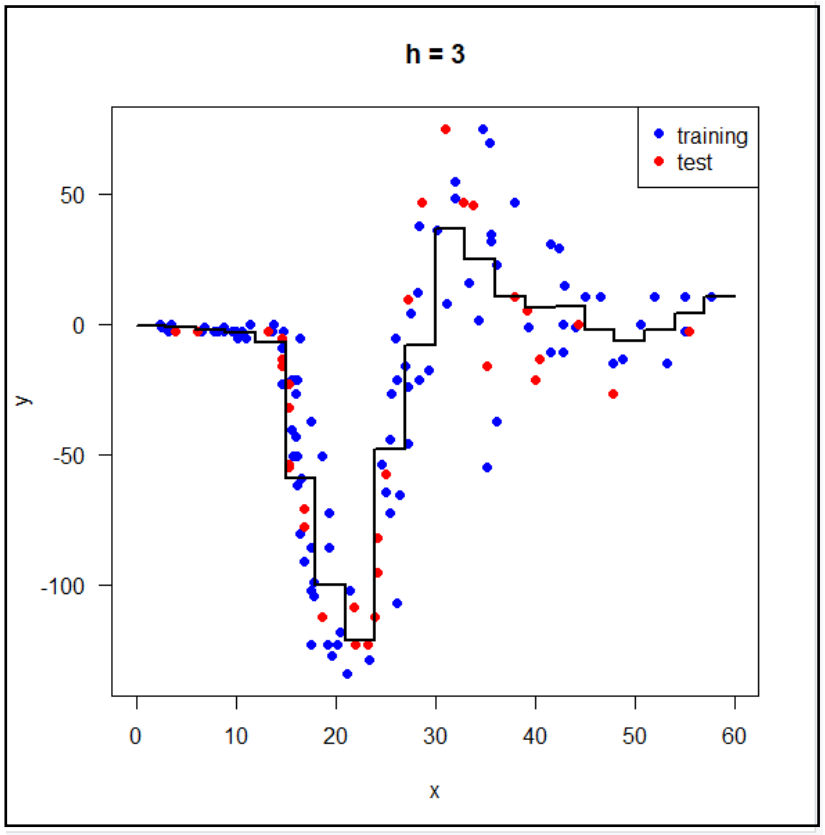
COMP421 HW#4, Prepared by Arda Kırkağaç, 0049799

In this homework, our aim was to train some nonparametric classifiers and then test it on our test data to show effectiveness. In the first step, I defined the training and test data as you can see in above figure.

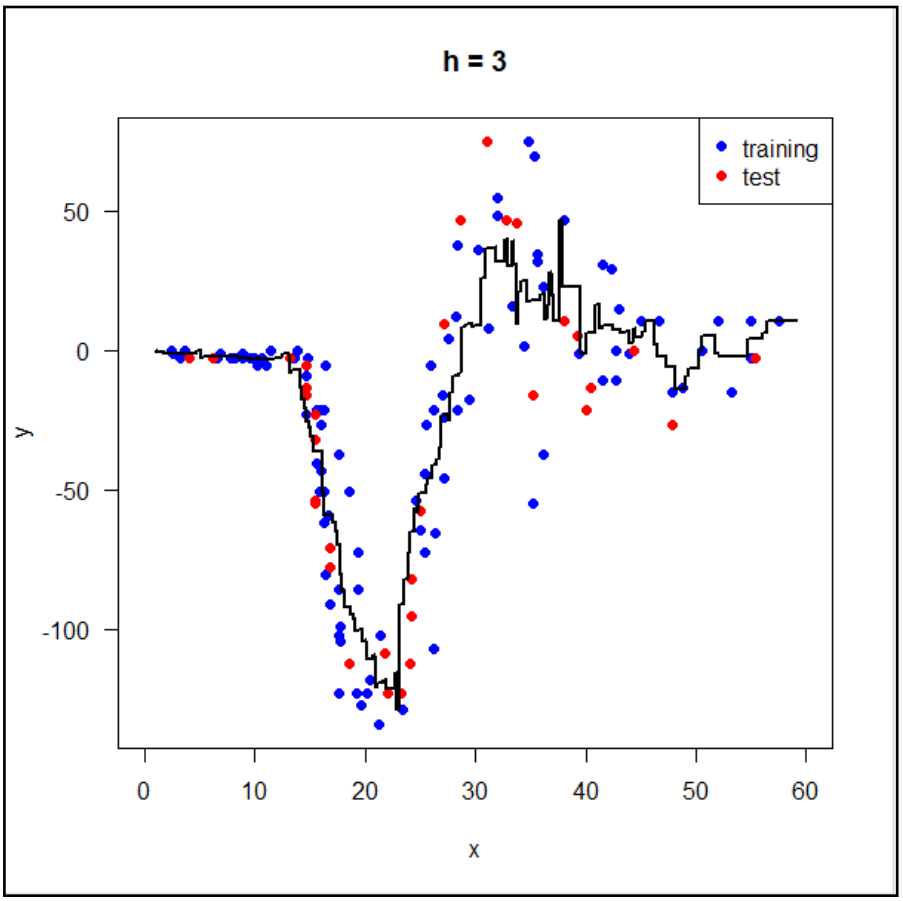
```
#separate train and test data
train_indices <-  c(sample(which(x==x),100))
x_train <- x[train_indices]
y_train <- y[train_indices]
x_test <- x[-train_indices]
y_test <- y[-train_indices]
```

Then, I first defined the regressogram. It basically separates the training data into bins, and for each bin, calculates the mean y value for the points whose x values fall into that particular bin. You can see my resulting plot on the right, as the bin width set to 3. Blue points correspond to training data, and the red ones to test data.



The resulting error value is at the end of this submission. I basically compared each test point's y value, with calculated means, after looking at which bin each point falls.

The second assignment was to train a running mean smoother using 100 points as training data, and the remaining 33 points as test data to calculate our prediction error. I calculated the running means for test data separately. However, it is also functional that we match each test point to the nearest point of our "data_interval", which we define in order to provide graphs between 0 and 60. My resulting plot is below with bin width set to 3:

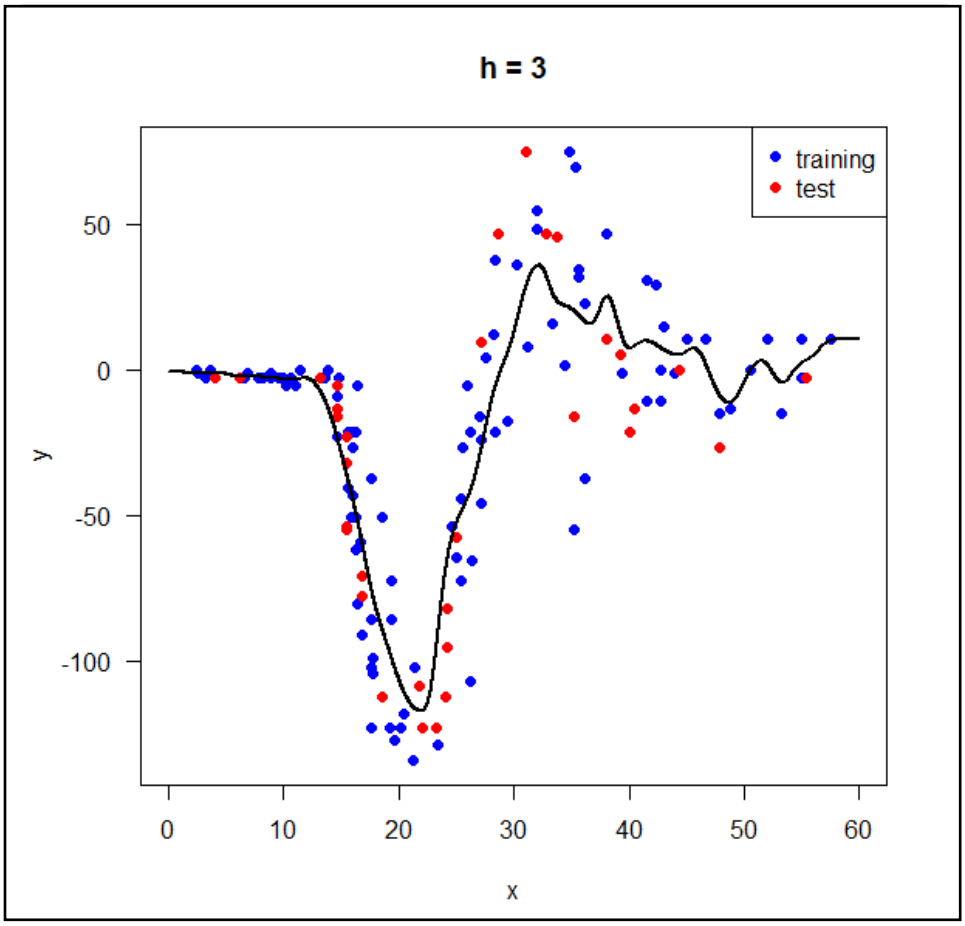


The final assignment was to train a kernel smoother over training data. Here is my definition of the kernel function, and the calculation of predictions over the data interval:

```
#kernel function
kernel <- function(u) {
  return (1 / sqrt(2 * pi) * exp(-0.5 * (u^2)))
}

g_head <- sapply(data_interval, function(X) {
                  sum(y_train*kernel((X-x_train)/bin_width3))
                 /sum(kernel((X-x_train)/bin_width3))
    })
```

You can see my resulting plot below. Blue points correspond to training data, and the red ones to test data:



The error values calculated for each part is shown below, where the values match the ones in the assignment document. For running mean smoother and kernel smoother, test data were processed with the estimator to reach precise values for corresponding x indices. For the regressogram, I ran an error calculation over the bins, returning an array having a length of the number of bins. Then the values are added together to obtain the total value.

```
[1] "Regressogram => RMSE is 21.659496 when h is 3"
> sprintf("Running Mean Smoother => RMSE is %f when h is %i", RMSE_value2, bin_width2)
[1] "Running Mean Smoother => RMSE is 21.921977 when h is 3"
> sprintf("Kernel Smoother => RMSE is %f when h is %i", RMSE_value3, bin_width3)
[1] "Kernel Smoother => RMSE is 22.108834 when h is 1"
>
```