# STA3007_hw10_codes

## Yuzhou Peng

## 2025-04-20

```
library(np)
```

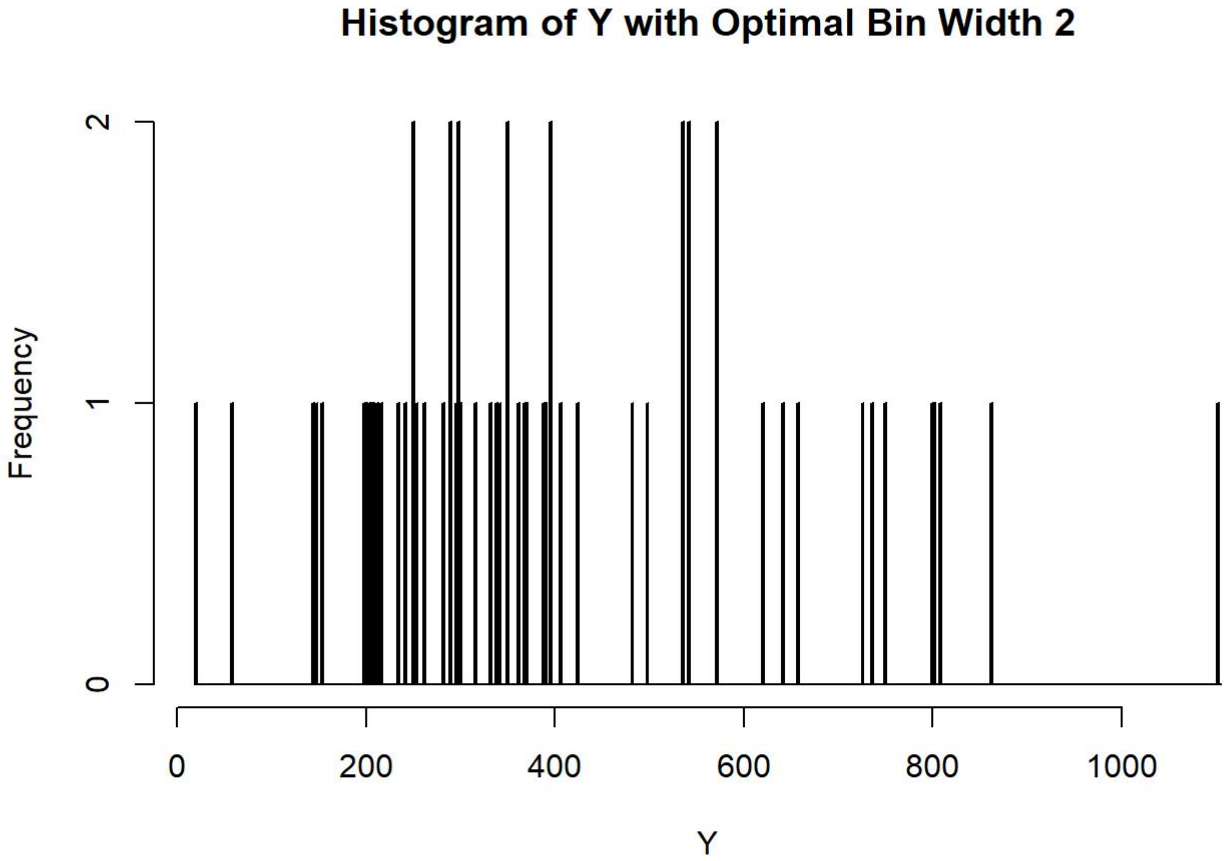
```
## Nonparametric Kernel Methods for Mixed Datatypes (version 0.60-18)
## [vignette("np_faq",package="np") provides answers to frequently asked questions]
## [vignette("np",package="np") an overview]
## [vignette("entropy_np",package="np") an overview of entropy-based methods]
```

```
library(ggplot2)
```

```
ceodat <- read.table("C:\\Users\\Penguin\\Desktop\\STA3007\\ceodat.txt")
ceodat <- ceodat[-31,]

ceodat <- ceodat[-1,]

colnames(ceodat) <- c("AGE", "SAL")
rownames(ceodat) <- 1:nrow(ceodat)

Y <- ceodat$SAL
X <- ceodat$AGE

Y <- as.numeric(Y)
X <- as.numeric(X)
```

#Histogram

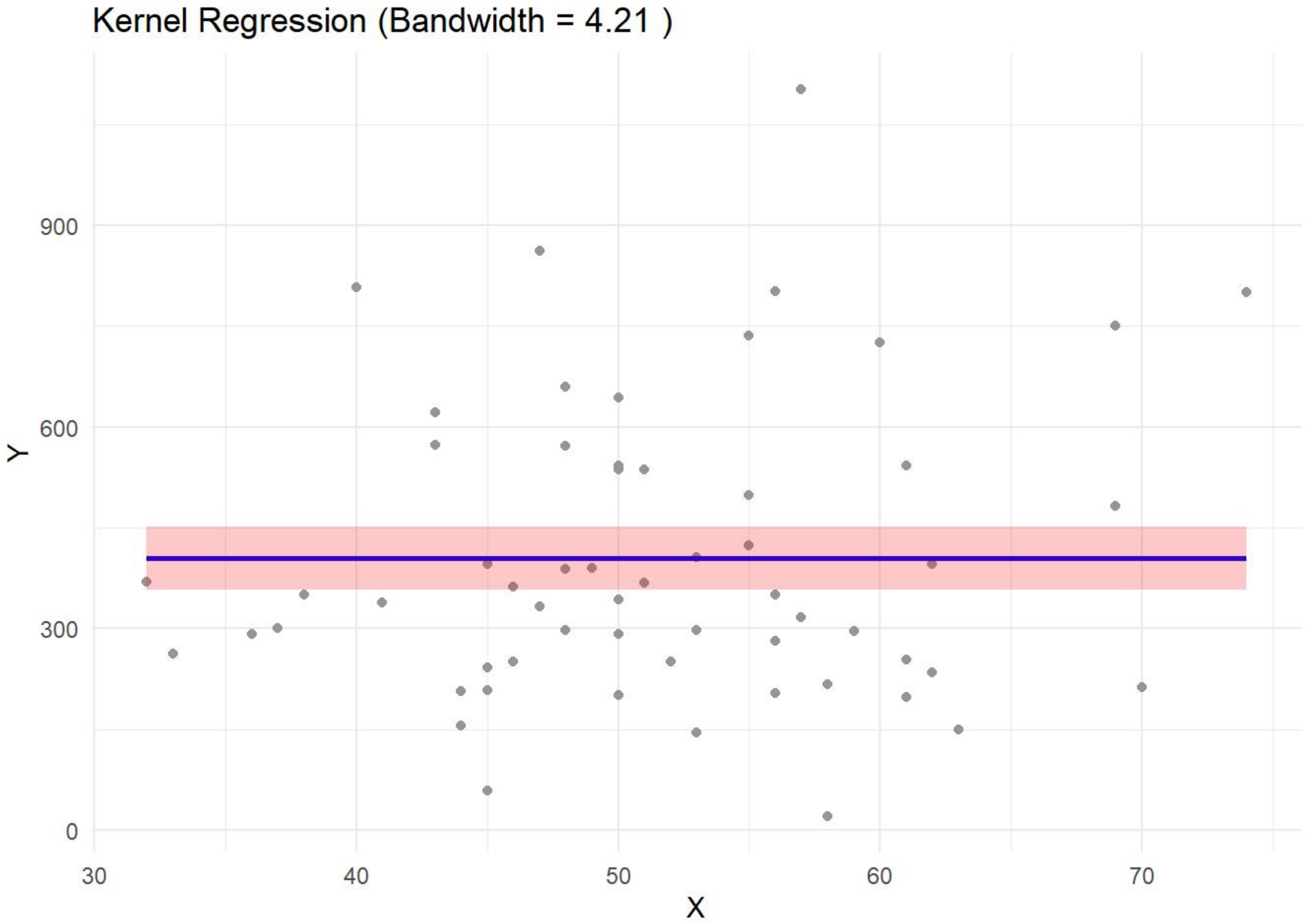
```
LSCV_hist <- function(Y, h) {
  n <- length(Y)
  if (n < 2) stop("Need at least two data points")
  range_Y <- range(Y)
  # Create breaks covering the data range with bin width h
  breaks <- seq(from = floor(range_Y[1]/h)*h - h,
                to = ceiling(range_Y[2]/h)*h + h,
                by = h)
  hist_counts <- hist(Y, breaks = breaks, plot = FALSE)$counts
  term1 <- sum(hist_counts^2) / (n^2 * h)
  sum_term2 <- sum(hist_counts * (hist_counts - 1))
  term2 <- 2 * sum_term2 / (n * h * (n - 1))
  return(term1 - term2)
}




h_values <- seq(0.1, 2, by = 0.1)  # Adjust based on data spread


lscores <- sapply(h_values, function(h) LSCV_hist(Y, h))


optimal_h <- h_values[which.min(lscores)]


hist(Y, breaks = seq(min(Y) - optimal_h, max(Y) + optimal_h, by = optimal_h),
     main = paste("Histogram of Y with Optimal Bin Width", round(optimal_h, 2)),
     xlab = "Y", col = "lightblue", border = "black")
```



# Kernel Method

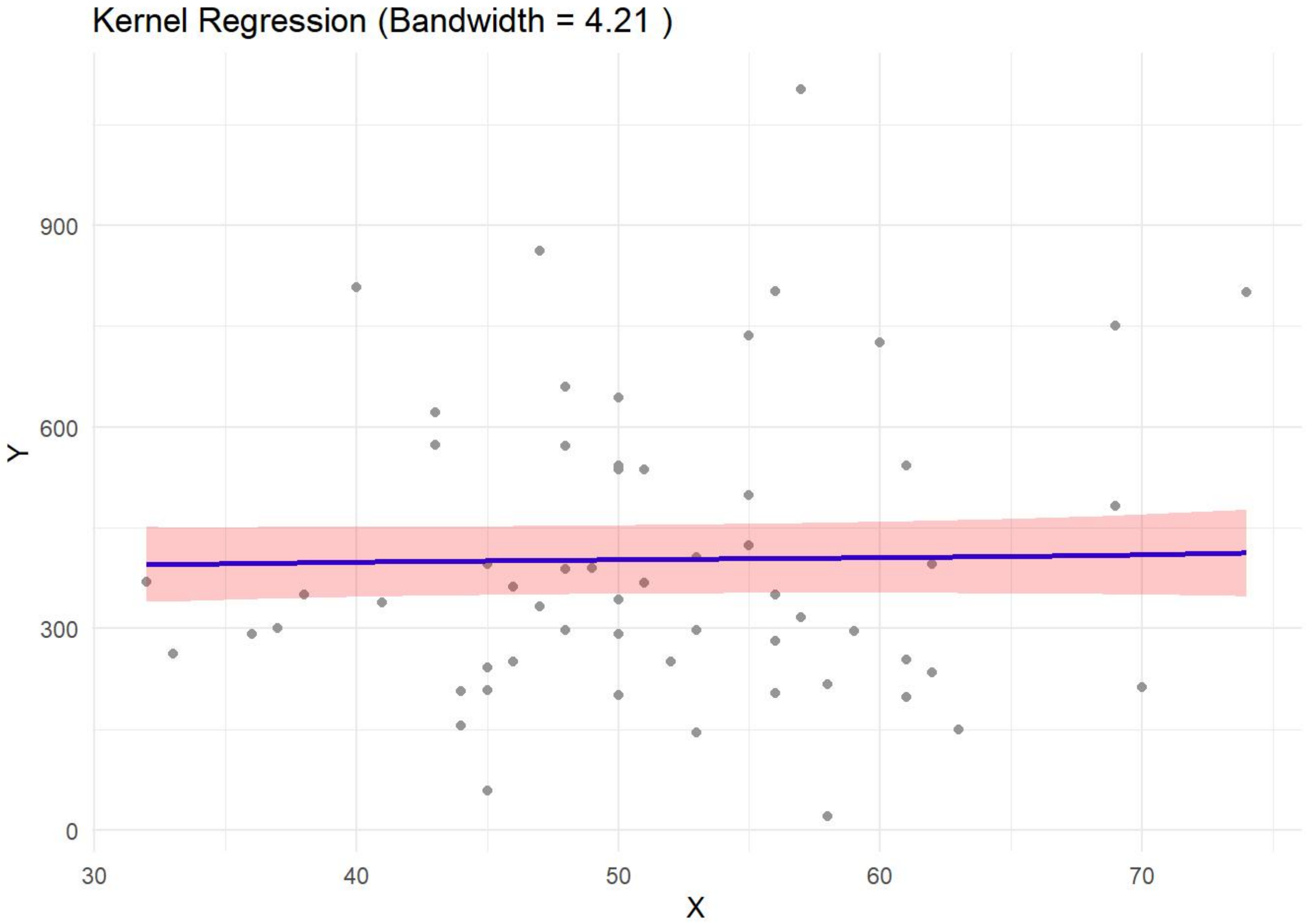
```
data <- data.frame(X = X, Y = Y)
n <- nrow(ceodat)

h_normal <- 1.06 * sd(X) * n^(-1/5)

# Fit kernel regression model with the computed bandwidth
bw <- npregbw(
  formula = Y ~ X,
  data = data,
  bws = h_normal,     # Use the precomputed bandwidth
  bwtype = "fixed",   # Fix the bandwidth (no cross-validation)
  ckertype = "gaussian" # Gaussian kernel
)
```

```
##
Multistart 1 of 1 |
Multistart 1 of 1 |
Multistart 1 of 1 |
Multistart 1 of 1 /
Multistart 1 of 1 |
Multistart 1 of 1 |
# Fit the model
model <- npreg(bw)

#  Generate predictions and confidence intervals
x_grid <- seq(min(X), max(X), length.out = 100) # Grid of X values
pred <- predict(model, newdata = data.frame(X = x_grid), se.fit = TRUE) # Predictions with SEs

# Compute 95% confidence bands
conf_lower <- pred$fit - 1.96 * pred$se.fit
conf_upper <- pred$fit + 1.96 * pred$se.fit

# Plot results with confidence bands
ggplot() +
  geom_point(data = data, aes(x = X, y = Y), color = "gray60") +
  geom_line(aes(x = x_grid, y = pred$fit), color = "blue", linewidth = 1) +
  geom_ribbon(
    aes(x = x_grid, ymin = conf_lower, ymax = conf_upper),
    fill = "red", alpha = 0.2
  ) +
  labs(
    title = paste("Kernel Regression (Bandwidth =", round(h_normal, 2), ")"),
    x = "X", y = "Y"
  ) +
  theme_minimal()
```


Kernel Regression (Bandwidth = 4.21 )
Y
900
600
300
0
30
40
50
60
70
X

```
data <- data.frame(X = X, Y = Y)
n <- nrow(ceodat)

h_normal <- 1.06 * sd(X) * n^(-1/5)

# Fit kernel regression model with the computed bandwidth
bw <- npregbw(
  formula = Y ~ X,
  data = data,
  bws = h_normal,     # Use the precomputed bandwidth
  bwtype = "fixed",   # Fix the bandwidth (no cross-validation)
  ckertype = "epanechnikov"
)
```

```
##
Multistart 1 of 1 |
Multistart 1 of 1 |
Multistart 1 of 1 |
Multistart 1 of 1 /
Multistart 1 of 1 |
Multistart 1 of 1 |
# Fit the model
model <- npreg(bw)

#  Generate predictions and confidence intervals
x_grid <- seq(min(X), max(X), length.out = 100) # Grid of X values
pred <- predict(model, newdata = data.frame(X = x_grid), se.fit = TRUE) # Predictions with SEs

# Compute 95% confidence bands
conf_lower <- pred$fit - 1.96 * pred$se.fit
conf_upper <- pred$fit + 1.96 * pred$se.fit

# Plot results with confidence bands
ggplot() +
  geom_point(data = data, aes(x = X, y = Y), color = "gray60") +
  geom_line(aes(x = x_grid, y = pred$fit), color = "blue", linewidth = 1) +
  geom_ribbon(
    aes(x = x_grid, ymin = conf_lower, ymax = conf_upper),
    fill = "red", alpha = 0.2
  ) +
  labs(
    title = paste("Kernel Regression (Bandwidth =", round(h_normal, 2), ")"),
    x = "X", y = "Y"
  ) +
  theme_minimal()
```


Kernel Regression (Bandwidth = 4.21 )
Y
900
600
300
0
30
40
50
60
70
X