Use of Linear Regression Modeling To Predict Mountain Pine Beetle Correlation to Carcinogen Levels in Water

Russ Steil

12 December 2022

Abstract

Given the mountain pine beetle data set, this project was conducted to try and construct a linear model for dissolved organic carbon found within the affected pine needles as a result of the mountain pine needle. To construct the linear model, several methods were used. These methods were the backward stepwise BIC method, the LASSO method, a correlation method, the RIDGE method, and the LOESS method. The model selected out of these methods was the correlation model. This model had an R^2 of ~0.52, an RMSE of ~2.25, and an MAE of ~1.34.

Introduction

The mountain pine beetle is an invasive species of beetle typically found in the Rocky Mountains and other western mountain regions of the United States. These beetles cause massive destruction to the pine trees that are found in these areas by infesting and killing the trees. As these trees die, the needles start to change color from green, to red, to a brownish/grayish color. Once the trees have been killed the needles fall to the forest floor where they decay. As the needles decay, they release dissolved organic carbon (DOC). This is where the problem starts to appear because when these needles that release DOC get swept up into streams, they eventually end up in the water supply. When this water supply that has been saturated with DOC gets treated with chlorine, carcinogens are formed. This is a massive problem that is caused by these mountain pine beetles. Hence the need to model dissolved organic carbon.

Methods

Cleaning and Standardizing Data

The first step that needed to be taken in this project was cleaning and standardizing the data from the data frame that was given. To do this, the data was loaded, then the response variable, meanDOC, was saved into a seperate variable. Then a new predictor called the aspect ratio was created by expanding a column from the original data set into 8 seperate columns. Then all the predictors from the original data set and the 8 new aspect ratio columns were combined into a new data frame. Then all the variables from this data frame were standardized and put into a new standardized data frame.

See Appendix `A1` for code

Splitting into Train and Test Data

The next step was to split all the data into train and test sets. To do this, the first half of both the response variable and the standardized data frame was designated as the training data. The latter half was designated as the testing data.

```
A2
```

Building a Correlation Model

The first model that was tested was a correlation model. To build this model the correlation between all of the predictors and the response variable was checked. If the correlation was higher than plus or minus 0.5, the predictor was kept. Then the correlation between all of the remaining predictors was checked against one another. If the correlation was high between the predictors, then one was dropped. The predictors that remained were then used to construct a model with the meanDOC values from the training set. The final part of this process was making predictions against the testing set using this model.

```
A3
```

```
## Warning in cor(y_response_train, dataFrame_X_standardized_train): the standard
## deviation is zero
```

Building a Backwards Stepwise Model

The next model that was tested was a backwards stepwise model. To create this model a linear model containing the response variable modeled against all the predictors was made. This was put into the backwards selection function with the BIC parameter. After this model was created, it was used to make predictions on the testing set.

```
A4
```

Building a LASSO Model

The next model that was made was a LASSO model. To do this the predictors training data was turned into a matrix and saved as x. Then the response training data was saved as y. Then the LASSO model was created with these x and y parameters. This model was then used to predict both the training and testing data of the predictors.

```
A5
```

Building a Ridge Model

The next model created was a Ridge model. The same x and y from the LASSO model were used to create the Ridge model. Then the Ridge model was used to make predictions on the test data set of the predictors.

```
A6
```

Building a LOESS Model

The final model that was constructed was a LOESS model. To do this, the predictor matrix x was filtered down so that it only included the 4 best variables from the backwards selection model.Then the LOESS model was created using this filtered down predictor matrix and the response variable. This model was then used to make predictions for the predictors test set.

```
A7
```

Results

Calculating R^2, RMSE, and MAE values

The first part of determining which of all these models was the best was to calculate some measures of success. The measures decided upon were R^2, root mean square error (RMSE), and mean absolute error (MAE). The result of all of these calculations is displayed below. For reference, the closer to $|1|$ that the R^2 value is the better and the lower that RMSE and MAE are the better.

Table of RMSE, MAE, and R^2 for all Models

```
##           model_type model_RMSE model_MAE    model_R2
## 1              Ridge  15.070320  3.445869 -20.8017941
## 2              LASSO   2.435295  1.502410   0.4306879
## 3        Correlation   2.245913  1.344358   0.5157909
## 4              LOESS   4.019281  2.844985  -0.5507580
## 5 Backwards Stepwise   2.318904  1.395687   0.4838060
```

fig 1. This table shows all of the RMSE, MAE, and R^2 values for all of the models that were created.

Chosen Model

After looking at the table of the measures of success it was determined that the correlation model best fit the response variable. More specifically, the model fits very well for low values of meanDOC and not as well for higher values of meanDOC. The variables used in the regression of this model were "'maxTemp_baseFlow", "maxTemp_nonBaseFlow", "precip_yearAvg", "SWE_yearAvg", "soil_Afraction", "elevation_mean", "wasteWaterPointSources_count", and "aspect_predominant_3".

Plots

Below are 2 plots that help to show the results of the correlation model.

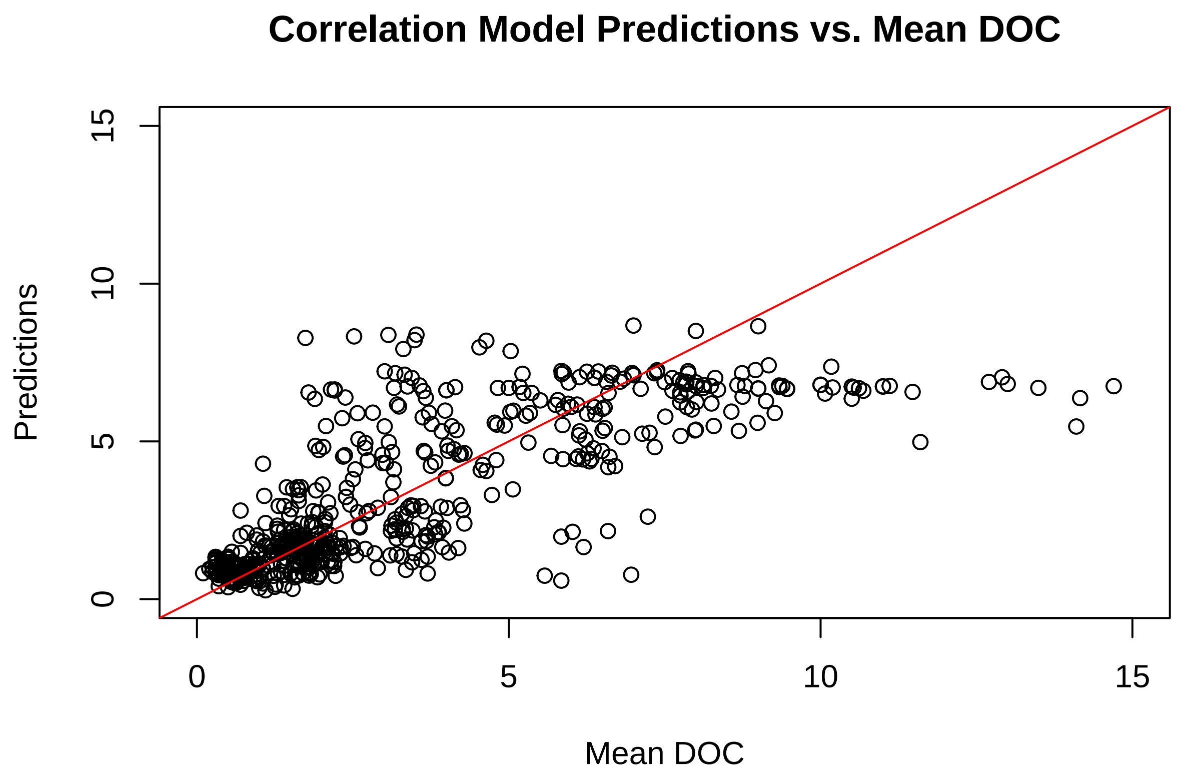

fig 2. This figure plots the predictions made by the correlation model against the mean DOC. The red line signifies where the points would lie if the predictions were the same as mean DOC.

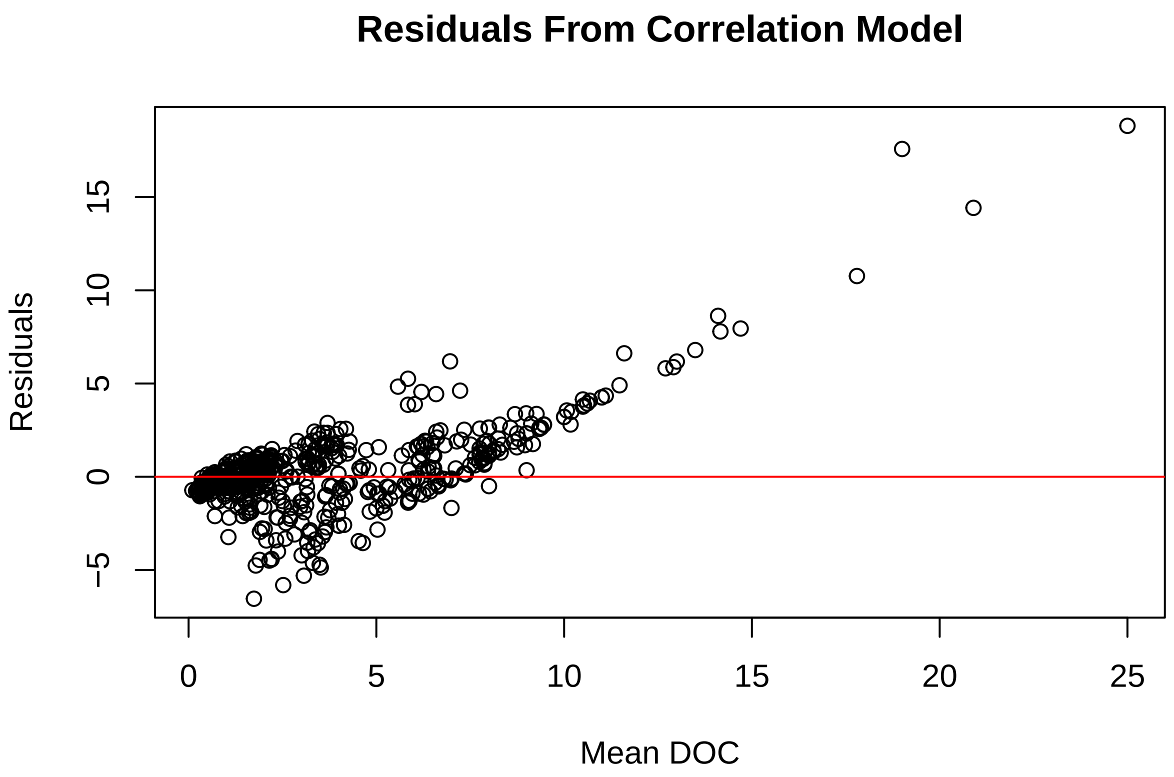

fig 3. This figure plots the residuals from the correlation model against the actual values of mean DOC. A horizontal line is added to show how the residuals increase as mean DOC does.

Discussion

Overall, it was determined based on R2, RMSE, and MAE values that the model created by finding the highest correlated predictor variables to mean DOC and determining their correlation to one another was the best functioning on the test data. So what does this mean as far as the implications within the larger problem. The model that we selected can be used to predict dissolved organic carbon in areas that are affected by the mountain pine beetle. This could help researchers to protect water supplies from carcinogens. They could use these predictions to find where there are large amounts of dissolved organic carbon near bodies of water that eventually feed into water supplies. Then stations can be set up where water can be treated so that carcinogens are not formed when chlorine is added in the water purification process.

APPENDIX

A1 Data Cleaning

```
#Load the data and save it to a shorter name
load("DOC_baseFlow_weightedAvg_and_predictors.RData")
data <- DOC_baseFlow_weightedAvg_and_predictors

#Save the response variable
y_response <- data[,"meanDOC"]

#Create the aspect ratio
xFactors <- model.matrix(y_response ~ as.factor(data[,"aspect_predominant"]))[, -1]

colNames_xFactors <- c("aspect_predominant_1")
for (index in 2:8)
{
  colNames_xFactors <- c(colNames_xFactors,paste("aspect_predominant_",as.character(index),sep=""))
}

colnames(xFactors) <- colNames_xFactors

#Get all the predictors together in an unstandardized form
X_predictors_unStandardized <- as.matrix( cbind( DOC_baseFlow_weightedAvg_and_predictors[,c(2,6,9:30,32
dataFrame_X_predictors <- as.data.frame(X_predictors_unStandardized)

#Standardize all the predictors and put them in a standardized data frame
X_predictors_means <- colMeans(X_predictors_unStandardized)
X_predictors_sd <- apply(X_predictors_unStandardized, MARGIN=2, 'sd')

time_index <-           1
fire_indices <-         2
MPB_indecies <-         3:5
landCover_indices <-    6:8
temp_indices <-         9:14
precipSnow_indices <-   15:20
soil_indices <-         21:23
elevation_index <-      24
wasteWater_index <-     25
aspec_indices <-        26:33
X_indices_subtractMean <- c(time_index, temp_indices, precipSnow_indices, soil_indices, elevation_index
X_indices_scaleBySD <- c(fire_indices)

X_predictors_standardized <- X_predictors_unStandardized
X_predictors_standardized[,X_indices_subtractMean] <- sweep(X_predictors_unStandardized[,X_indices_subt

dataFrame_X_predictors <- as.data.frame(X_predictors_unStandardized)
dataFrame_X_standardized <- as.data.frame(X_predictors_standardized)
```

A2 Train and Test Data

```
y_response_train <- data$meanDOC[1:587]
y_response_test <- data$meanDOC[588:1174]

dataFrame_X_standardized_train <- dataFrame_X_standardized[1:587,]
dataFrame_X_standardized_test <- dataFrame_X_standardized[588:1174,]
```

A3 Correlation Model Construction

```
#Checking correlation of all the independent variables with the dependent
correlation <- round(cor(y_response_train, dataFrame_X_standardized_train), 2)
```

```
## Warning in cor(y_response_train, dataFrame_X_standardized_train): the standard
## deviation is zero
```

```
#Checking correlation between all the dependent variables with over |.5| with
#each other.
high_cor_vars <- c("minTemp_baseFlow", "minTemp_nonBaseFlow", "minTemp_yearAvg",
                   "maxTemp_baseFlow", "maxTemp_nonBaseFlow", "maxTemp_yearAvg",
                   "precip_nonBaseFlow", "precip_yearAvg", "SWE_nonBaseFlow",
                   "SWE_yearAvg", "soil_Afraction", "elevation_mean",
                   "wasteWaterPointSources_count", "aspect_predominant_3")

predictor_data_corr <- dataFrame_X_standardized_train[,high_cor_vars]

#Checks the correlation between all the predictors to better filter the data
correlation_matrix_independent <- as.matrix(cor(predictor_data_corr))

correlation_dependent <- cor(y_response_train, predictor_data_corr)
#Removed minTemp_baseFlow, minTemp_nonBaseFlow

filtered_cor_vars <- c("maxTemp_baseFlow", "maxTemp_nonBaseFlow",
                       "precip_yearAvg", "SWE_yearAvg", "soil_Afraction",
                       "elevation_mean", "wasteWaterPointSources_count",
                       "aspect_predominant_3")
predictor_data_corr2 <- dataFrame_X_standardized_train[,filtered_cor_vars]

correlation_matrix_independent2 <- as.matrix(cor(predictor_data_corr2))

corr_model <- lm(y_response_train ~ ., data = predictor_data_corr2)

corr_model_predict <- predict(corr_model, dataFrame_X_standardized_test)
```

A4 Backward Stepwise Model Construction

```
#Create a full model for the knn function
DOC_linear <- lm(y_response_train ~ ., data = dataFrame_X_standardized_train)
```

```
n = 587

#Create the backwards stepwise model
back_model <- step(DOC_linear, direction="backward", k=log(n), trace = 0)

back_model_predict <- predict(back_model, dataFrame_X_standardized_test)
```

A5 Lasso Model Construction

```
#Rename dataFrame_X_standardized_train
data_real_train <- dataFrame_X_standardized_train
data_real_test <- dataFrame_X_standardized_test

#Create lasso model
suppressMessages(library(glmnet))

x <- data.matrix(data_real_train)
y <- y_response_train

lasso_model <- cv.glmnet(x, y, alpha = 1)



predict_lasso_train <- predict(lasso_model, as.matrix(data_real_train))

predict_lasso_test <- predict(lasso_model, as.matrix(data_real_test))
```

A6 Ridge Model Construction

```
#Create a Ridge model
x <- data.matrix(data_real_train)
y <- y_response_train

ridge_model <- cv.glmnet(x, y, alpha = 0)

predict_ridge_test <- predict(ridge_model, as.matrix(data_real_test))
```

A7 Loess Model Construction

```
#Filter down the predictor matrix
x_best <- x[,c(1,7,18,22)]

#Create a LOESS model
loess_model <- loess(y ~ x_best)

predict_loess <- predict(loess_model, data_real_test)
```