Section2_PartA&B

Section 2 — Predicting Good Health (UNSDG 3)

Part A: Building the Model with Maximum Likelihood

Task 1 - Log-likelihood

We model whether life expectancy exceeds the median using a binary response and a single predictor (log GDP per capita).

- Data: (y_i, x_i) for $i = 1, \dots, n$, where $y_i \in \{0, 1\}$ and $x_i = \log(\text{gdpPercap}_i)$.
- Model: $\text{logit}(p_i) = \beta_0 + \beta_1 x_i$, where $p_i = \Pr(Y_i = 1 \mid x_i)$.

Likelihood

$$L(\beta_0, \beta_1) = \prod_{i=1}^{n} p_i^{y_i} (1 - p_i)^{1-y_i},$$

$$p_i = \frac{\exp(\beta_0 + \beta_1 x_i)}{1 + \exp(\beta_0 + \beta_1 x_i)}.$$

Log-likelihood

$$\ell(\beta_0, \beta_1) = \sum_{i=1}^{n} \Big[y_i(\beta_0 + \beta_1 x_i) \; - \; \log(1 + \exp(\beta_0 + \beta_1 x_i)) \Big].$$

Task 2 - Implement log-likelihood

```r
loglik_logistic <- function(par, y, x){
beta0 <- par[1]
beta1 <- par[2]
eta <- beta0 + beta1 * x
sum(y * eta - log(1 + exp(eta)))
}
```

This function `loglik_logistic` takes the parameter vector `par = c(beta0, beta1)` together with the data vectors `y` (binary 0/1) and `x = log(gdpPercap)`, forms the linear predictor i= 0+ 1xi _i= _0+ _1 x_i i= 0+ 1xi, and returns the single numeric value of the logistic/Bernoulli log-likelihood

Data preparation

```
library(dplyr)
```

```

Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(gapminder)

data <- gapminder %>%
  filter(year == 2007) %>%
mutate(
high_lifeExp = ifelse(lifeExp > median(lifeExp), 1, 0),
log_gdp = log(gdpPercap)
)
```

Response and Predictor

```
y <- data$high_lifeExp
x <- data$log_gdp
stopifnot(length(y) == length(x))
```

Task 3 - Maximise log-likelihood with `optim()`

```r
# Initial values for beta0 and beta1
par0 <- c(0, 0)

# Maximise the log likelihood
optim_fit <- optim(
par = par0,
fn = loglik_logistic,
y = y,
x = x,
method = "BFGS",
control = list(fnscale = -1),
hessian = TRUE
)

# Estimates and checks

beta_hat <- optim_fit$par
names(beta_hat) <- c("beta0_hat","beta1_hat")
ll_max <- optim_fit$value
conv <- optim_fit$convergence


beta_hat
```

```
beta0_hat beta1_hat
-20.38243   2.34889
```

```r
ll_max
```

```
[1] -42.44485
```

```r
conv
```

```
[1] 0
```

Task 4 - Fit the model with `glm()`

```r
model_glm <- glm(high_lifeExp ~ log_gdp, data = data, family = binomial)
coef_glm <- coef(model_glm)
coef_glm
```

```
(Intercept)     log_gdp
 -20.382309    2.348878
```

Task 5 - Compare `optim()` and `glm()` coefficients

```r
# Comparison
comp <- rbind(optim = beta_hat,
glm = coef_glm)
comp
```

```
      beta0_hat beta1_hat
optim -20.38243  2.348890
glm   -20.38231  2.348878
```

```r
# Numerical differences
diff <- comp["optim", ] - comp["glm", ]
diff
```

```
    beta0_hat     beta1_hat
-1.228719e-04  1.234431e-05
```

The estimates from `optim()` and `glm()` are essentially identical, differing only by numerical tolerance: about 10^{-4} for $\hat{\beta}_0$ and 10^{-5} for $\hat{\beta}_1$. This confirms that our log-likelihood implementation and maximisation with `optim()` (BFGS, `fnscale = -1`) recover the same MLEs as the built-in `glm()` fit.

Part B: Estimating Uncertainty with Fisher Information

Task 1 - Extract and display the Hessian matrix

```r
H <- optim_fit$hessian
H
```

```
           [,1]       [,2]
[1,]  -13.12452  -114.6084
[2,] -114.60837 -1007.5049
```

Task 2 - Compute and display the Fisher information matrix

```
I_hat <- -H
I_hat
```

```
           [,1]      [,2]
[1,]   13.12452  114.6084
[2,]  114.60837 1007.5049
```

Task 3 - Compute and display the standard errors

```
V_hat <- solve(I_hat)
se <- sqrt(diag(V_hat))
names(se) <- c("beta0","beta1")
se
```

```
    beta0     beta1
3.3848272 0.3863266
```

Task 4 - Manually construct the 95% confidence intervals

```
beta_hat_vec <- setNames(optim_fit$par, c("beta0","beta1"))
z <- 1.96
CI <- cbind(
lower = beta_hat_vec - z * se,
upper = beta_hat_vec + z * se
)
CI
```

```
          lower     upper
beta0 -27.01669 -13.74817
beta1   1.59169   3.10609
```