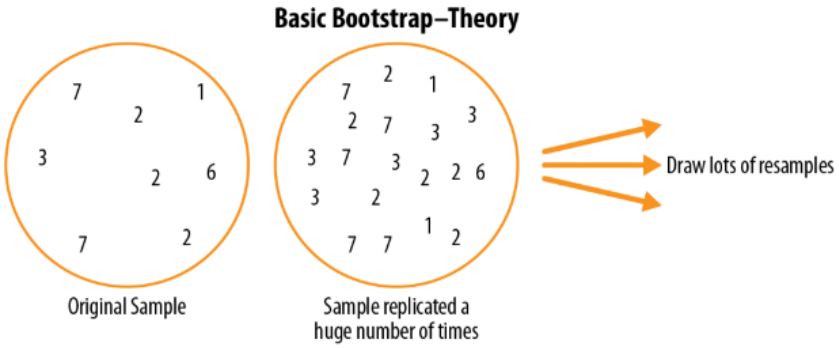
MODUL 6 KOMPUTASI STATISTIK

Resampling Methods: Bootstrap

Pendahuluan

Metode Bootstrap adalah metode resampling yang memungkinkan kita untuk memperkirakan distribusi statistik dengan mengambil sampel berulang kali dari data yang ada melalui proses pengambilan sampel dengan pengembalian (*sampling with replacement*).



Langkah-langkah Metode Bootstrap:

1. Ambil Sampel dari Data Asli

   Ambil sampel acak dari data asli dengan pengembalian *(sampling with replacement)*. Ini berarti beberapa elemen bisa muncul lebih dari sekali dalam sampel baru.

2. Hitung Statistik dari Sampel Bootstrap

   Hitung statistik yang akan diketahui (misalnya rata-rata, median, deviasi standar) dari sampel bootstrap yang diambil.

3. Ulangi Proses

   Ulangi langkah 1 dan 2 sebanyak n kali, untuk menghasilkan banyak nilai statistik dari berbagai sampel bootstrap.

4. Analisis Distribusi Statistik

   Gunakan distribusi dari nilai-nilai statistik yang diperoleh untuk memperkirakan ketidakpastian dari estimator.

Kelebihan Metode Bootstrap:

- Tidak Memerlukan Asumsi Distribusi: Bootstrap tidak memerlukan asumsi distribusi data yang kuat, sehingga bisa digunakan dalam berbagai situasi.
- Fleksibilitas: Dapat diterapkan pada hampir semua statistik dan estimator, baik sederhana maupun kompleks.
- Mudah Diimplementasikan: Konsep dan penerapan relatif sederhana terutama dengan bantuan perangkat lunak statistik.

Kekurangan Metode Bootstrap:

- Kebutuhan Komputasi: Proses bootstrap memerlukan komputasi yang besar, terutama jika jumlah iterasi yang dilakukan sangat banyak.
- Tidak Selalu Cocok untuk Semua Jenis Data: Dalam beberapa kasus, terutama dengan data yang sangat kecil atau sangat tidak terdistribusi, hasil bootstrap mungkin tidak stabil.

Contoh Kasus 1:

Misalkan kita memiliki dataset yang berisi nilai-nilai pengukuran suhu (dalam derajat Celsius) dari 10 lokasi {22.5, 21.8, 23.1, 22.9, 24.0, 21.6, 23.4, 22.1, 23.0, 22.8}

Gunakan teknik bootstrap untuk memperkirakan interval kepercayaan 95% untuk rata-rata suhu!

```
# Data asli
data <- c(22.5, 21.8, 23.1, 22.9, 24.0, 21.6, 23.4, 22.1, 23.0, 22.8)
set.seed(123)

# Parameter bootstrap
n_iterations <- 1000
n_size <- length(data)

# Menyimpan hasil bootstrap
bootstrap_means <- numeric(n_iterations)

for (i in 1:n_iterations) {
  sample <- sample(data, size = n_size, replace = TRUE)
  bootstrap_means[i] <- mean(sample)
}
```

```
# Hitung interval kepercayaan
lower_bound <- quantile(bootstrap_means, 0.025)
upper_bound <- quantile(bootstrap_means, 0.975)

#Output Hasil
cat(sprintf("Rata-rata data asli: %.2f\n", mean(data)))
cat(sprintf("Interval kepercayaan 95%% untuk rata-rata: (%.2f, %.2f)\n",
lower_bound, upper_bound))
```

Penjelasan:

- set.seed(): penggunaan seed memungkinkan hasil bootstrap yang konsisten setiap kali kode dijalankan.
- sample <- sample(data, size = n_size, replace = TRUE): Mengambil sampel bootstrap dengan pengembalian dari data. Artinya, kita mengambil 10 elemen secara acak dari data, dengan kemungkinan pengulangan elemen yang sama.
- bootstrap_means[i] <- mean(sample): Menghitung rata-rata dari sampel bootstrap yang diambil dan menyimpannya dalam elemen ke-i dari bootstrap_means.
- lower_bound <- quantile(bootstrap_means, 0.025): Menghitung batas bawah dari interval kepercayaan 95% dengan mengambil kuantil 2.5% dari distribusi bootstrap_means. Ini memberikan estimasi batas bawah dari interval kepercayaan.
- upper_bound <- quantile(bootstrap_means, 0.975): Menghitung batas atas dari interval kepercayaan 95% dengan mengambil kuantil 97.5% dari distribusi bootstrap_means. Ini memberikan estimasi batas atas dari interval kepercayaan.
- Fungsi cat() digunakan untuk mencetak output ke konsol di R. Fungsi ini menggabungkan string dan menulis hasilnya tanpa menambahkan karakter baris baru secara otomatis di akhir.
- Fungsi sprintf() digunakan untuk format string dengan cara yang mirip dengan fungsi printf di bahasa pemrograman lain. Ini memungkinkan Anda untuk menyusun string dengan variabel yang diformat dengan cara tertentu.
- %.2f menunjukkan bahwa nilai lower_bound dan upper_bound harus dicetak sebagai angka desimal dengan dua tempat desimal.
- %% adalah cara untuk mencetak tanda persen % dalam format string.
- \n adalah karakter newline, menambahkan baris baru setelah output.

Contoh Kasus 2:

Misalkan terdapat dataset yang berisi hasil tes matematika dasar dari 7 mahasiswa {75, 80, 85, 90, 88, 92, 86}. Gunakan teknik bootstrap untuk memperkirakan interval kepercayaan 95% untuk variansi hasil tes!

```
# Data asli
data <- c(75, 80, 85, 90, 88, 92, 86)
set.seed(123)

# Parameter bootstrap
n_iterations <- 1000
n_size <- length(data)

# Menyimpan hasil bootstrap
bootstrap_variances <- numeric(n_iterations)

for (i in 1:n_iterations) {
  sample <- sample(data, size = n_size, replace = TRUE)
  bootstrap_variances[i] <- var(sample)
}

# Hitung interval kepercayaan
lower_bound <- quantile(bootstrap_variances, 0.025)
upper_bound <- quantile(bootstrap_variances, 0.975)

#Output Hasil
cat(sprintf("Variansi data asli: %.2f\n", var(data)))
cat(sprintf("Interval kepercayaan 95%% untuk variansi: (%.2f, %.2f)\n",
lower_bound, upper_bound))
```

Penjelasan:

- sample <- sample(data, size = n_size, replace = TRUE):

  Mengambil sampel bootstrap dengan pengembalian dari data. Artinya, Anda membuat sampel baru yang berukuran sama dengan data asli, dengan kemungkinan elemen yang sama muncul lebih dari sekali.
- bootstrap_variances[i] <- var(sample):

Menghitung variansi dari sampel bootstrap yang diambil dan menyimpannya dalam elemen ke-i dari bootstrap_variances. Variansi dihitung menggunakan fungsi var(), yang mengukur seberapa tersebar data di sekitar rata-ratanya.

Contoh Kasus 3:

Misalkan terdapat dataset yang berisi data tinggi badan (x) dan berat badan (y) dari 6 individu yang diukur.

Tinggi badan (x): {150, 160, 170, 180, 190, 200}

Berat badan (y): {50, 60, 65, 75, 80, 90}

Gunakan teknik bootstrap untuk memperkirakan interval kepercayaan 95% untuk koefisien regresi dari model linear $y = \beta_0 + \beta_1 x$!

```
# Data asli
x <- c(150, 160, 170, 180, 190, 200)
y <- c(50, 60, 65, 75, 80, 90)
set.seed(123)

# Parameter bootstrap
n_iterations <- 1000
n_size <- length(x)

# Menyimpan hasil bootstrap
bootstrap_slopes <- numeric(n_iterations)

for (i in 1:n_iterations) {
  indices <- sample(1:n_size, size = n_size, replace = TRUE)
  x_sample <- x[indices]
  y_sample <- y[indices]
  model <- lm(y_sample ~ x_sample)
  bootstrap_slopes[i] <- coef(model)[2]
}

# Hitung interval kepercayaan
lower_bound <- quantile(bootstrap_slopes, 0.025)
upper_bound <- quantile(bootstrap_slopes, 0.975)
```

```
#Output Hasil
cat(sprintf("Koefisien regresi data asli: %.2f\n", coef(lm(y ~ x))[2]))
cat(sprintf("Interval kepercayaan 95%% untuk koefisien regresi: (%.2f, %.2f)\n", lower_bound, upper_bound))
```

Penjelasan:

- n_iterations <- 1000: Menetapkan jumlah iterasi untuk proses bootstrap. Di sini, kita melakukan bootstrap sebanyak 1000 kali.
- n_size <- length(x): Menyimpan ukuran sampel (jumlah data) ke dalam variabel n_size.
- bootstrap_slopes <- numeric(n_iterations)*:* Menciptakan vektor bootstrap_slopes dengan panjang 1000 (jumlah iterasi), yang akan menyimpan koefisien regresi dari setiap iterasi bootstrap.
- indices <- sample(1:n_size, size = n_size, replace = TRUE): Mengambil sampel acak dengan pengembalian dari indeks data. Ini menciptakan sampel bootstrap dengan ukuran yang sama seperti data asli.
- x_sample <- x[indices]: Mengambil elemen dari x sesuai dengan indeks sampel bootstrap.
- y_sample <- y[indices]: Mengambil elemen dari y sesuai dengan indeks sampel bootstrap.
- model <- lm(y_sample ~ x_sample): Membuat model regresi linear pada data sampel bootstrap.
- bootstrap_slopes[i] <- coef(model)[2]: Menyimpan koefisien regresi (kemiringan) dari model regresi bootstrap ke dalam *bootstrap_slopes.*