

Rezultat:

```
Z Score:  2.2773312888939934
Critical Value:  1.959963984540054
Failed to reject the null hypothesis
```