BS755 / MA575 Lab 6: Multiple Linear Regression

Friday, October 13, 2023

Contents

Data Processing

This Rmarkdown document presents the analysis of multiple linear regression on air quality data.

Set the working directory

Load the required packages

```
# Load required packages if not already installed
library(ggplot2)
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(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

Reading and cleaning the data

```
# Read data from a CSV file using semicolon as delimiter
# Use '..' to move up one directory level to access the 'Lab 03' directory ...
# ... and read the 'AirQualityData.csv' file.
aq_data <- read.csv("../Lab 03/AirQualityData.csv", header=TRUE, as.is=TRUE, sep=';')

# In this data set missing values are stored as -200
# Replace missing data (-200) with NA
```

```r
aq_data[aq_data == -200] <- NA
# Note that this replaces -200 to NA across all columns in aq_data

# Create a new data frame 'tempdataset' by selecting specific columns from 'aq_data'
# For 'Time' column: extract the numeric components from a ...
# ... character string in a specific format and convert it into a numeric value.
tempdataset <- data.frame(Time = as.numeric(sub("^(\\d+).(\\d+).*", "\\1.\\2",
                                                aq_data$Time)),
                          Temperature = aq_data$Temperature,
                          AbsoluteHumidity = aq_data$AbsoluteHumidity,
                          RelativeHumidity = aq_data$RelativeHumidity,
                          SensorCO = aq_data$PT08.S1.CO.,
                          GroundCO = aq_data$CO.GT.)

# Remove rows with missing values (NA)
Dataset <- tempdataset[complete.cases(tempdataset), ]

# Remove the 'tempdataset' data frame from the R environment
rm(tempdataset)
```

Model Building

Model Specifications

Ordinary least squares model

```r
# Ordinary LS
m.ols <- lm(GroundCO ~ SensorCO, data = Dataset)
```

Quadratic least squares model

```r
# Quadratic LS
m.quadls <- lm(GroundCO ~ SensorCO + I(SensorCO^2), data = Dataset)
```

Quartic least squares model

```r
# Quartic LS
m.quartls <- lm(GroundCO ~ SensorCO + I(SensorCO^2) +
                  I(SensorCO^3) + I(SensorCO^4), data = Dataset)
```

Model Evaluation

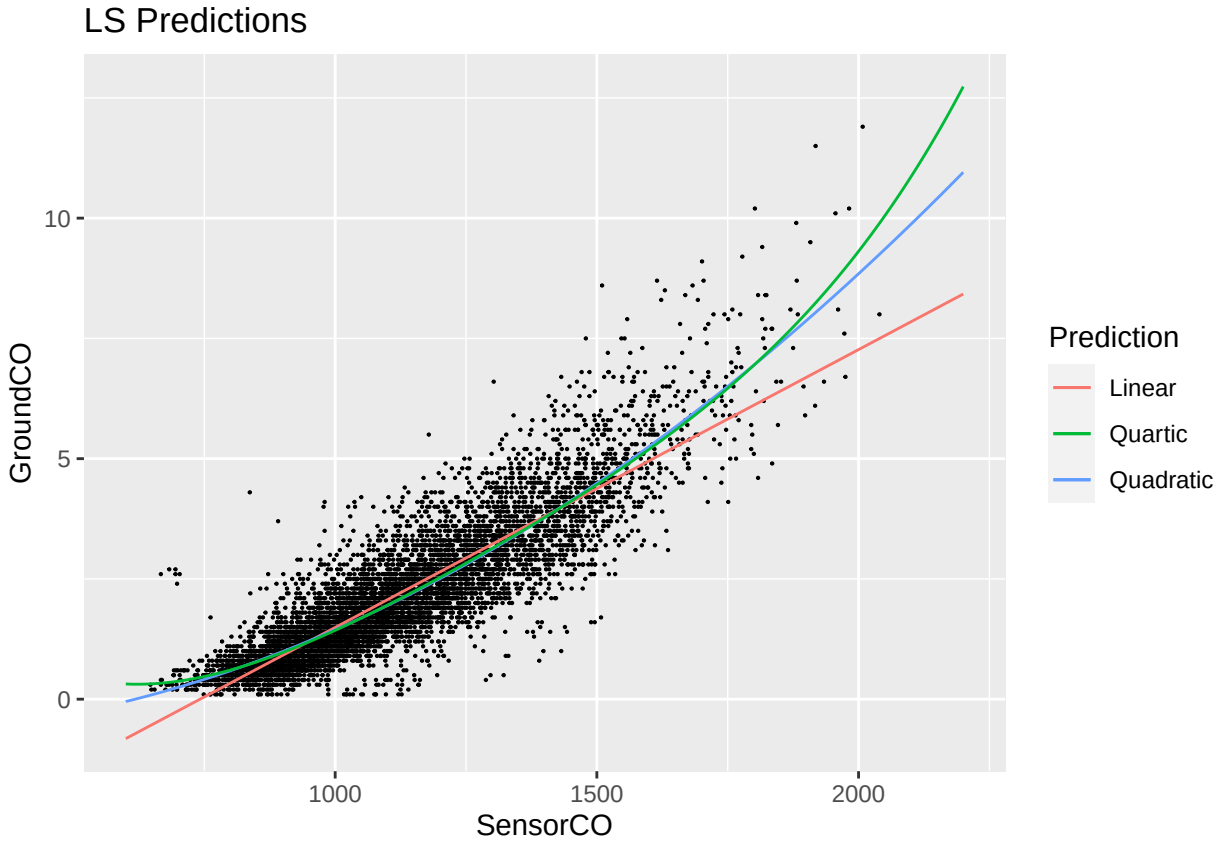
Fitted values

```r
# Predictions/fitted-values for all three models
SensorCONew <- seq(600, 2200, len = length(Dataset$SensorCO))
predLinear <- predict(m.ols, newdata = data.frame(SensorCO = SensorCONew))
predQuad <- predict(m.quadls, newdata = data.frame(SensorCO = SensorCONew))
predQuart <- predict(m.quartls, newdata = data.frame(SensorCO = SensorCONew))
```

Fitted values plot

```r
# Fitted values plot
# Add a scatterplot layer to the plot with specified point size (0.1)
# Add a line plot layer to the plot using the predLinear values, coloring them blue
# Add second line plot layer using the predQuad values coloring them red
```

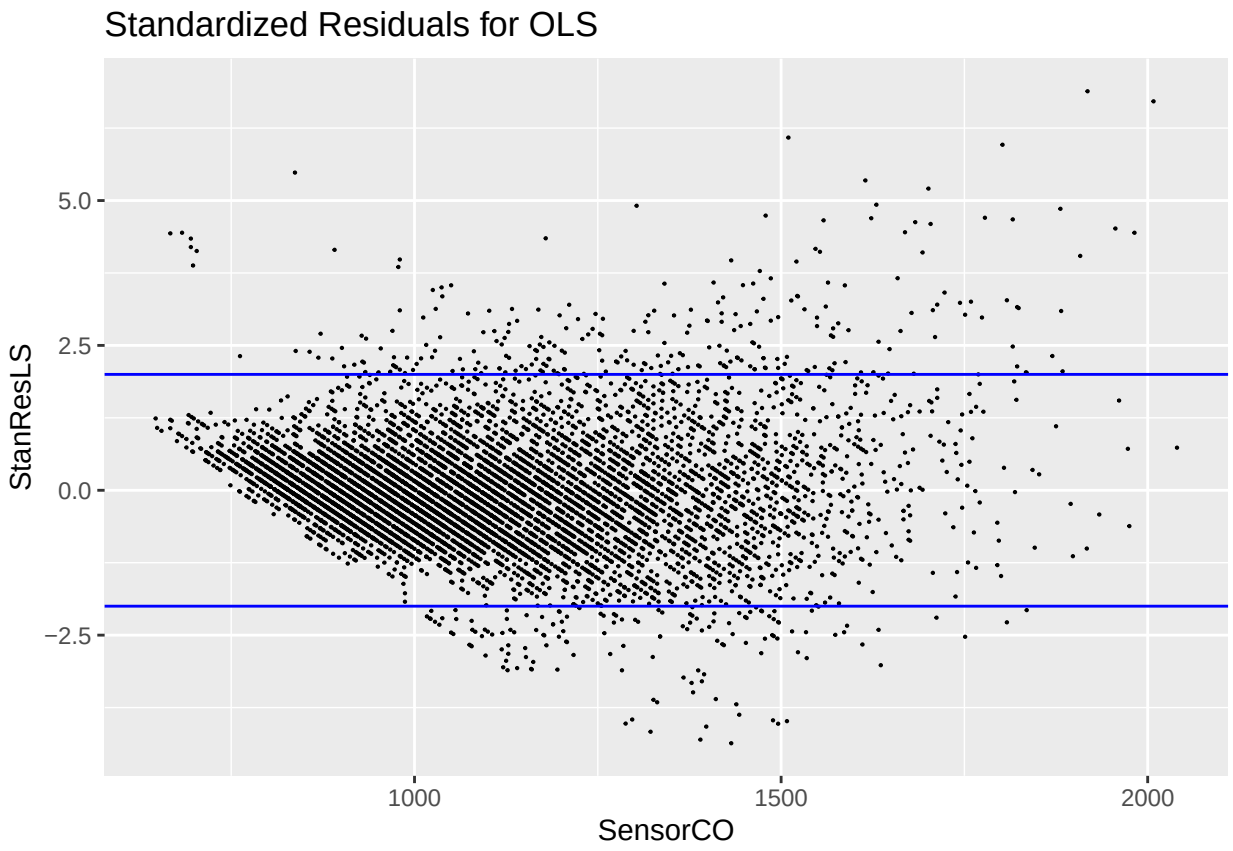
```
# Add third  line plot layer using the predQuart values coloring them green
# Set the color scale for the lines with labels "Linear", "Quartic", ...
# ... and "Quadratic" and name it "Prediction" in the legend
# Set the title of the plot to "LS Predictions"
ggplot(Dataset, aes(x = SensorCO, y = GroundCO)) +
  geom_point(size = 0.1) +
  geom_line(mapping = aes(x = SensorCONew, y = predLinear, color = "blue")) +
  geom_line(mapping = aes(x = SensorCONew, y = predQuad, color = "red")) +
  geom_line(mapping = aes(x = SensorCONew, y = predQuart, color = "green")) +
  scale_color_discrete(name = "Prediction", labels = c("Linear", "Quartic", "Quadratic")) +
  ggtitle("LS Predictions")
```



Standardized residuals for OLS

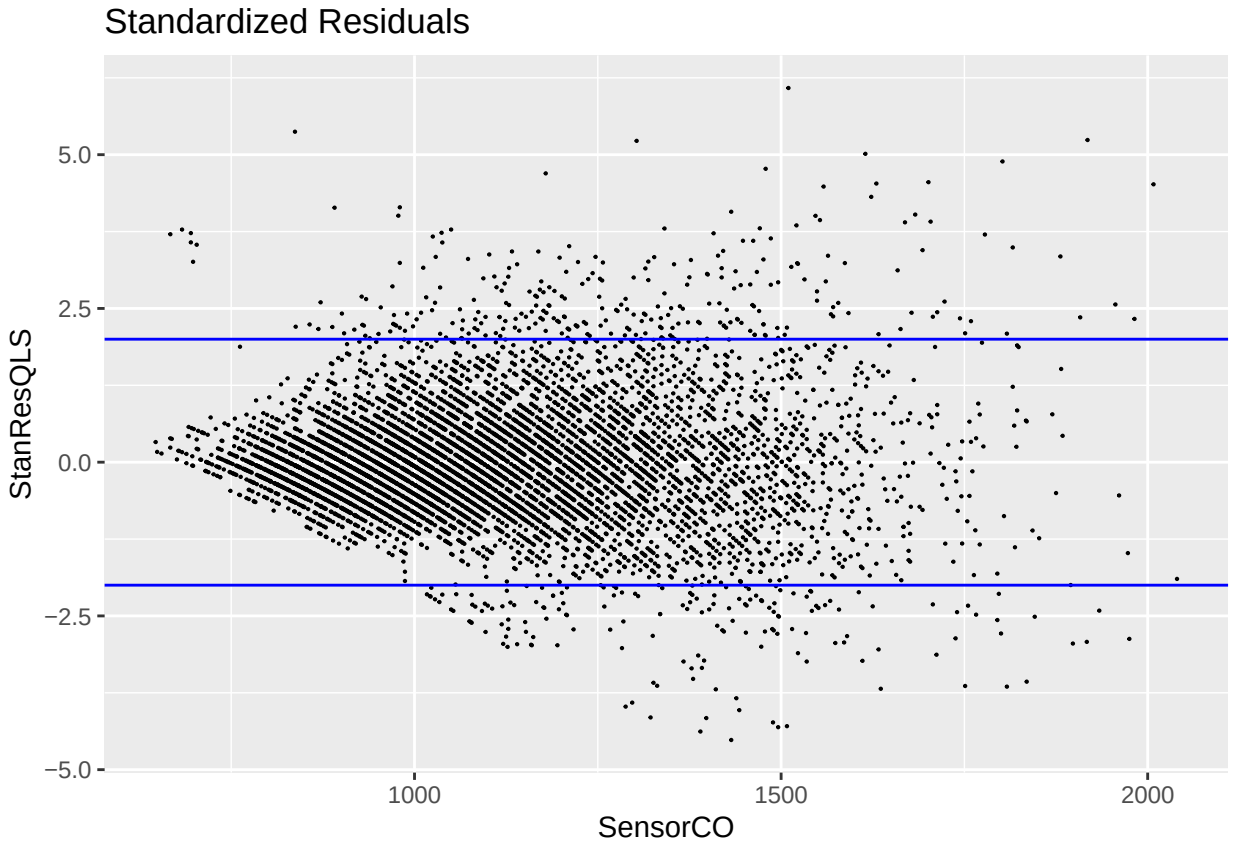
```
# Standardized Residual, Ordinary LS Standard Residual vs SensorCO
# Calculate the standardized residuals for the OLS model
StanResLS <- rstandard(m.ols)
# Create a new data frame containing two columns: SensorCO and StanResLS
dataLS <- data.frame(SensorCO = Dataset$SensorCO, StanResLS)
# Create a scatterplot using the data in dataLS, with SensorCO ...
# ... on the x-axis and StanResLS on the y-axis
# Set the point size to 0.1 for better visibility
# Add horizontal lines at y = 2 and y = -2 in blue to indicate ...
# ... the threshold for standardized residuals
# Set the title of the plot to "Standardized Residuals for OLS"
ggplot(dataLS, aes(x = SensorCO, y = StanResLS)) +
  geom_point(size = 0.1) +
  geom_hline(yintercept = 2, color = 'blue') +
```

```
  geom_hline(yintercept = -2, color = 'blue') +
  ggtitle("Standardized Residuals for OLS")
```



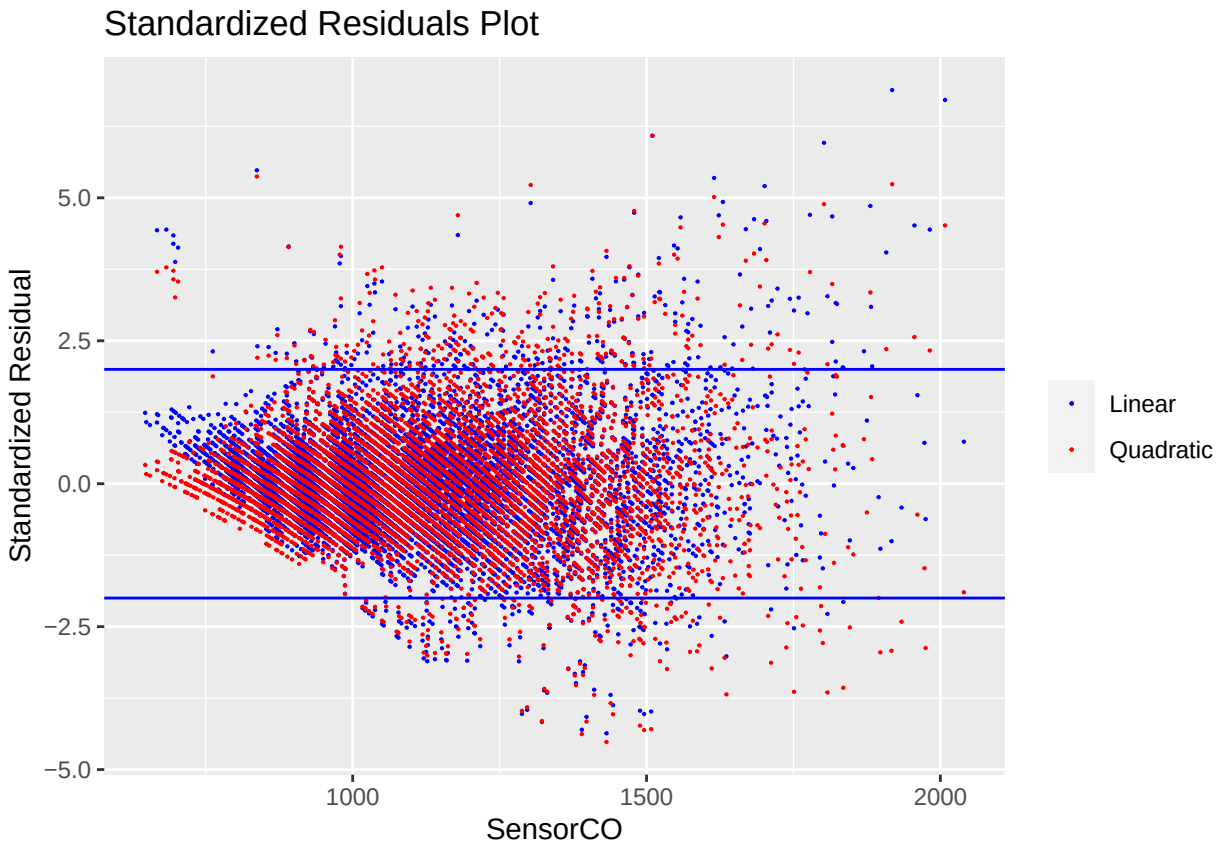
Standardized residuals for Quadratic LS

```
# Quadratic LS Standard Residual vs SensorCO
# Calculate standardized residuals for the quadratic linear model (m.quadls)
StanResQLS <- rstandard(m.quadls)
# Create a data frame 'dataQLS' with two columns: 'SensorCO' and 'StanResQLS'
dataQLS <- data.frame(SensorCO = Dataset$SensorCO, StanResQLS)
# Create a ggplot scatterplot with 'SensorCO' on the x-axis and 'StanResQLS' on the y-axis
# Add points to the plot with a specified point size (0.1)
# Add horizontal lines at y = 2 and y = -2 in blue to indicate threshold lines
# Set the title of the plot to "Standardized Residuals"
ggplot(dataQLS, aes(x = SensorCO, y = StanResQLS)) +
  geom_point(size = 0.1) +
  geom_hline(yintercept = 2, color = 'blue') +
  geom_hline(yintercept = -2, color = 'blue') +
  ggtitle("Standardized Residuals")
```



Combined standardized residuals plot

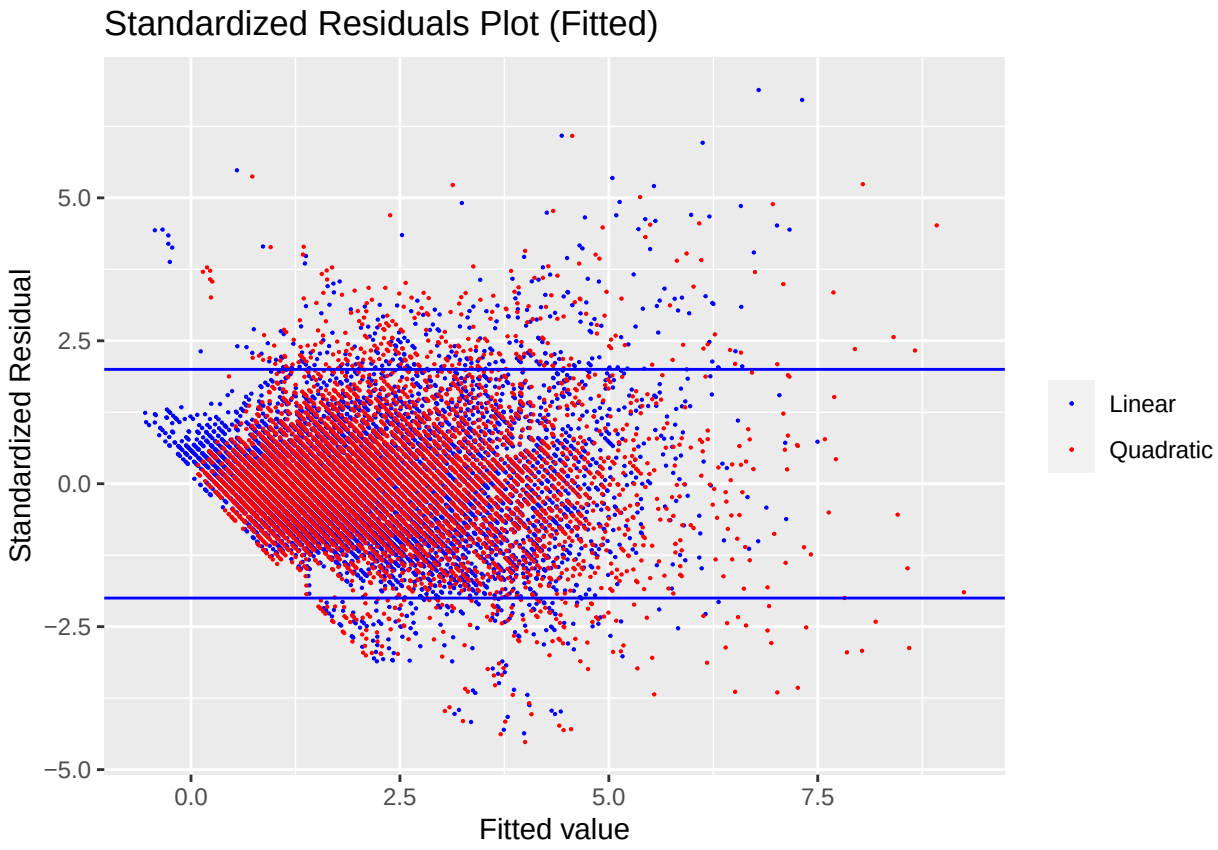
```
# Combine LS and Quadratic Stan. Res. plot in the same
# Create a ggplot scatterplot with 'SensorCO' on the x-axis and 'StanResQLS' ...
# ... as well as 'StanResQLS' on the y-axis
# Add points to the plot with a specified point size (0.1)
# Add horizontal lines at y = 2 and y = -2 in blue to indicate threshold lines
# Set a manual color scale for the points with labels "Linear" and "Quadratic"
# Add a label for the y-axis
# Set the title of the plot to "Standardized Residuals"
ggplot() +
  geom_point(data = dataLS, aes(x = SensorCO, y = StanResLS, color = "Linear"),
             size = 0.1) +
  geom_point(data = dataQLS, aes(x = SensorCO, y = StanResQLS, color = "Quadratic"),
             size = 0.1) +
  geom_hline(yintercept = 2, color = 'blue') +
  geom_hline(yintercept = -2, color = 'blue') +
  scale_color_manual(name = element_blank(), labels = c("Linear", "Quadratic"),
                     values = c("blue", "red")) +
  labs(y = "Standardized Residual") +
  ggtitle("Standardized Residuals Plot")
```



Standardized residuals vs fitted for OLS and Quadratic LS

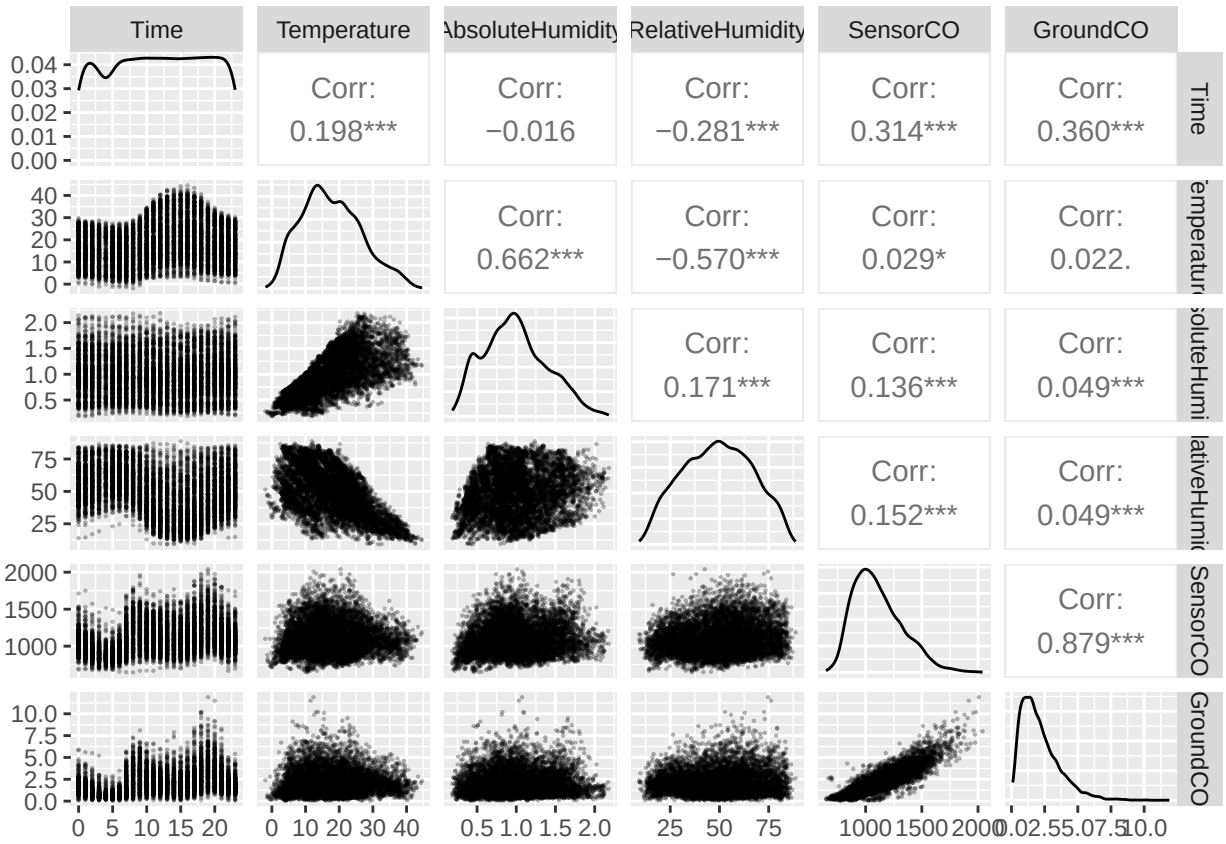
```
# Standardized Residuals vs Fitted
# Calculate the fitted values for the OLS model
Fitted <- fitted(m.ols)
# Create a data frame containing the fitted values and the standardized residuals
dataLSFitted <- data.frame(Fitted, StanResLS)
# Calculate the fitted values for the QuadLS model
Fitted <- fitted(m.quadls)
# Create a data frame containing the fitted values and the standardized residuals
dataQLSFitted <- data.frame(Fitted, StanResQLS)

# Combine OLS and Quadratic LS standardized residuals vs fitted values plot
# Create a ggplot scatterplot with fitted values on the x-axis and 'StanResQLS' ...
# ... as well as 'StanResQLS' on the y-axis
# Add points to the plot with a specified point size (0.1)
# Add horizontal lines at y = 2 and y = -2 in blue to indicate threshold lines
# Set a manual color scale for the points with labels "Linear" and "Quadratic"
# Add a label for the y-axis
# Set the title of the plot to "Standardized Residuals"
ggplot() +
  geom_point(data = dataLSFitted, aes(x = Fitted, y = StanResLS, color = "Linear"),
             size = 0.1) +
  geom_point(data = dataQLSFitted, aes(x = Fitted, y = StanResQLS, color = "Quadratic"),
             size = 0.1) +
  geom_hline(yintercept = 2, color = 'blue') +
  geom_hline(yintercept = -2, color = 'blue') +
  scale_color_manual(name = element_blank(), labels = c("Linear", "Quadratic"),
```

```
                    values = c("blue", "red")) +
  labs(y = "Standardized Residual") +
  labs(x = "Fitted value") +
  ggtitle("Standardized Residuals Plot (Fitted) ")
```



Scatterplot matrix with correlation values

```
# Scatterplot Matrix with Correlation Values
# Create a scatterplot matrix (pairs plot) for the 'Dataset' data frame
# The 'upper' panel contains correlation values with a point size of 4
# This visualization helps to understand variable relationships and correlations
# The 'lower' panel contains scatterplots with points and alpha transparency of 0.3
# The point size in the upper panel is set to 0.1
ggpairs(Dataset,
        upper = list(continuous = wrap('cor', size = 4)),
        lower = list(continuous = wrap("points", alpha = 0.3, size = 0.1)))
```



Multiple Linear Regression (MLR)

Building a multiple linear regression model

```
# Create a Multiple Linear Regression Model
# Model 'GroundCO' variable using multiple predictors including ...
# ... 'SensorCO', 'Temperature', 'RelativeHumidity', 'AbsoluteHumidity', ...
# ... and 'Time' from the 'Dataset' data frame.
m.mlr <- lm(GroundCO ~ SensorCO + Temperature + RelativeHumidity +
              AbsoluteHumidity + Time,
            data = Dataset)
```

Summary of the MLR model

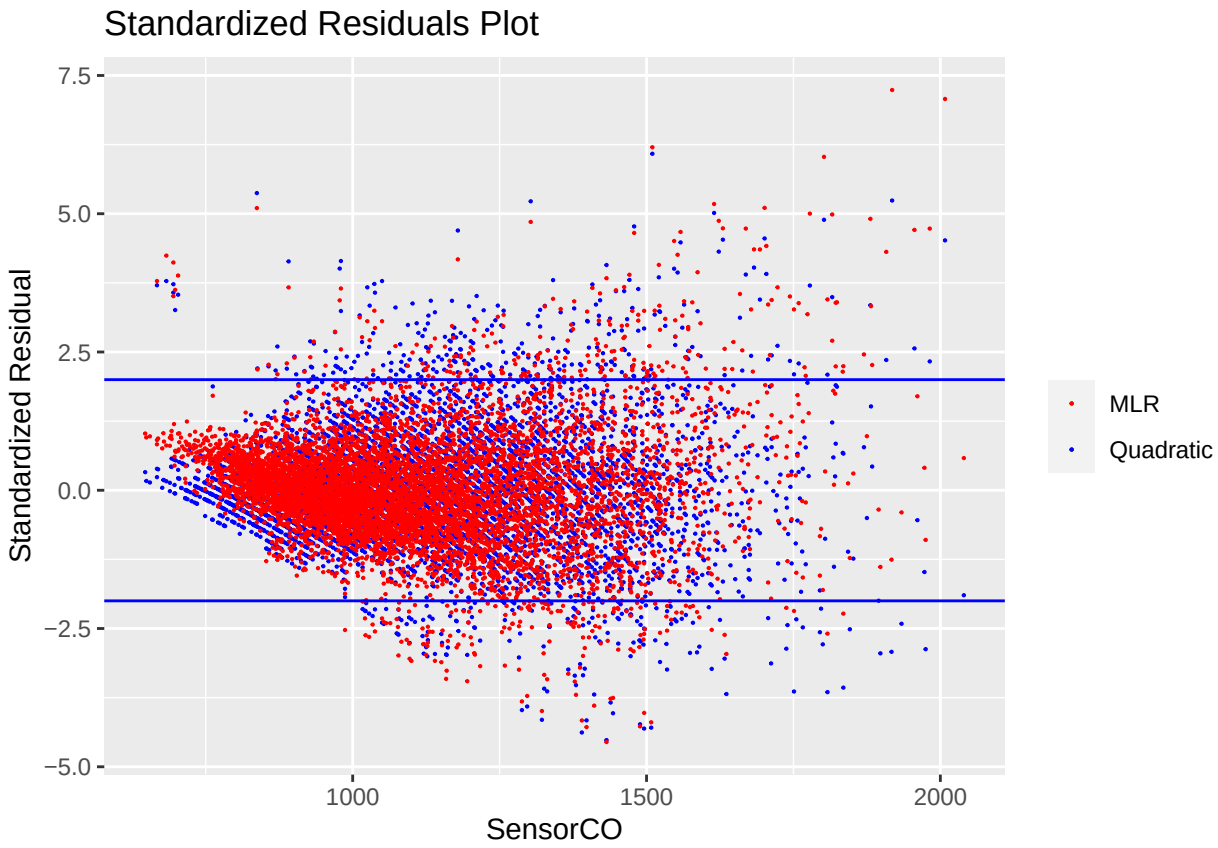
```
# Generate a summary of the Multiple Linear Regression (MLR) model
summary(m.mlr)
```

```
##
## Call:
## lm(formula = GroundCO ~ SensorCO + Temperature + RelativeHumidity +
##     AbsoluteHumidity + Time, data = Dataset)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.0062 -0.3584 -0.0200  0.3234  4.7756
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)      -3.6957392  0.0687179 -53.781  < 2e-16 ***
## SensorCO          0.0057615  0.0000389 148.099  < 2e-16 ***
## Temperature      -0.0169546  0.0030664  -5.529 3.33e-08 ***
## RelativeHumidity -0.0105606  0.0012023  -8.784  < 2e-16 ***
## AbsoluteHumidity  0.0783070  0.0564520   1.387    0.165
## Time              0.0144227  0.0012553  11.489  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6606 on 7338 degrees of freedom
## Multiple R-squared:  0.7886, Adjusted R-squared:  0.7885
## F-statistic:  5476 on 5 and 7338 DF,  p-value: < 2.2e-16
```

Standard residuals for MLR

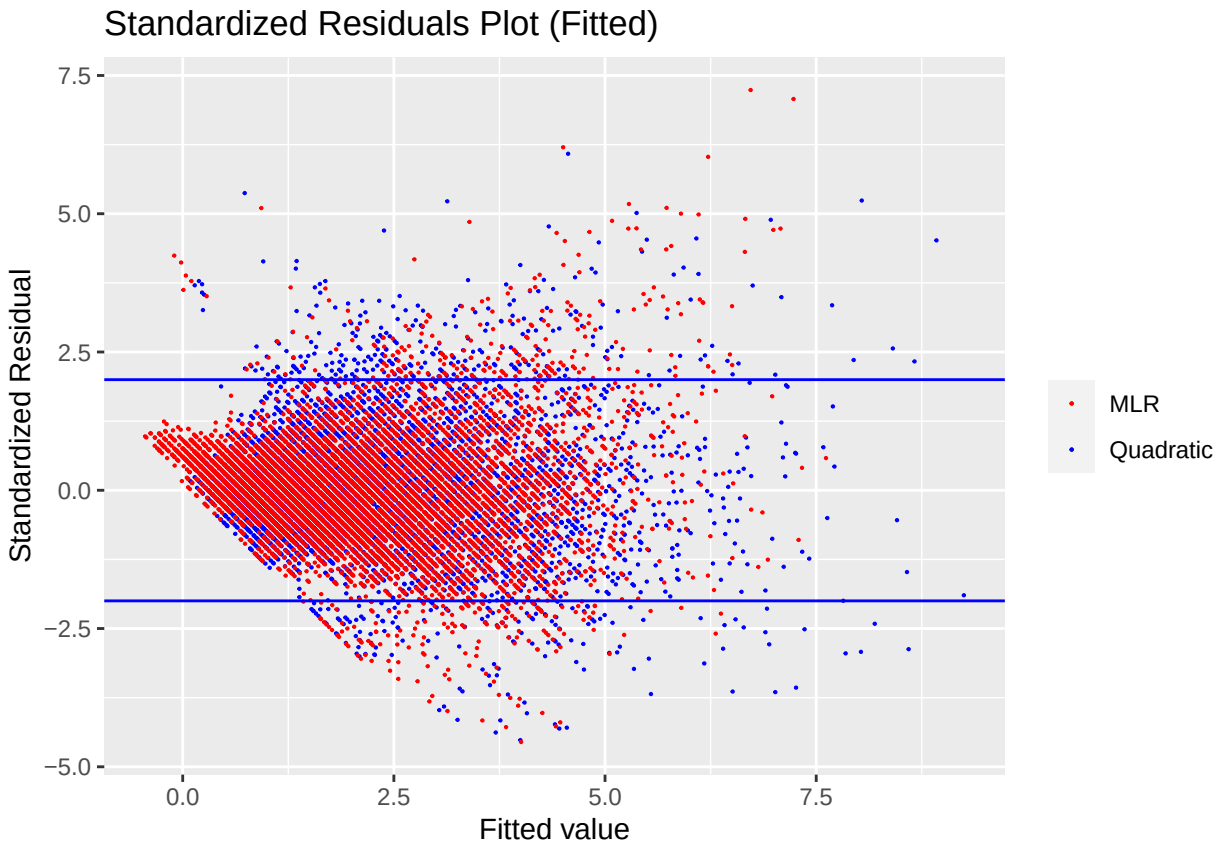
```r
# Standard Residuals vs SensorCO
# Examine Standardized residuals
# 'rstandard' calculates the standardized residuals for a linear regression model
StanResMLR <- rstandard(m.mlr)
# Create a data frame containing the SensorCO values and the standardized residuals
dataMLS <- data.frame(SensorCO = Dataset$SensorCO, StanResMLR)

# Combine Quad LS and MLR standardized residuals vs SensorCO plot
# Create a ggplot scatterplot with SensorCO on the x-axis and 'StanResQLS' ...
# ... as well as 'StanResMLR' on the y-axis
# Add points to the plot with a specified point size (0.1)
# Add horizontal lines at y = 2 and y = -2 in blue to indicate threshold lines
# Set a manual color scale for the points with labels "Linear" and "Quadratic"
# Add a label for the y-axis
# Set the title of the plot to "Standardized Residuals"
ggplot() +
  geom_point(data = dataQLS, aes(x = SensorCO, y = StanResQLS, color = "Quadratic"),
             size = 0.1) +
  geom_point(data = dataMLS, aes(x = SensorCO, y = StanResMLR, color = "MLR"),
             size = 0.1) +
  geom_hline(yintercept = 2, color = 'blue') +
  geom_hline(yintercept = -2, color = 'blue') +
  scale_color_manual(name = element_blank(), labels = c("MLR", "Quadratic"),
                     values = c("red", "blue")) +
  labs(y = "Standardized Residual") +
  ggtitle("Standardized Residuals Plot")
```



Standardized residuals vs fitted for MLR and Quadratic LS

```
# Standardized Residuals vs Fitted
# Calculate the fitted values for the Quad LS model
Fitted <- fitted(m.quadls)
# Create a data frame containing the fitted values and the standardized residuals
dataQLSFitted <- data.frame(Fitted, StanResQLS)
# Calculate the fitted values for the MLR model
Fitted <- fitted(m.mlr)
# Create a data frame containing the fitted values and the standardized residuals
dataMLRFitted <- data.frame(Fitted, StanResMLR)

# Combine Quadratic and MLR standardized residuals vs fitted values plot
# Create a ggplot scatterplot with fitted values on the x-axis and 'StanResQLS' ...
# ... as well as 'StanResMLR' on the y-axis
# Add points to the plot with a specified point size (0.1)
# Add horizontal lines at y = 2 and y = -2 in blue to indicate threshold lines
# Set a manual color scale for the points with labels "MLR" and "Quadratic"
# Add a label for the y-axis
# Set the title of the plot to "Standardized Residuals"
ggplot() +
  geom_point(data = dataQLSFitted, aes(x = Fitted, y = StanResQLS, color = "Quadratic"),
             size = 0.1) +
  geom_point(data = dataMLRFitted, aes(x = Fitted, y = StanResMLR, color = "MLR"),
             size = 0.1) +
  geom_hline(yintercept = 2, color = 'blue') +
  geom_hline(yintercept = -2, color = 'blue') +
  scale_color_manual(name = element_blank(), labels = c("MLR", "Quadratic"),
```

```
                    values = c("red", "blue")) +
  labs(y = "Standardized Residual") +
  labs(x = "Fitted value") +
  ggtitle("Standardized Residuals Plot (Fitted) ")
```



Comparing Models

Comparison with Quadratic LS

Building a quadratic MLR model

```
# Create a Multiple Linear Regression Model
# Model 'GroundCO' variable using multiple predictors including ...
# ... 'SensorCO', 'SensorCO^2', 'Temperature', 'RelativeHumidity', 'AbsoluteHumidity', ...
# ... and 'Time' from the 'Dataset' data frame.
mquad.mls <- lm(GroundCO ~ SensorCO + I(SensorCO^2) + Temperature + RelativeHumidity +
                  AbsoluteHumidity + Time,
                data = Dataset)
```

Summary of the quadratic MLR model

```
# Generate a summary of the mquad.mls model
summary(mquad.mls)
```

```
##
## Call:
## lm(formula = GroundCO ~ SensorCO + I(SensorCO^2) + Temperature +
##     RelativeHumidity + AbsoluteHumidity + Time, data = Dataset)
##
```

```
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.9960 -0.3231 -0.0335  0.2907  3.9478
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -6.960e-02  1.691e-01  -0.412  0.68069
## SensorCO         -9.176e-04  2.890e-04  -3.175  0.00151 **
## I(SensorCO^2)     2.782e-06  1.194e-07  23.308  < 2e-16 ***
## Temperature      -9.149e-03  2.978e-03  -3.072  0.00213 **
## RelativeHumidity -7.673e-03  1.167e-03  -6.576 5.16e-11 ***
## AbsoluteHumidity -2.021e-02  5.464e-02  -0.370  0.71147
## Time              1.820e-02  1.222e-03  14.893  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6375 on 7337 degrees of freedom
## Multiple R-squared:  0.8032, Adjusted R-squared:  0.8031
## F-statistic:  4991 on 6 and 7337 DF,  p-value: < 2.2e-16
```

Compare with quadratic model

```
# Compare with the summary of m.quadls model
summary(m.quadls)
```

```
##
## Call:
## lm(formula = GroundCO ~ SensorCO + I(SensorCO^2), data = Dataset)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.9977 -0.3653 -0.0478  0.2939  4.0370
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -7.159e-01  1.709e-01  -4.189 2.84e-05 ***
## SensorCO      -4.603e-04  2.925e-04  -1.574    0.116
## I(SensorCO^2)  2.620e-06  1.220e-07  21.482  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6637 on 7341 degrees of freedom
## Multiple R-squared:  0.7866, Adjusted R-squared:  0.7865
## F-statistic: 1.353e+04 on 2 and 7341 DF,  p-value: < 2.2e-16
```