Introduktion til deskriptiv statistik i Python og numpy

Af Henrik Sterner (henrik.sterner@gmail.com)

Deskriptiv statistik er en gren af statistik, der beskæftiger sig med at beskrive data.

Deskriptiv statistik er en gren af statistik, der beskæftiger sig med at beskrive data. Det er en af de mest grundlæggende discipliner inden for statistik og er en forudsætning for at kunne analysere data. I det følgende gennemgås nogle af de mest basale begreber inden for deskriptiv statistik og hvordan de kan beregnes i Python ved brug af numpy.

Vigtige begreber

- Middelværdi
- Median
- Varians
- frekvenser
- akkumulerede frekvenser
- standardafvigelse
- kvartilsæt
- histogram
- boxplot

Datasæt

I det følgende vi arbejde med nogle højde- og vægtmålinger af 10 personer. Målingerne er foretaget på en tilfældigt udvalgt gruppe af personer i Danmark. Målingerne er foretaget med en nøjagtighed på 1 cm og 1 kg:

```
import numpy as np
import matplotlib.pyplot as plt

# Højde i cm
hoejde = np.array([ 184, 159, 192, 171, 185, 175, 169, 181,
# Vægt i kg
vaegte = np.array([ 74,  58,  96,  70,  80,  75,  65,  80,
```

Middelværdi

Middelværdien er gennemsnittet af alle målingerne. Middelværdien beregnes ved at lægge alle målingerne sammen og dividere med antallet af målinger. Middelværdien er et centralt mål for dataene, da den ligger midt i dataene. Middelværdien er den værdi, der er mest repræsentativ for dataene.

Givet en række målinger $x_1, x_2, x_3, \ldots, x_n$ er middelværdien givet ved:

Den generelle formel for middelværdi er:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$$

Beregning af middelværdi i Python ved brug af numpy

Middelværdien beregnes i Python ved brug af funktionen `mean()` fra numpy:

```
# Beregning af middelværdi
hoejde_mean = np.mean(hoejde)
vaegt_mean = np.mean(vaegte)
print("Middelværdi for højde:", hoejde_mean)
print("Middelværdi for vægt:", vaegt_mean)
```

Frekvenser

Frekvensen angiver, hvor mange gange en bestemt værdi forekommer i datasættet. Frekvensen angiver, hvor mange gange en bestemt værdi forekommer i datasættet.

Givet en række målinger $x_1, x_2, x_3, \ldots, x_n$ er frekvensen for en værdi x_i givet ved:

$$f_i = \sum_{j=1}^{n} \delta_{ij}$$

hvor δ_{ij} er en funktion, der er givet ved:

$$\delta_{ij} = \begin{cases} 1 & \text{hvis } x_i = x_j \\ 0 & \text{hvis } x_i \neq x_j \end{cases}$$

Beregning af frekvenser i Python ved brug af numpy

Frekvenserne beregnes i Python ved brug af funktionen `bincount()` fra numpy:

```
# Beregning af frekvenser
hoejde_freq = np.bincount(hoejde)
vaegt_freq = np.bincount(vaegte)
print("Frekvenser for højde:", hoejde_freq)
print("Frekvenser for vægt:", vaegt_freq)
```

Visualisering af frekvenser

Frekvenserne kan visualiseres ved brug af et histogram. Et histogram er en graf, der viser frekvenserne for de forskellige værdier i datasættet. Histogrammet er en graf, der viser frekvenserne for de forskellige værdier i datasættet.

Histogrammet kan visualiseres i Python ved brug af funktionen `hist()` fra matplotlib.pyplot:

```
# Visualisering af frekvenser
plt.figure()
plt.hist(hoejde)
plt.title("Frekvenser for højde")
plt.xlabel("Højde")
plt.ylabel("Frekvens")
plt.show()

plt.figure()
plt.hist(vaegte)
plt.title("Frekvenser for vægt")
```

Akkumulerede frekvenser

Akkumulerede frekvenser angiver, hvor mange gange en bestemt værdi eller en mindre værdi forekommer i datasættet. Akkumulerede frekvenser angiver, hvor mange gange en bestemt værdi eller en mindre værdi forekommer i datasættet.

Givet en række målinger $x_1, x_2, x_3, \ldots, x_n$ er den akkumulerede frekvens for en værdi x_i givet ved:

$$F_i = \sum_{j=1}^{i} f_j$$

hvor f_j er frekvensen for værdien x_j.

Beregning af akkumulerede frekvenser i Python ved brug af numpy

Akkumulerede frekvenser beregnes i Python ved brug af funktionen `cumsum()` fra numpy:

```
# Beregning af akkumulerede frekvenser
hoejde_cumfreq = np.cumsum(hoejde_freq)
vaegt_cumfreq = np.cumsum(vaegt_freq)
print("Akkumulerede frekvenser for højde:", hoejde_cumfreq)
print("Akkumulerede frekvenser for vægt:", vaegt_cumfreq)
```

Visualisering af akkumulerede frekvenser

Akkumulerede frekvenser kan visualiseres ved brug af et trappediagram. Et trappediagram er en graf, der viser akkumulerede frekvenser for de forskellige værdier i datasættet. Trappediagrammet er en graf, der viser akkumulerede frekvenser for de forskellige værdier i datasættet.

Trappediagrammet kan visualiseres i Python ved brug af funktionen `bar()` fra matplotlib.pyplot:

```python
# Visualisering af akkumulerede frekvenser
plt.figure()
plt.bar(np.arange(len(hoejde_cumfreq)), hoejde_cumfreq)
plt.title("Akkumulerede frekvenser for højde")
plt.xlabel("Højde")
plt.ylabel("Akkumuleret frekvens")
plt.show()
```

Median

Medianen er den midterste værdi i en række af målinger. Medianen er et centralt mål for dataene, da den ligger midt i dataene. Medianen er den værdi, der er mest repræsentativ for dataene.

Givet en række målinger $x_1, x_2, x_3, \ldots, x_n$ er medianen givet ved:

Den generelle formel for medianen er:

$$\tilde{x} = \begin{cases} x_{\frac{n+1}{2}} & \text{hvis } n \text{ er ulige} \\ \frac{1}{2}\left(x_{\frac{n}{2}} + x_{\frac{n}{2}+1}\right) & \text{hvis } n \text{ er lige} \end{cases}$$

Beregning af median i Python ved brug af numpy

Medianen beregnes i Python ved brug af funktionen `median()` fra numpy:

```
# Beregning af median
hoejde_median = np.median(hoejde)
vaegt_median = np.median(vaegte)
print("Median for højde:", hoejde_median)
print("Median for vægt:", vaegt_median)
```

Varians

Variansen er et mål for spredningen af dataene. Variansen er givet ved gennemsnittet af kvadratet på afvigelserne fra middelværdien. Variansen er et mål for spredningen af dataene. Jo større variansen er, jo mere spredte er dataene.

Givet en række målinger $x_1, x_2, x_3, \ldots, x_n$ er variansen givet ved.

Den generelle formel for variansen er:

$$s^2 = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})^2$$

Beregning af varians i Python ved brug af numpy

Variansen beregnes i Python ved brug af funktionen `var()` fra numpy:

```
# Beregning af varians

hoejde_var = np.var(hoejde, ddof=1)
vaegt_var = np.var(vaegte, ddof=1)
print("Varians for højde:", hoejde_var)
print("Varians for vægt:", vaegt_var)
```

Her er ddof=1, da vi ønsker at beregne variansen for et udsnit af en population. Den generelle formel for variansen er beregnet ud fra et helt datasæt, hvorfor vi skal dividere med n-1. Hvis vi havde ønsket at beregne variansen for hele populationen, skulle vi have sat ddof=0.

Standardafvigelse

Standardafvigelsen er kvadratroden af variansen. Standardafvigelsen er et mål for spredningen af dataene. Jo større standardafvigelsen er, jo mere spredte er dataene.

Givet en række målinger $x_1, x_2, x_3, \ldots, x_n$ er standardafvigelsen givet ved:

$$s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

Beregning af standardafvigelse i Python ved brug af numpy

Standardafvigelsen beregnes i Python ved brug af funktionen `std()` fra numpy:

```
# Beregning af standardafvigelse
hoejde_std = np.std(hoejde, ddof=1)
vaegt_std = np.std(vaegte, ddof=1)
print("Standardafvigelse for højde:", hoejde_std)
print("Standardafvigelse for vægt:", vaegt_std)
```

Typetal

Typetallet er den værdi, der forekommer flest gange i datasættet.
Typetallet er den værdi, der forekommer flest gange i datasættet.

Givet en række målinger $x_1, x_2, x_3, \ldots, x_n$ er typetallet givet ved:

$$\text{mode} = \underset{x_i}{\text{argmax}} \sum_{j=1}^{n} \delta_{ij}$$

hvor δ_{ij} er en funktion, der er givet ved:

$$\delta_{ij} = \begin{cases} 1 & \text{hvis } x_i = x_j \\ 0 & \text{hvis } x_i \neq x_j \end{cases}$$

Beregning af typetal i Python ved brug af numpy

Typetallet beregnes i Python ved brug af funktionen `mode()` fra numpy:

```
# Beregning af typetal
hoejde_mode = np.mode(hoejde)
vaegt_mode = np.mode(vaegte)
print("Typetal for højde:", hoejde_mode)
print("Typetal for vægt:", vaegt_mode)
```

Kvartilsæt

Et kvartilsæt er et sæt af fire værdier, der deler datasættet op i fire lige store dele. Et kvartilsæt er et sæt af fire værdier, der deler datasættet op i fire lige store dele.

Givet en række målinger $x_1, x_2, x_3, \ldots, x_n$ er kvartilsættet givet ved:

$$Q_1 = x_{\frac{n+1}{4}}$$

$$Q_2 = \begin{cases} x_{\frac{n+1}{2}} & \text{hvis } n \text{ er ulige} \\ \frac{1}{2}\left(x_{\frac{n}{2}} + x_{\frac{n}{2}+1}\right) & \text{hvis } n \text{ er lige} \end{cases}$$

$$Q_3 = x_{\frac{3(n+1)}{4}}$$

$$Q_4 = x_n$$

Beregning af kvartilsæt i Python ved brug af numpy

Kvartilsættet beregnes i Python ved brug af funktionen `quantile()` fra numpy:

```
# Beregning af kvartilsæt
hoejde_quantiles = np.quantile(hoejde, [0.25, 0.5, 0.75])
vaegt_quantiles = np.quantile(vaegte, [0.25, 0.5, 0.75])
print("Kvartilsæt for højde:", hoejde_quantiles)
print("Kvartilsæt for vægt:", vaegt_quantiles)
```

Visualisering af kvartilsæt

Kvartilsættet kan visualiseres ved brug af et boxplot. Et boxplot er en graf, der viser kvartilsættet for datasættet. Boxplot er en graf, der viser kvartilsættet for datasættet.

Boxplot kan visualiseres i Python ved brug af funktionen `boxplot()` fra matplotlib.pyplot:

```
# Visualisering af kvartilsæt
plt.figure()
plt.boxplot(hoejde)
plt.title("Kvartilsæt for højde")
plt.ylabel("Højde")
plt.show()

plt.figure()
plt.boxplot(vaegte)
plt.title("Kvartilsæt for vægt")
plt.ylabel("Vægt")
plt.show()
```