# Exercise 8 Advanced Methods for Regression and Classification

Stefan Merdian

2024-12-09

```
library(gclus)
```

```
## Warning: package 'gclus' was built under R version 4.4.2
```

```
## Loading required package: cluster
```

```
data("ozone", package="gclus")
head(ozone)
```

```
##   Ozone Temp InvHt Pres Vis  Hgt Hum InvTmp Wind
## 1     3   40  2693  -25 250 5710  28  47.66    4
## 2     5   45   590  -24 100 5700  37  55.04    3
## 3     5   54  1450   25  60 5760  51  57.02    3
## 4     6   35  1568   15  60 5720  69  53.78    4
## 5     4   45  2631  -33 100 5790  19  54.14    6
## 6     4   55   554  -28 250 5790  25  64.76    3
```

```
set.seed(123)
n <- nrow(ozone)
train_indices <- sample(1:n, size = floor(2 * n / 3))
train_data <- ozone[train_indices, ]
test_data <- ozone[-train_indices, ]
```

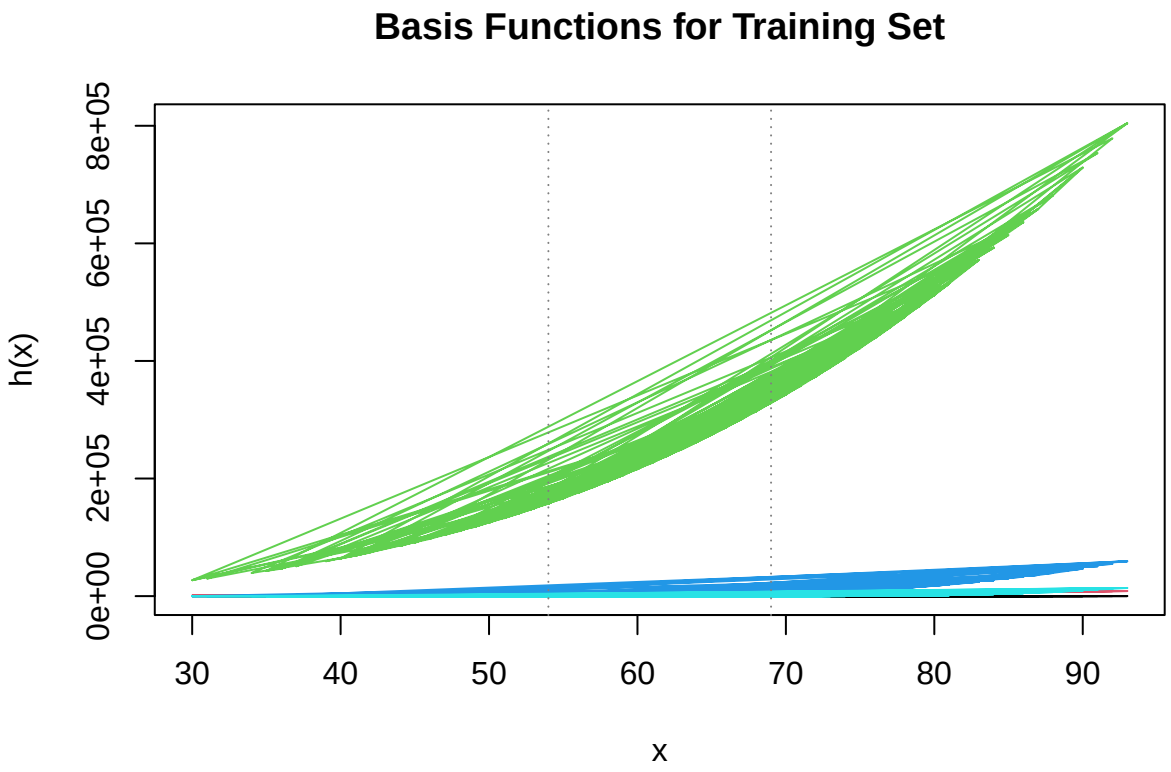
We first defining the basisfunction from the lecture as well as given plot function.

```
lecturespl <- function(x, nknots = 2, M = 4) {
  n <- length(x)
  # X will not get an intercept column
  X <- matrix(NA, nrow = n, ncol = (M - 1) + nknots)
  for (i in 1:(M - 1)) { X[, i] <- x^i }
  # now the basis functions for the constraints:
  quant <- seq(0, 1, 1 / (nknots + 1))[c(2:(nknots + 1))]
  qu <- quantile(x, quant)
  for (i in M:(M + nknots - 1)) {
    X[, i] <- ifelse(x - qu[i - M + 1] < 0, 0, (x - qu[i - M + 1])^(M - 1))
  }
  list(X = X, quantiles = quant, xquantiles = qu)
}
```

```r
plotspl <- function(splobj, ...) {
  matplot(splobj$x, splobj$X, type = "l", lty = 1,
          xlab = "x", ylab = "h(x)", ...)
  abline(v = splobj$xquantiles, lty = 3, col = gray(0.5))
}
```

## 1) Visualize basis functions

```r
train_spline <- lecturespl(train_data$Temp, nknots = 2, M = 4)

train_spline$x <- train_data$Temp

plotspl(train_spline, main = "Basis Functions for Training Set")
```



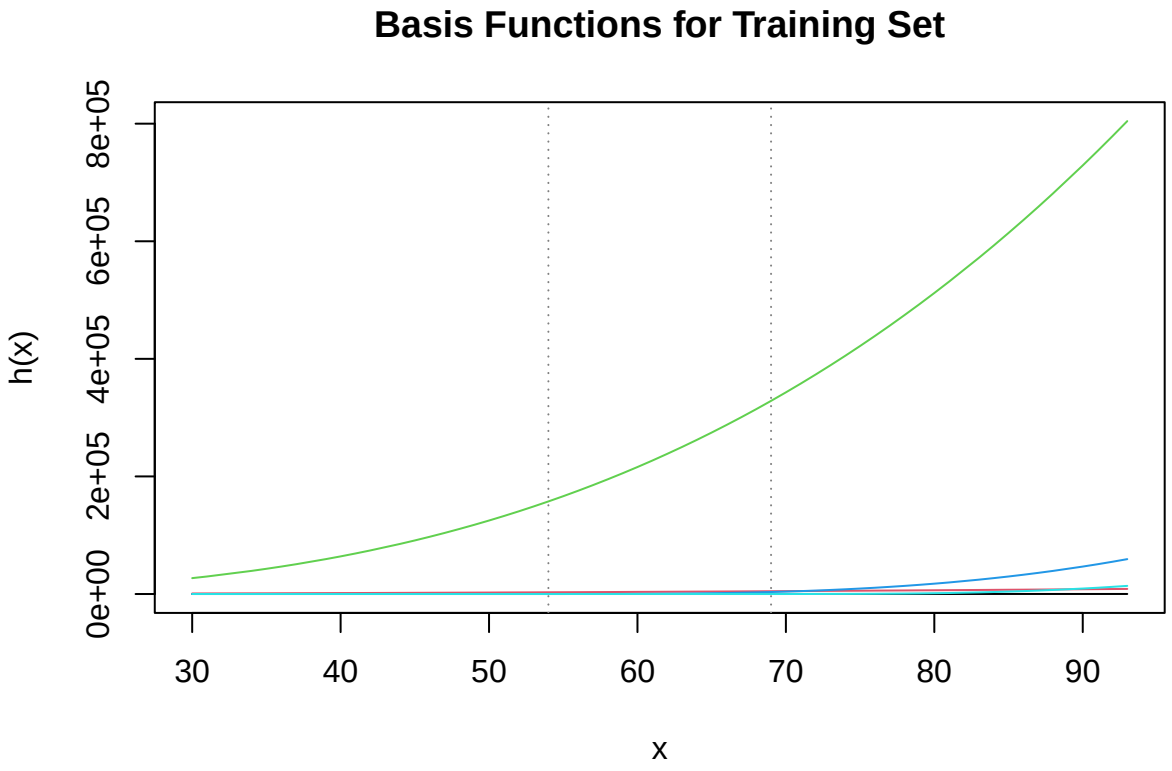

We should see 5 basis functions in this plot because we set M=4M=4 (giving us M−1=3 p) and nknots=2(adding 2 knot-based basis functions). However, the plot appears confusing and cluttered. This happens because we did not sort the "Temp" variable before calculating the basis functions.

If the "Temp" variable is unsorted, the basis functions are trained on an unsorted version of "Temp" but are plotted using the original, differently ordered values. This mismatch causes the basis functions to look irregular and inconsistent.

By sorting the "Temp" variable beforehand, the resulting basis functions will align properly, producing a clear and interpretable plot.

```r
train_data_sorted <- train_data[order(train_data$Temp), ]
train_spline <- lecturespl(train_data_sorted$Temp, nknots=2, M=4)
```

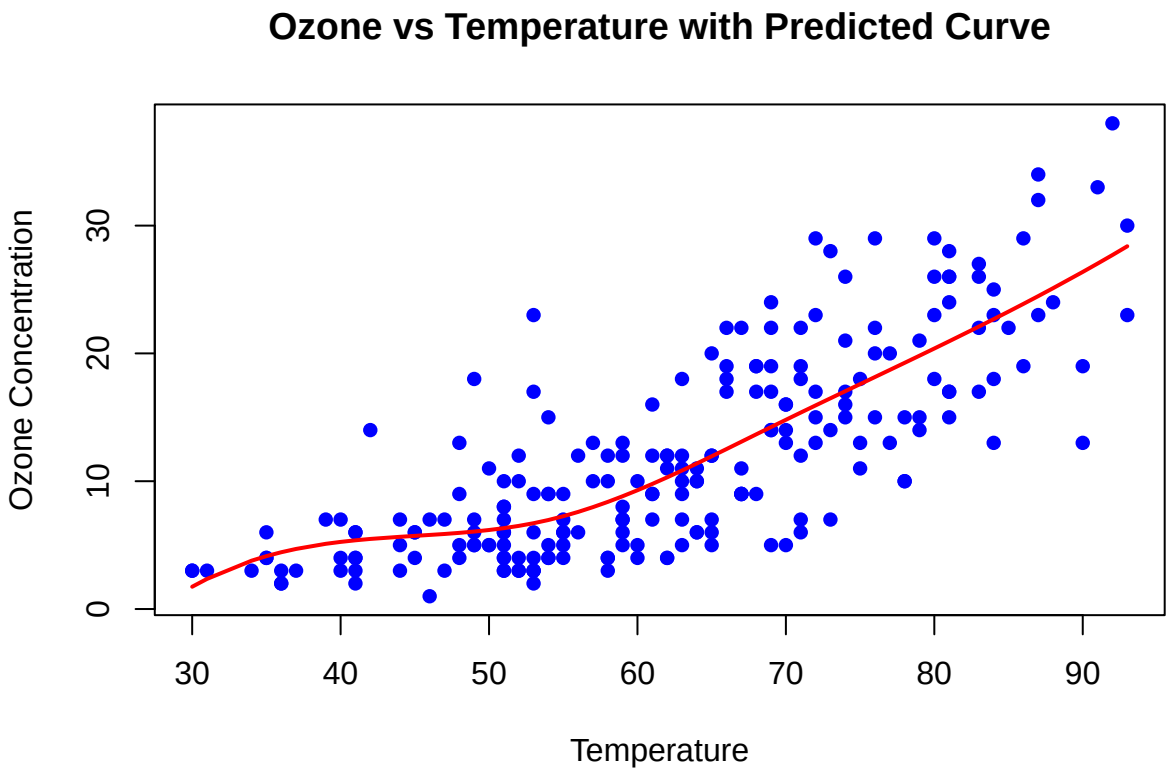
```r
train_spline$x <- train_data_sorted$Temp
plotspl(train_spline, main = "Basis Functions for Training Set")
```




## 2) Predict the training data

Now we calculate the coefficients using the basis functions, which will allow us to make predictions on new data. We provide the response variable (our target data) and the calculated XX-matrix with 5 basis functions as predictors. Our model will estimate 5 coefficients, one for each basis function. These coefficients determine how much each basis function contributes to the final model.

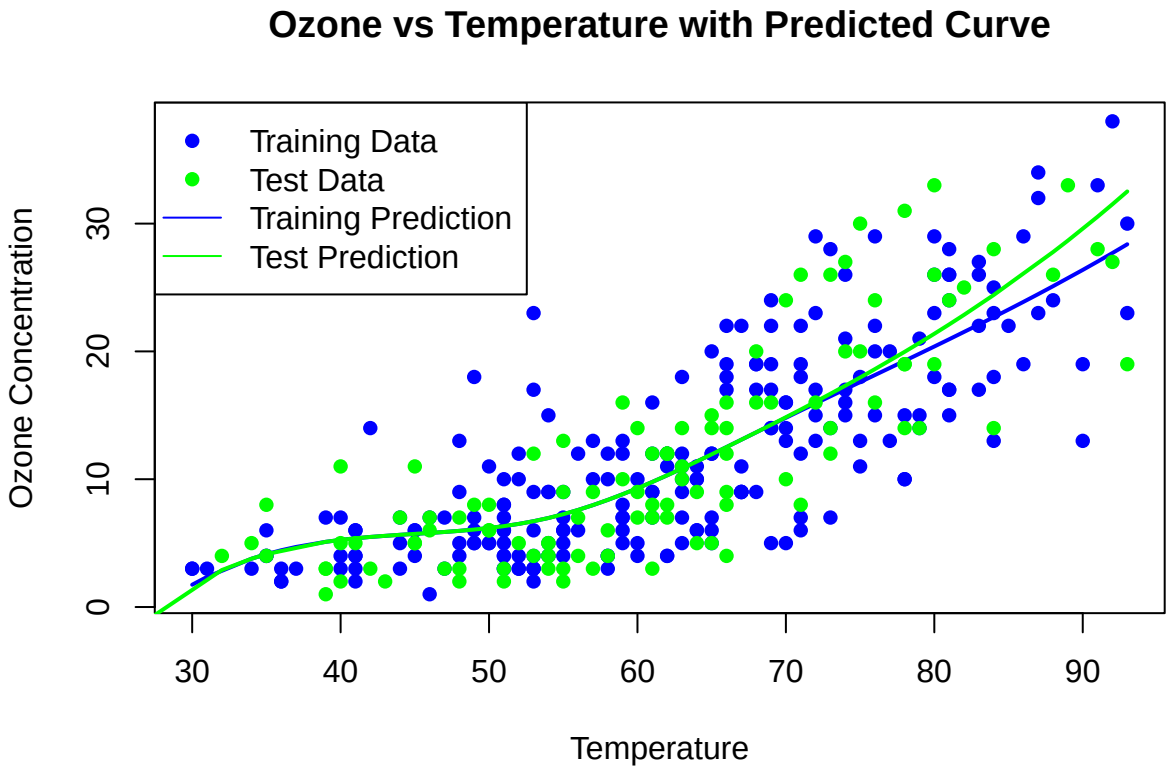
```r
lm_fit <- lm(train_data_sorted$Ozone ~ ., data = data.frame(train_spline$X))

prediction <- predict(lm_fit)

plot(train_data_sorted$Temp, train_data_sorted$Ozone,
     main = "Ozone vs Temperature with Predicted Curve",
     xlab = "Temperature", ylab = "Ozone Concentration",
     pch = 16, col = "blue")
lines(train_data_sorted$Temp, prediction,
      col = "red", lwd = 2)
```




## 3) Predictiction on test set

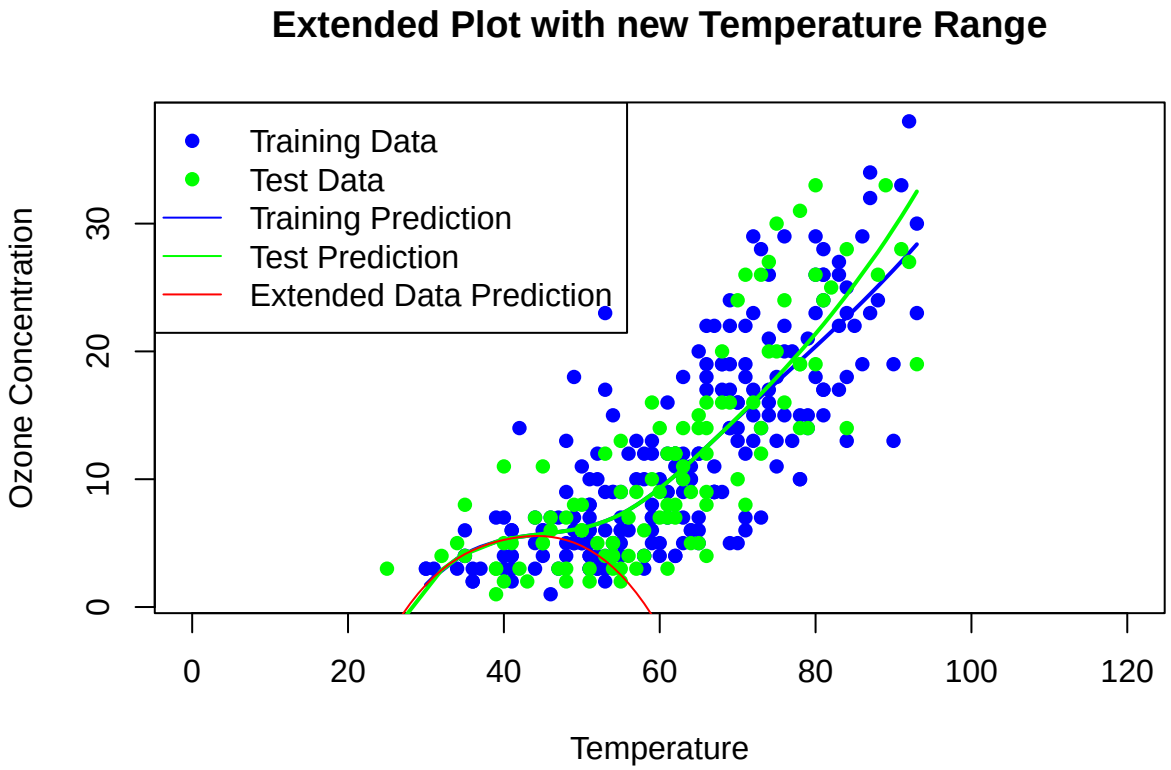
We do the same for the test set. First, we sort the "Temp" variable, then create the basis functions. Finally, we use the trained model to make predictions, but this time with the basis matrix generated from the test data.

```
test_data_sorted <- test_data[order(test_data$Temp), ]
test_spline <- lecturespl(test_data_sorted$Temp, nknots = 2, M = 4)

prediction_test <- predict(lm_fit, newdata = data.frame(test_spline$X))


plot(train_data_sorted$Temp, train_data_sorted$Ozone,
     main = "Ozone vs Temperature with Predicted Curve",
     xlab = "Temperature", ylab = "Ozone Concentration",
     pch = 16, col = "blue")
lines(train_data_sorted$Temp, prediction, col = "blue", lwd = 2)

points(test_data_sorted$Temp, test_data_sorted$Ozone, pch = 16, col = "green")
lines(test_data_sorted$Temp, prediction_test, col = "green", lwd = 2)

legend("topleft", legend = c("Training Data", "Test Data",
                             "Training Prediction", "Test Prediction"),
       col = c("blue", "green", "blue", "green"),
       pch = c(16, 16, NA, NA), lty = c(NA, NA, 1, 1))
```




## 4) Generate new temperature data with seq(0,120)

```
new_temp <- seq(0, 120)

new_temp_spline <- lecturespl(new_temp, nknots = 2, M = 4)

prediction_newTemp <- predict(lm_fit, newdata = data.frame(new_temp_spline$X))


plot(train_data_sorted$Temp, train_data_sorted$Ozone,
     main = "Extended Plot with new Temperature Range",
     xlab = "Temperature", ylab = "Ozone Concentration",
     pch = 16, col = "blue", xlim = range(new_temp))
lines(train_data_sorted$Temp, prediction, col = "blue", lwd = 2)

points(test_data_sorted$Temp, test_data_sorted$Ozone,pch = 16, col = "green")
lines(test_data_sorted$Temp, prediction_test, col = "green", lwd = 2)
lines(new_temp, prediction_newTemp, col = "red")

legend("topleft", legend = c("Training Data", "Test Data",
                             "Training Prediction", "Test Prediction", "Extended Data Prediction"),
       col = c("blue", "green", "blue", "green","red"),
       pch = c(16, 16, NA, NA,NA), lty = c(NA, NA, 1, 1, 1))
```



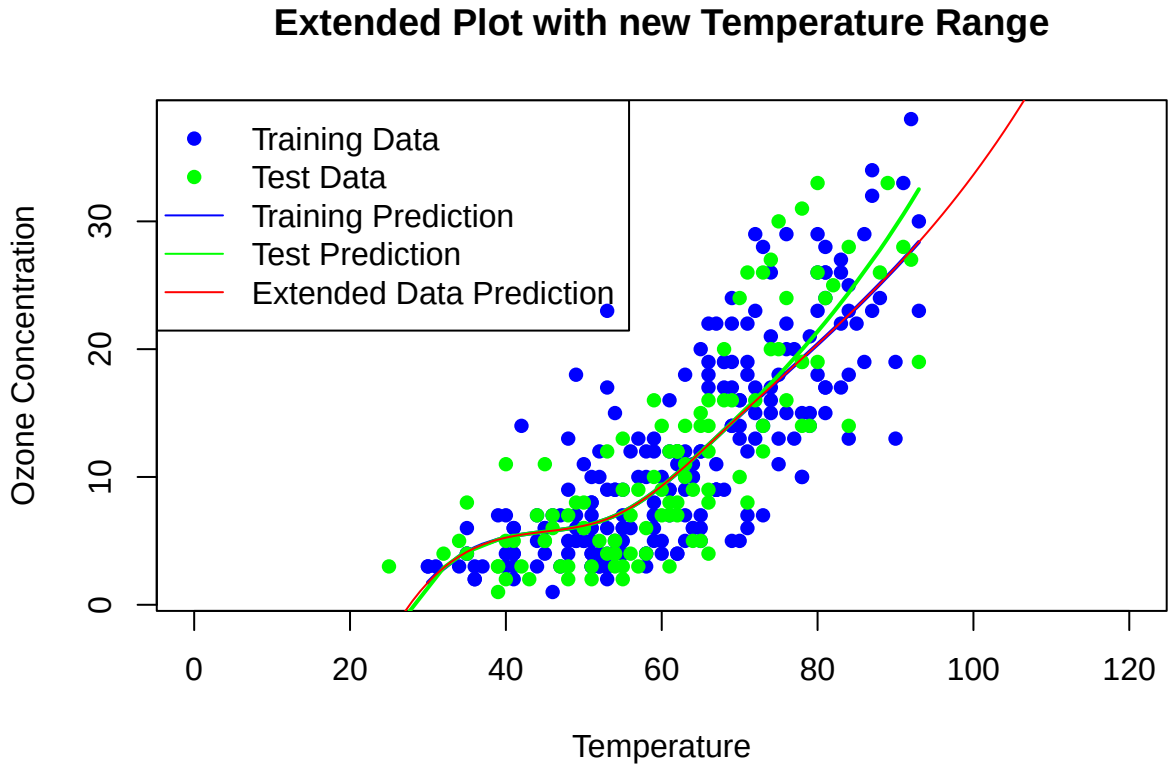

Since the knots are derived from the training data range, the basis functions are only well-defined within this range. When we extend the temperature range beyond the training data (Temp=[0,120]), the spline basis functions behave unpredictably, leading to unreliable predictions.

## 5) Modifing lecturespl()

To address this problem, we need to adjust how the knots are placed in our basis function. Instead of using quantiles of the input variable, we should explicitly define the knots to cover the entire extended range of the temperature (e.g., from 0 to 120). This ensures the basis functions remain well-defined and produce consistent predictions across both the training and extended data range.

```
lecturespl_modified <- function(x_train, x_test, nknots=2, M=4){
  n <- length(x_test)
  X <- matrix(NA,nrow=n,ncol=(M-1)+nknots)
  for (i in 1:(M-1)){ X[,i] <- x_test^i }
    quant <- seq(0,1,1/(nknots+1))[c(2:(nknots+1))]
    qu <- quantile(x_train,quant)
  for (i in M:(M+nknots-1)){
    X[,i] <- ifelse(x_test-qu[i-M+1]<0,0,(x_test-qu[i-M+1])^(M-1))
    }
  list(X=X,quantiles=quant,xquantiles=qu)
}

new_temp_spline <- lecturespl_modified(train_data_sorted$Temp,new_temp, nknots = 2, M = 4)
prediction_newTemp <- predict(lm_fit, newdata = data.frame(new_temp_spline$X))


plot(train_data_sorted$Temp, train_data_sorted$Ozone,
```

```
     main = "Extended Plot with new Temperature Range",
     xlab = "Temperature", ylab = "Ozone Concentration",
     pch = 16, col = "blue", xlim = range(new_temp))
lines(train_data_sorted$Temp, prediction, col = "blue", lwd = 2)

points(test_data_sorted$Temp, test_data_sorted$Ozone,pch = 16, col = "green")
lines(test_data_sorted$Temp, prediction_test, col = "green", lwd = 2)
lines(new_temp, prediction_newTemp, col = "red")

legend("topleft", legend = c("Training Data", "Test Data",
                             "Training Prediction", "Test Prediction", "Extended Data Prediction"),
       col = c("blue", "green", "blue", "green","red"),
       pch = c(16, 16, NA, NA,NA), lty = c(NA, NA, 1, 1, 1))
```




## 6) Log Transformation

Since negative ozone concentrations are unrealistic, we transformed the response variable using a logarithmic transformation during the lm() fitting process. By modeling the logarithm of ozone concentrations, we ensure the predictions remain positive when we reverse-transform them using the exponential function (exp()).

```
log_model <- lm(log(train_data_sorted$Ozone) ~ ., data = data.frame(train_spline$X))

prediciton <- exp(predict(log_model, data.frame(train_spline$X)))
prediciton_test <- exp(predict(log_model, data.frame(test_spline$X)))
prediciton_extTemp <- exp(predict(log_model, data.frame(new_temp_spline$X)))

plot(ozone$Temp, ozone$Ozone,
```

```
     main = "Log Models Plot",
     xlab = "Temperature", ylab = "Ozone Concentration",
     pch = 16, col = "blue", xlim = range(new_temp))
lines(train_data_sorted$Temp, prediciton,col = "red", lwd = 4)
lines(test_data_sorted$Temp, prediciton_test,col = "green", lwd = 2)
lines(new_temp, prediciton_extTemp,col = "black", lwd = 2)

legend("topleft", legend = c("Data",
                             "Training Prediction", "Test Prediction", "Extended Data Prediction"),
       col = c("blue", "red", "green", "black"),
       pch = c(16, NA, NA, NA,NA), lty = c(NA, 1, 1, 1, 1) )
```