¶ Technical Deliverable

Team 8

Zhiang Li

Yiwei He

Rende Wu

Introduction

Methane is an important energy source for sea-floor life, and we are interested in understanding how the micro-organisms convert it into other compounds. One of the mechanisms for this conversion is called anaerobic oxidation of methane (AOM). Measurements and equipment for measuring AOM are relatively expensive, and therefore our goal is to predict AOM using statistics.

This data set is collected from soil samples from the floor of the Gulf of Mexico. There are a total of 275 observations.

Variables of Interest

- Site Type (shelf, abyssal, or oil seep)
- Depth (various groups indicating the depth where the samples were taken)
- CH4 level
- NO3 level
- NO2 level
- NH4 level
- Sulfide level
- SO4 level
- AOM level

Proposed Approach

After exploring our data set, we found extreme variations among the variables, including substantial outliers. There are a few observations where AOM level exceeds thousands, where the majority has a value of less than 100. Therefore, we hope to first explore the existence of AOM by converting AOM level into 'Yes' and 'No' indicating its presence. Then, using the best model we find, we will subset our data set to those being predicted as containing AOM existence. Then, we hope to fit another model that best predicts the actual level of AOM. In other words, we first try and fit a classification model, then use it to subset our data and fit it into a regression model.

We would also want to note that our main goal is prediction, not interpretation. Therefore, the relationships between predictors and the response variable become less important. Our two most important metrics are: prediction accuracy for classification, and prediction MSE for regression.

Data Cleaning

```
## Load Data
raw <- read.csv("AOM_Data.csv") # raw version
temp <- read.csv("Cleaned_AOM_Data.csv") # cleaned version

## get cleaned version of Sed.Depth from cleaned data set
raw$Sed.Depth <- temp$Sed.Depth
## remove rows with NA values and columns that are not needed
data <- na.omit(raw)[,c(-1,-2,-4,-13,-14)]
## change charactors to factors
data$Site.Type <- factor(data$Site.Type)
## manipulate Sed.Depth by converting it into categorical variables
```

```r
data$Sed.Depth[which(data$Sed.Depth == 'Oily Layer')] <- 0
data$Sed.Depth <- as.numeric(data$Sed.Depth)
## either shallow or deep
data$Sed.Depth <- ifelse(data$Sed.Depth < 15,'shallow','deep')
data$Sed.Depth <- as.factor(data$Sed.Depth)
# adjust and standardize the units (conversion from micromoles to millimoles)
data$Sulfide <- data$Sulfide/1000
data$SO4 <- data$SO4/1000
```

68 out of 275 observations contain NA values. We decide to remove all of the rows that have one or more NA values since the model trained with the removal of all missing values creates a robust model and our data size is still sufficiently large after the removal so we do not need to worry about loss of much information.

Analysis - Classification

We will explore whether there exists certain amount of AOM in sea water.

Data Cleaning

```
## create a new data set for logistic regression
logi_data <- data
## convert AOM into a binary/categorical factor
logi_data$AOM <- ifelse(logi_data$AOM==0,'No','Yes')
logi_data$AOM <- factor(logi_data$AOM)
##evenly split data into train and test sets
set.seed(111)
sample.data<-sample.int(nrow(logi_data), floor(.50*nrow(logi_data)),replace = F)
train<-logi_data[sample.data, ]
test<-logi_data[-sample.data, ]
```

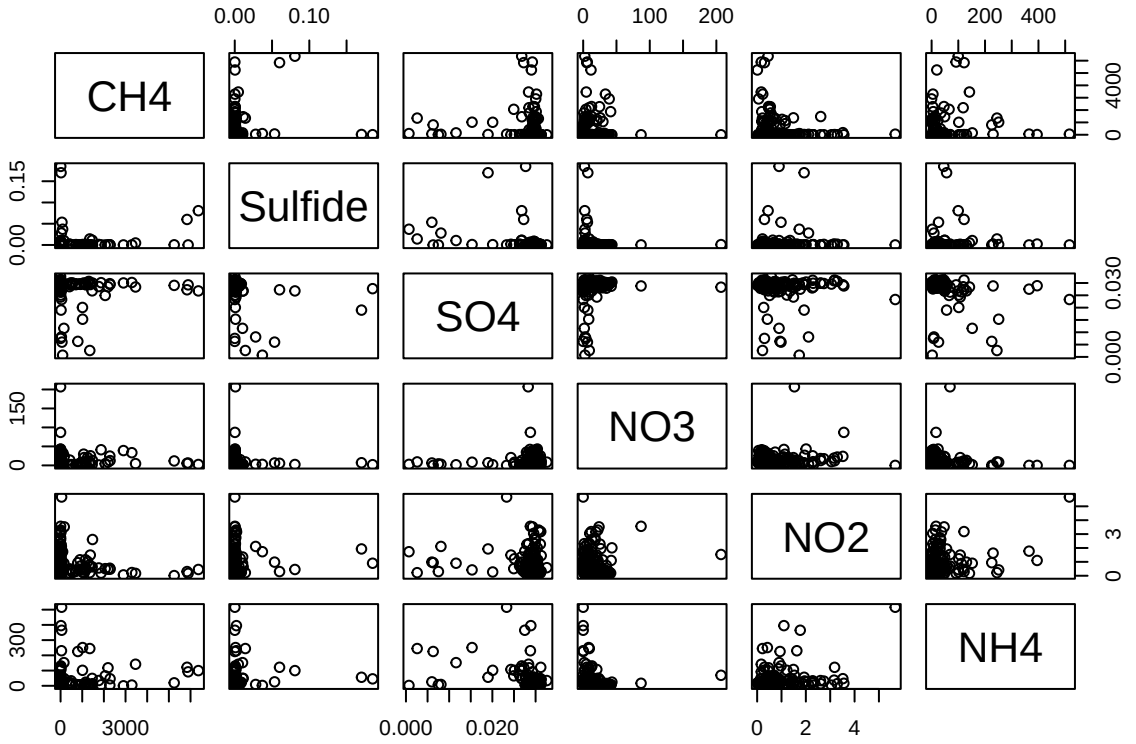
EDA

Quantitative

```
## correlations
cor(logi_data[,c(-1,-2,-3)])
```

```
##                 CH4     Sulfide         SO4         NO3         NO2         NH4
## CH4      1.00000000  0.17306322 -0.05229834 -0.04932157 -0.12957149  0.15120811
## Sulfide  0.17306322  1.00000000 -0.28408835 -0.09244916  0.08222960  0.08187395
## SO4     -0.05229834 -0.28408835  1.00000000  0.11580856 -0.10248666 -0.34023620
## NO3     -0.04932157 -0.09244916  0.11580856  1.00000000  0.08723033 -0.13003879
## NO2     -0.12957149  0.08222960 -0.10248666  0.08723033  1.00000000  0.35033357
## NH4      0.15120811  0.08187395 -0.34023620 -0.13003879  0.35033357  1.00000000
```

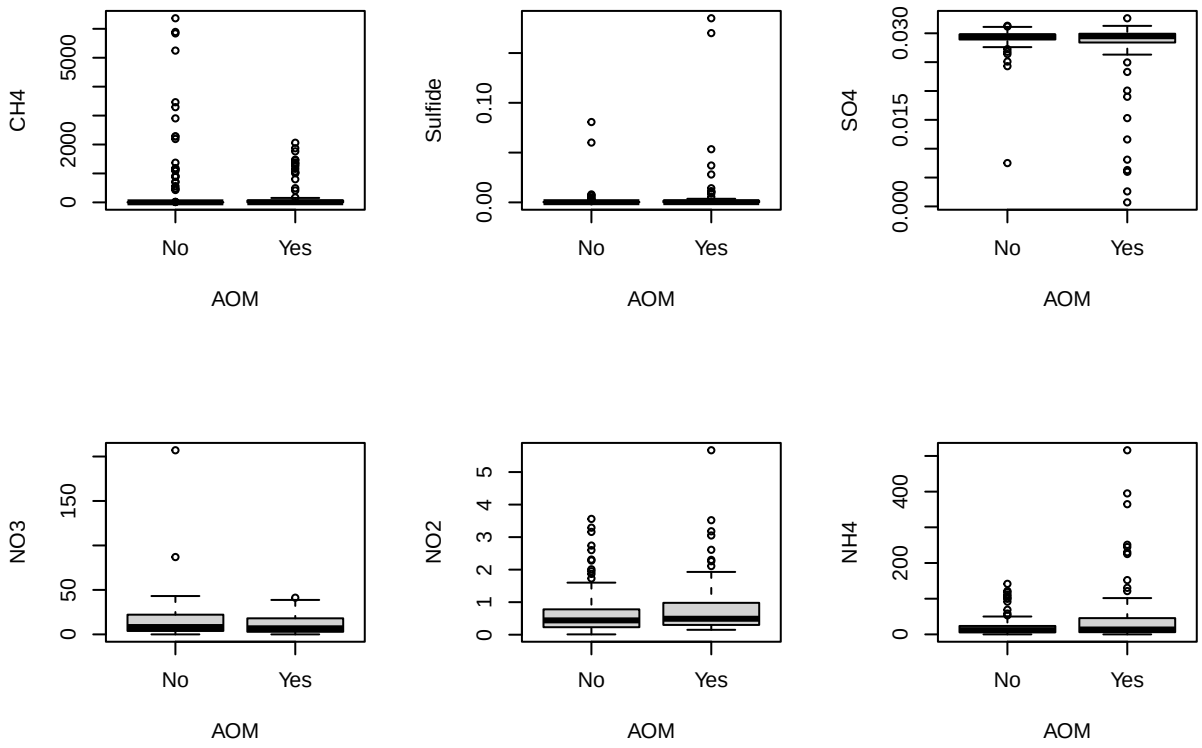
```
pairs(logi_data[,c(-1,-2,-3)])
```



There seems to be no apparent correlations observed and many potential outliers.

Qualitative

```
par(mfrow=c(2,3))
boxplot(CH4~AOM,logi_data)
boxplot(Sulfide~AOM,logi_data)
boxplot(SO4~AOM,logi_data)
boxplot(NO3~AOM,logi_data)
boxplot(NO2~AOM,logi_data)
boxplot(NH4~AOM,logi_data)
```



There seems to be many outliers between the response variable and 6 predictors of choice. The middle 50 percent distribution seems to be highly similar. The amount of outliers might impact our future model-building.

Logistic Regression

Full Model

```
result_train<-glm(AOM~., family=binomial, data=train)
summary(result_train)
```

Model Summary

```
##
## Call:
## glm(formula = AOM ~ ., family = binomial, data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6364  -0.8277  -0.5324   0.9257   2.0545
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
```

```
## (Intercept)       -7.358e-01  2.320e+00  -0.317  0.75112
## Sed.Depthshallow   1.385e+00  5.804e-01   2.387  0.01700 *
## Site.Typeoil seep  1.624e+00  9.535e-01   1.703  0.08862 .
## Site.Typeshelf     7.262e-02  9.727e-01   0.075  0.94049
## CH4               -1.373e-03  5.077e-04  -2.705  0.00684 **
## Sulfide            7.814e+01  6.381e+01   1.225  0.22073
## SO4               -4.470e+01  8.312e+01  -0.538  0.59076
## NO3               -1.790e-02  2.411e-02  -0.743  0.45775
## NO2               -3.346e-01  3.477e-01  -0.962  0.33596
## NH4                9.593e-03  7.531e-03   1.274  0.20271
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 136.66  on 102  degrees of freedom
## Residual deviance: 111.70  on  93  degrees of freedom
## AIC: 131.7
##
## Number of Fisher Scoring iterations: 5
```

Under a significance level of 0.05, only two variables (depth, CH4) are significant. We need to consider the possibility of having a reduced model or even an intercept-only model.

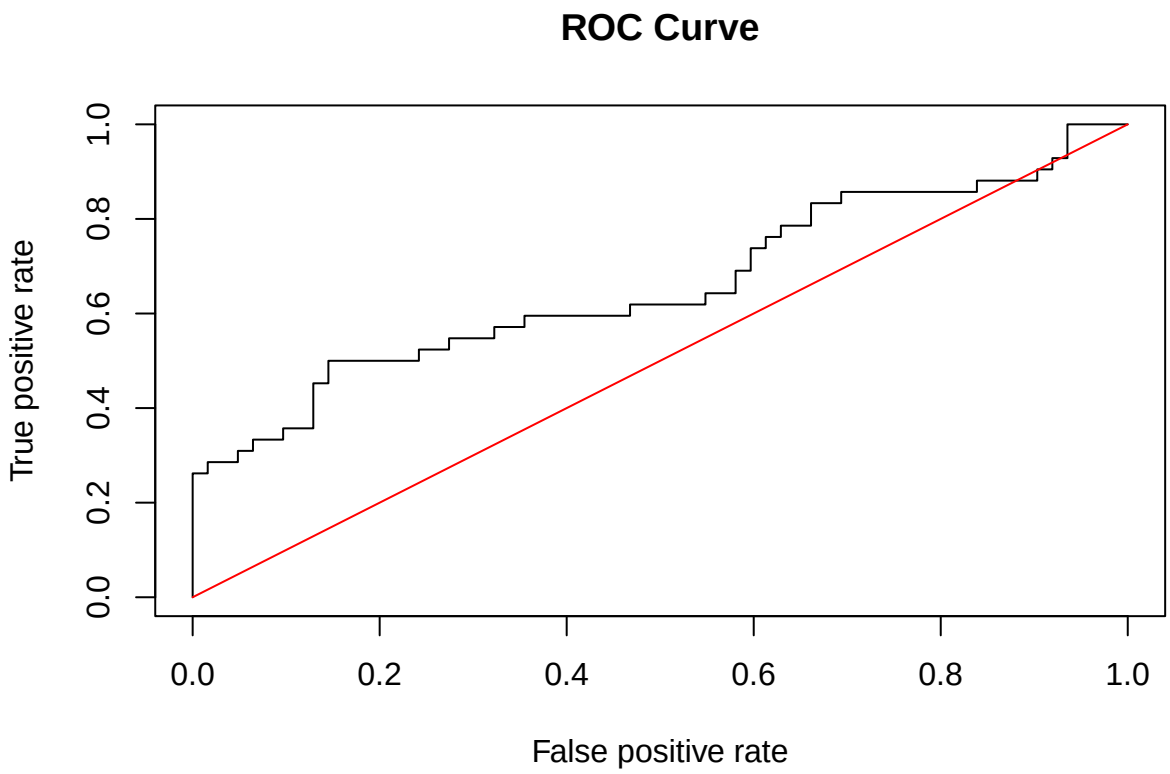
```
TS1<-result_train$null - result_train$dev
1-pchisq(TS1,2)
```

Likelihood Ratio Test

```
## [1] 3.811524e-06
```

We conduct a likelihood ratio test (LRT) to compare our model with an intercept-only model. With a p-value smaller than 0.05, we reject the null and retain the full model.

```
##predicted probabilities for test data based on training data
preds<-predict(result_train,newdata=test, type="response")
##produce the numbers associated with classification table
rates<-ROCR::prediction(preds, test$AOM)
##store the true positive and false postive rates
roc_result<-ROCR::performance(rates,measure="tpr", x.measure="fpr")
##plot ROC curve and overlay the diagonal line for random guessing
plot(roc_result, main="ROC Curve")
lines(x = c(0,1), y = c(0,1), col="red")
```



ROC & AUC

```
auc<-ROCR::performance(rates, measure = "auc")
paste('AUC is', auc@y.values)
```

```
## [1] "AUC is 0.658986175115208"
```

An ROC curve that is above the diagonal indicates the logistic regression does better than random guessing; and ROC curve that is below this diagonal does worse than random guessing. As observed, the curve is mostly above the diagonal, which means our model does better than random guessing.

Another measure is the area under the curve (AUC). A classifier that randomly guesses will have an AUC of 0.5, and we would want our AUC to be closer to 1. With an AUC value of around 0.66, our model seems to just perform a bit well than random guessing model.

```
confusion.mat<-table(test$AOM,preds > 0.5)
confusion.mat
```

Accuracy Rate

```
##
##       FALSE TRUE
##   No     52   10
##   Yes    21   21
```

```
acc.full <- (confusion.mat[1,1]+confusion.mat[2,2])/nrow(test)
paste('Accuracy is',acc.full)
```

```
## [1] "Accuracy is 0.701923076923077"
```

A model accuracy of about 70% is pretty satisfactory.

Stepwise Selection

Backward Selection We will perform a backward selection since the number of samples (n) is larger than the number of variables p in our case.

```
stepAIC(result_train,direction = 'backward')
```

```
## Start:  AIC=131.7
## AOM ~ Sed.Depth + Site.Type + CH4 + Sulfide + SO4 + NO3 + NO2 +
##     NH4
##
##             Df Deviance    AIC
## - SO4        1   111.99 129.99
## - NO3        1   112.26 130.26
## - NO2        1   112.66 130.66
## - Sulfide    1   112.91 130.91
## <none>           111.70 131.70
## - NH4        1   113.91 131.91
## - Sed.Depth  1   118.03 136.03
## - Site.Type  2   120.22 136.22
## - CH4        1   122.21 140.21
##
## Step:  AIC=129.99
## AOM ~ Sed.Depth + Site.Type + CH4 + Sulfide + NO3 + NO2 + NH4
##
##             Df Deviance    AIC
## - NO3        1   112.58 128.58
## - NO2        1   113.02 129.02
## <none>           111.99 129.99
## - NH4        1   115.12 131.12
## - Sulfide    1   115.36 131.36
## - Sed.Depth  1   118.23 134.23
## - Site.Type  2   120.59 134.59
## - CH4        1   124.38 140.38
##
## Step:  AIC=128.58
## AOM ~ Sed.Depth + Site.Type + CH4 + Sulfide + NO2 + NH4
##
##             Df Deviance    AIC
## - NO2        1   113.61 127.61
## <none>           112.58 128.58
## - NH4        1   116.40 130.40
## - Sulfide    1   116.44 130.44
## - Sed.Depth  1   118.23 132.23
```

```
## - Site.Type  2   121.28 133.28
## - CH4        1   125.02 139.02
##
## Step:  AIC=127.61
## AOM ~ Sed.Depth + Site.Type + CH4 + Sulfide + NH4
##
##             Df Deviance    AIC
## <none>           113.61 127.61
## - Sulfide    1   116.77 128.77
## - NH4        1   116.78 128.78
## - Sed.Depth  1   118.38 130.38
## - Site.Type  2   121.28 131.28
## - CH4        1   125.16 137.16


##
## Call:  glm(formula = AOM ~ Sed.Depth + Site.Type + CH4 + Sulfide + NH4,
##     family = binomial, data = train)
##
## Coefficients:
##       (Intercept)   Sed.Depthshallow  Site.Typeoil seep     Site.Typeshelf
##         -2.067935           1.097316           1.280248          -0.106213
##               CH4            Sulfide                NH4
##         -0.001234          81.215169           0.009817
##
## Degrees of Freedom: 102 Total (i.e. Null);  96 Residual
## Null Deviance:       136.7
## Residual Deviance: 113.6     AIC: 127.6
```

Depth, site type, CH4 level, Sulfide level, and NH4 level are retained at last. We will further evaluate the reduced model with the predictors as mentioned.

Reduced Model

```
better_train<-glm(AOM~Sed.Depth+Site.Type+CH4+Sulfide+NH4,
                  family=binomial,data=train)
summary(better_train)
```

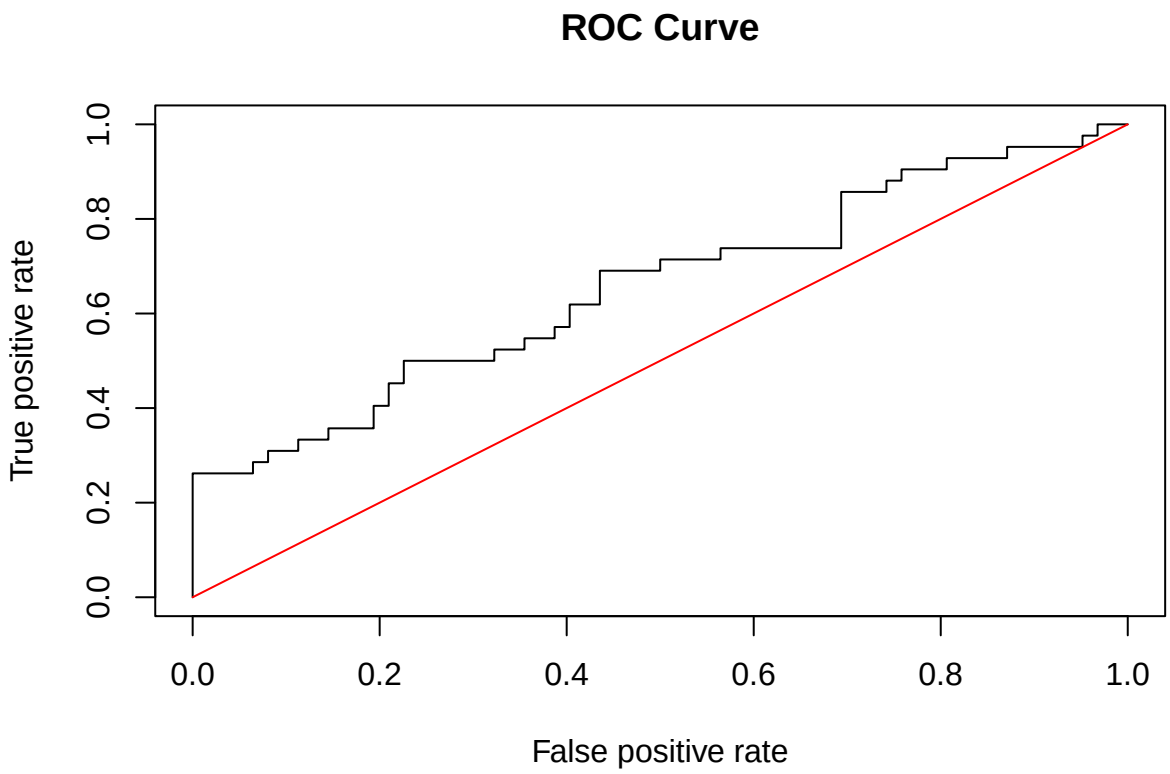
Model Summary

```
##
## Call:
## glm(formula = AOM ~ Sed.Depth + Site.Type + CH4 + Sulfide + NH4,
##     family = binomial, data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5610  -0.8240  -0.5226   0.9661   2.0172
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
```

```
## (Intercept)       -2.0679351  0.8987507  -2.301  0.02140 *
## Sed.Depthshallow   1.0973160  0.5240231   2.094  0.03626 *
## Site.Typeoil seep  1.2802483  0.8901152   1.438  0.15035
## Site.Typeshelf    -0.1062127  0.9168243  -0.116  0.90777
## CH4               -0.0012344  0.0004492  -2.748  0.00599 **
## Sulfide           81.2151693 48.9288164   1.660  0.09694 .
## NH4                0.0098168  0.0066118   1.485  0.13761
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 136.66  on 102  degrees of freedom
## Residual deviance: 113.61  on  96  degrees of freedom
## AIC: 127.61
##
## Number of Fisher Scoring iterations: 5
```

Under a significance level of 0.05, two variables (depth and CH4 level) are significant.

```
##predicted probabilities for test data based on training data
better_preds<-predict(better_train,newdata=test, type="response")
##produce the numbers associated with classification table
better_rates<-ROCR::prediction(better_preds, test$AOM)
##store the true positive and false postive rates
better_roc_result<-ROCR::performance(better_rates,measure="tpr", x.measure="fpr")
##plot ROC curve and overlay the diagonal line for random guessing
plot(better_roc_result, main="ROC Curve")
lines(x = c(0,1), y = c(0,1), col="red")
```



ROC & AUC

```
auc<-ROCR::performance(better_rates, measure = "auc")
paste('AUC is', auc@y.values)
```

```
## [1] "AUC is 0.655529953917051"
```

In comparison to the raw model, ROC seems to be improved, but AUC does not change much.

```
confusion.mat<-table(test$AOM,better_preds > 0.5)
confusion.mat
```

Accuracy Rate

```
##
##       FALSE TRUE
##   No     49   13
##   Yes    23   19
```

```
acc.reduced <- (confusion.mat[1,1]+confusion.mat[2,2])/nrow(test)
paste('Accuracy is',acc.reduced)
```

```
## [1] "Accuracy is 0.653846153846154"
```

The accuracy drops from 70% to 65%, which is concerning.

Model Comparison (Full vs. Reduced)

```
TS2<-better_train$dev - result_train$dev
1-pchisq(TS2,2)
```

```
## [1] 0.3859479
```

With a p-value larger than 0.05, we fail to reject the null hypothesis and move forward with the reduced model. However, the accuracy actually drops by 5% and our goal in this project is to predict AOM level or the existence of AOM. Therefore, we wonder if other model-building methods could provide stronger information.

LDA

Model Assumption 1

```
AOM_yes <- train[which(train$AOM=="Yes"),]
AOM_no <- train[which(train$AOM=="No"),]
ICS::mvnorm.kur.test(AOM_yes[,4:9])
```

```
## Warning in pchisqsum(n * W.stat, df = dfs, a = chi.fac, method = "integration"):
## Probable loss of accuracy

##
##  Multivariate Normality Test Based on Kurtosis
##
## data:  AOM_yes[, 4:9]
## W = 109.59, w1 = 0.625, df1 = 20.000, w2 = 1.000, df2 = 1.000, p-value
## = 2.332e-11
```

```
ICS::mvnorm.skew.test(AOM_yes[,4:9])
```

```
##
##  Multivariate Normality Test Based on Skewness
##
## data:  AOM_yes[, 4:9]
## U = 75.676, df = 6, p-value = 2.787e-14
```

```
ICS::mvnorm.kur.test(AOM_no[,4:9])
```

```
## Warning in pchisqsum(n * W.stat, df = dfs, a = chi.fac, method = "integration"):
## Probable loss of accuracy

##
##  Multivariate Normality Test Based on Kurtosis
##
## data:  AOM_no[, 4:9]
## W = 4335.6, w1 = 0.625, df1 = 20.000, w2 = 1.000, df2 = 1.000, p-value
## = 2.332e-11
```

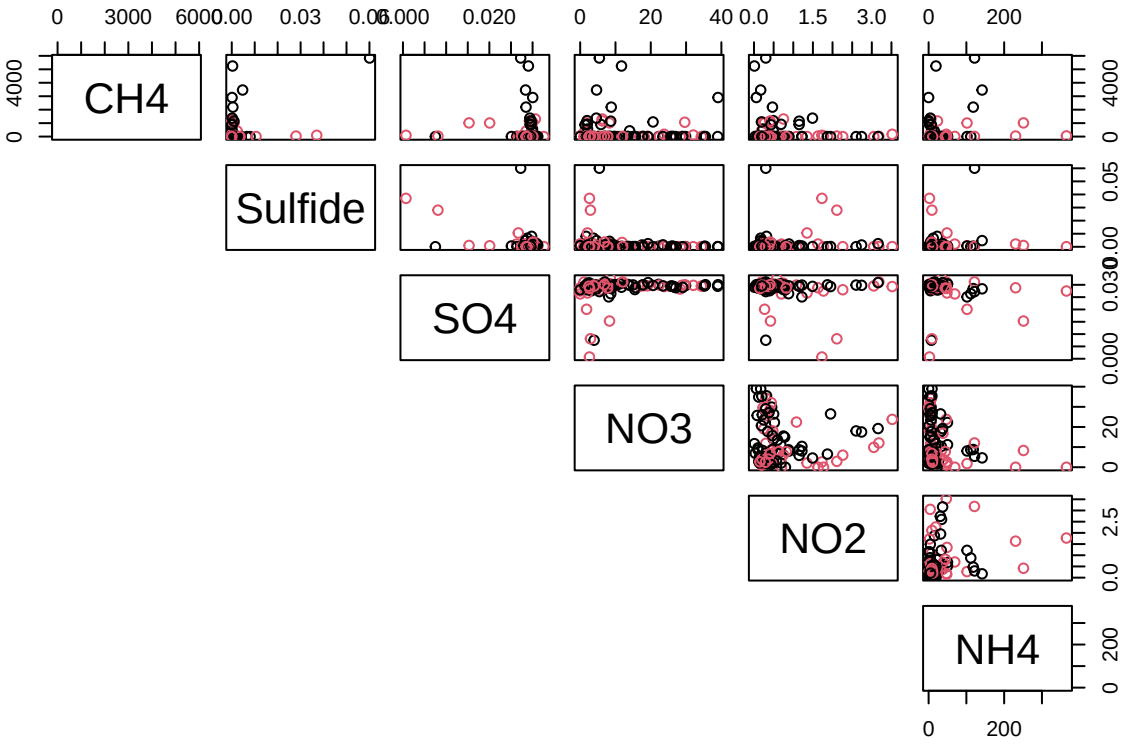
```
ICS::mvnorm.skew.test(AOM_no[,4:9])
```

```
##
##  Multivariate Normality Test Based on Skewness
##
## data:  AOM_no[, 4:9]
## U = 453, df = 6, p-value < 2.2e-16
```

The assumption associated with discriminant analysis is that the predictors follow a multivariate normal distribution for each class of the response variable. Under the MVN tests, we reject the null hypothesis, so we have evidence the assumption for discriminant analysis is not met, but since we only care about prediction not inference, we can keep going.

Model Assumption 2

```
pairs(train[,4:9], col = c(1,2)[train$AOM], lower.panel=NULL)
```



No obvious pattern is observed, but we still continue with model building to see if LDA performs better than logistic regression in prediction.

Model Summary

```
lda.data <- MASS::lda(AOM~CH4+Sulfide+SO4+NO3+NO2+NH4, data=train)
lda.data
```

```
## Call:
## lda(AOM ~ CH4 + Sulfide + SO4 + NO3 + NO2 + NH4, data = train)
##
## Prior probabilities of groups:
##        No       Yes
## 0.6213592 0.3786408
##
## Group means:
##          CH4     Sulfide        SO4      NO3       NO2      NH4
## No  441.6300 0.001681250 0.02905453 12.38937 0.5907813 21.68703
## Yes 169.0046 0.002565897 0.02755410 10.62795 0.8520513 43.89718
##
## Coefficients of linear discriminants:
##                   LD1
## CH4     -7.088888e-04
## Sulfide  3.124532e+01
## SO4     -6.639328e+01
## NO3     -2.838121e-03
## NO2      3.320076e-01
## NH4      1.220438e-02
```

Sulfide level contributes most information to AOM existence, while others have minimal impact, as indicated by the LD1 values.

Accuracy Rate

```
##predictions on test data.
lda.test <- predict(lda.data,test)
##confusion matrix. By default, threshold is 0.5
table(test$AOM,lda.test$class)
```

```
##
##       No Yes
##   No  61   1
##   Yes 33   9
```

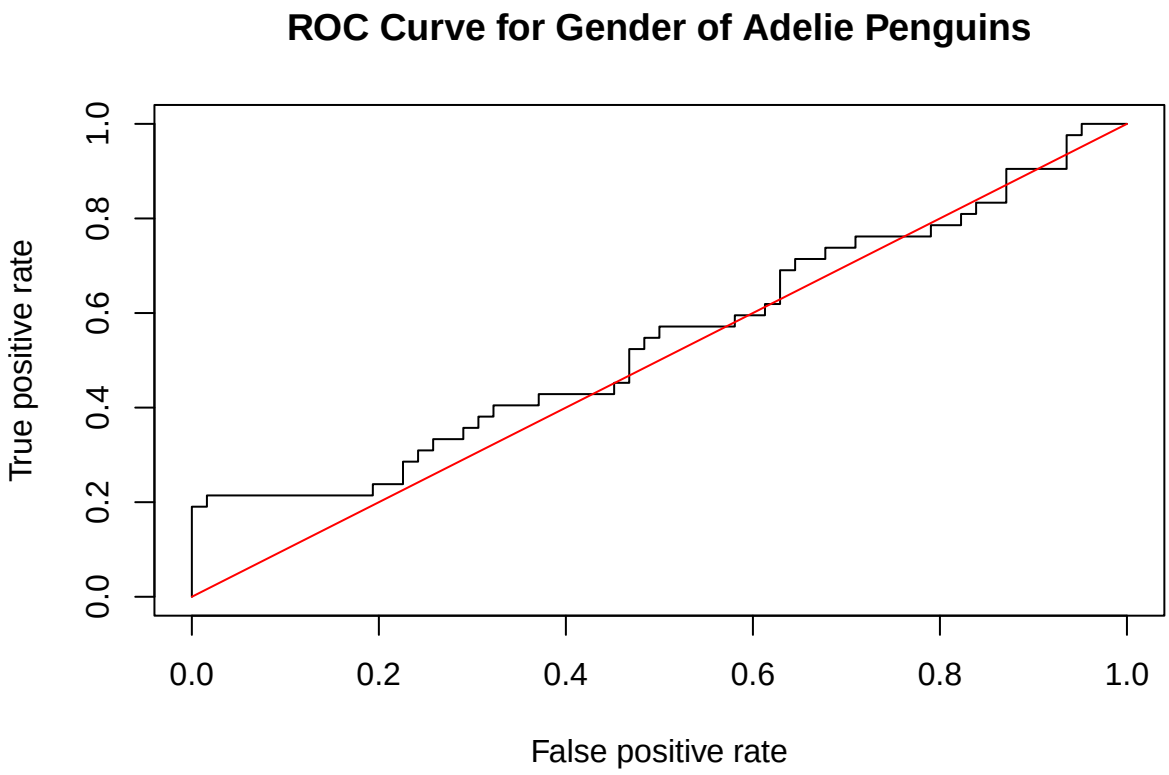
```
##accuracy on test data
acc.lda <- mean(test$AOM == lda.test$class)
acc.lda
```

```
## [1] 0.6730769
```

An accuracy of 0.6730769 is a bit better than our reduced logistic regression model, but worse than our full logistic regression model.

ROC & AUC

```
lda.preds<-lda.test$posterior[,2]
lda.rates<-ROCR::prediction(lda.preds, test$AOM)
lda.roc_result<-ROCR::performance(lda.rates,measure="tpr", x.measure="fpr")
plot(lda.roc_result, main="ROC Curve for Gender of Adelie Penguins")
lines(x = c(0,1), y = c(0,1), col="red")
```




```
auc<-ROCR::performance(lda.rates, measure = "auc")
paste('AUC is', auc@y.values)
```

```
## [1] "AUC is 0.542242703533026"
```

ROC seems to follow the diagonal, while the AUC is close to 0.5, both of which are worse than what we got in logistic regression and indicates that our LDA model performs similarly to random guessing.

Model Comparison (LDA vs. Logistic)

Based on our outputs in LDA and logistic regression, our full logistic regression model is the best in predicting AOM existence. We will continue to try other models since the accuracy level is still not high enough.

Classification Tree

Recursive Binary Splitting

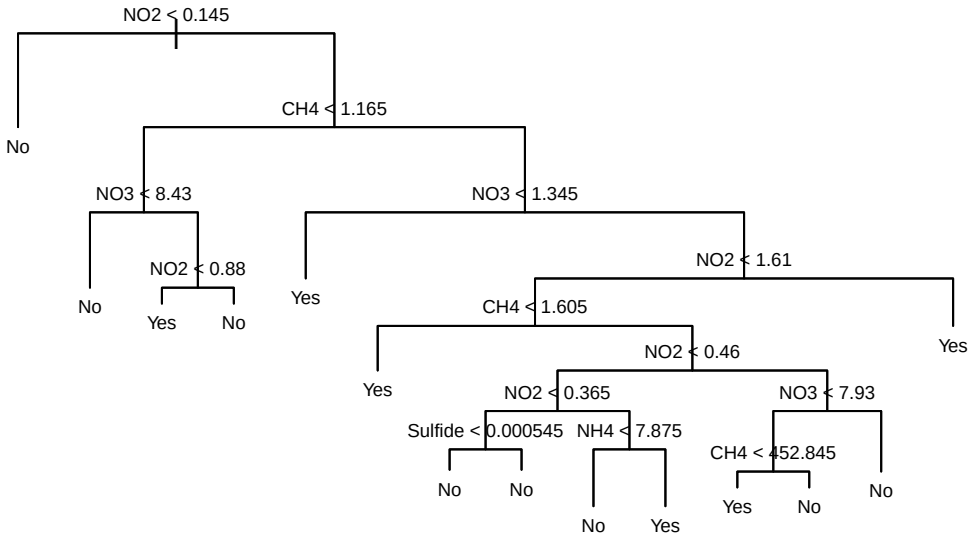
```
tree.class.train<-tree::tree(AOM~., data=train)
summary(tree.class.train)
```

Model Summary

```
##
## Classification tree:
## tree::tree(formula = AOM ~ ., data = train)
## Variables actually used in tree construction:
## [1] "NO2"     "CH4"     "NO3"     "Sulfide" "NH4"
## Number of terminal nodes:  14
## Residual mean deviance:  0.7307 = 65.03 / 89
## Misclassification error rate: 0.165 = 17 / 103
```

There are a total of 14 terminal nodes and five chosen predictors, which is acceptable.

```
plot(tree.class.train)
text(tree.class.train, cex=0.6)
```



Tree Plot

```
##prediction based on pruned tree for test data
tree.pred.prune<-predict(tree.class.train, newdata=test, type="class")
##confusion matrix for test data
table(test$AOM, tree.pred.prune)
```

Accuracy Rate

```
##      tree.pred.prune
##       No Yes
##   No  47  15
##   Yes 26  16
```

```
acc.tree <- mean(tree.pred.prune==test$AOM)
acc.tree
```

```
## [1] 0.6057692
```

The accuracy rate (61%) is still similar to what we got in logistic regression. We can consider pruning our tree.

Pruning

```
set.seed(4996)
cv.class <- tree::cv.tree(tree.class.train, K=10, FUN=prune.misclass)
trees.num.class<-cv.class$size[which.min(cv.class$dev)]
trees.num.class
```

```
## [1] 14
```

```
# prune.class<-tree::prune.misclass(tree.class.train, best=trees.num.class)
# plot(prune.class)
# text(prune.class, cex=0.75, pretty=0)
```

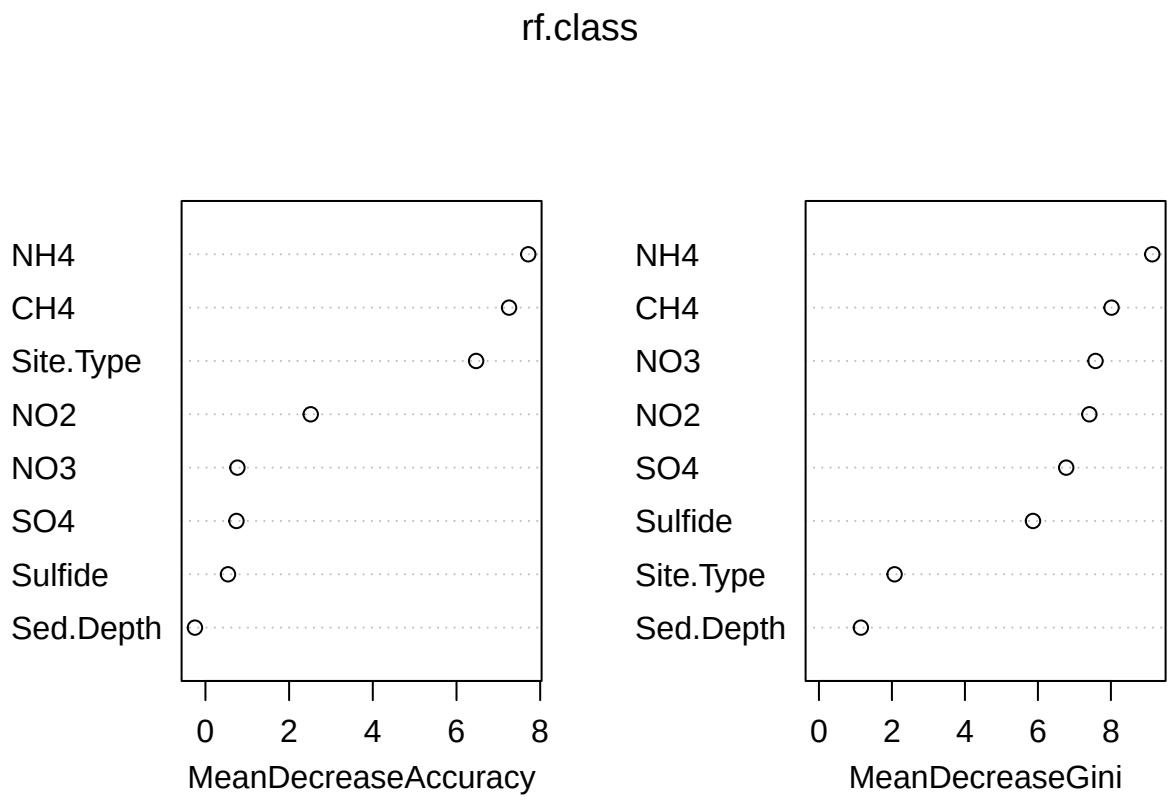
Pruning does not reduce our number of predictors or terminal nodes. However, we can still improve our prediction accuracy by optimizing our tree model.

Random Forest

```
set.seed(4996)
size.class <- floor(sqrt(ncol(train)))
rf.class <- randomForest::randomForest(AOM~., data=train,
                                       mtry=size.class, importance=TRUE)
randomForest::importance(rf.class)
```

```
##                   No        Yes MeanDecreaseAccuracy MeanDecreaseGini
## Sed.Depth  0.6636838 -1.0790914           -0.2508692         1.149195
## Site.Type  6.8729358  0.8698476            6.4690239         2.073107
## CH4        8.0466045  2.2117598            7.2564507         8.017058
## Sulfide   -2.3858624  4.3637941            0.5395078         5.864541
## SO4        1.5298107 -0.4837641            0.7415800         6.775125
## NO3        0.1628638  1.0507559            0.7643691         7.577892
## NO2        4.3934880 -1.4796694            2.5161864         7.408994
## NH4        8.8436834  1.9851169            7.7150959         9.132865
```

```
randomForest::varImpPlot(rf.class)
```



NH4 and CH4 level are the most important predictors in our random forest model.

```
# prediction
pred.rf.class <- predict(rf.class, newdata=test, type='class')
# confusion matrix
matrix.rf <- table(test$AOM, pred.rf.class)
acc.rf <- mean(pred.rf.class==test$AOM)
acc.rf
```

Accuracy Rate

```
## [1] 0.7211538
```

An accuracy level of around 0.72 is better than all models tried previously.

Model Comparison (Tree vs. Logistic)

Random forest performs slightly better in predicting the existence of AOM, in comparison to recursive binary splitting tree.

Conclusion

```
class.matrix <- data.frame(logistic.full=acc.full,
                           logistic.reduced=acc.reduced,
                           lda=acc.lda,
                           tree=acc.tree,
                           random.forest=acc.rf)
class.matrix
```

```
##   logistic.full logistic.reduced       lda      tree random.forest
## 1     0.7019231        0.6538462 0.6730769 0.6057692     0.7211538
```

As shown above, random forest model is the best model to predict AOM existence, with a prediction accuracy of 72%. We now end our classification investigation and will continue with predicting the actual AOM level with data points selected by our random forest model.

Analysis - Regression

Rather than using the entire data set, we apply random forest model to predict AOM existence, and subset those with predicted existence. We then exclude those that are falsely predicted by the random forest model. We also exclude AOM with zero values since it basically implies that there is no AOM existence.

However, there are some outliers with extremely high AOM values (most of them are hundreds, while certain ones are larger than 5000). This could impact our model performance so even though our random forest model might keep them, we decide to filter our data set again to remove these outliers. We will refer to the 1.5 IQR rule for checking outliers.

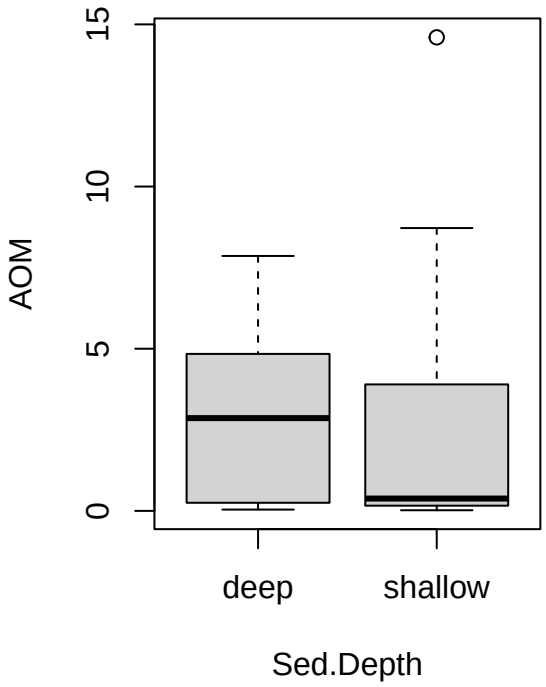
Data Cleaning

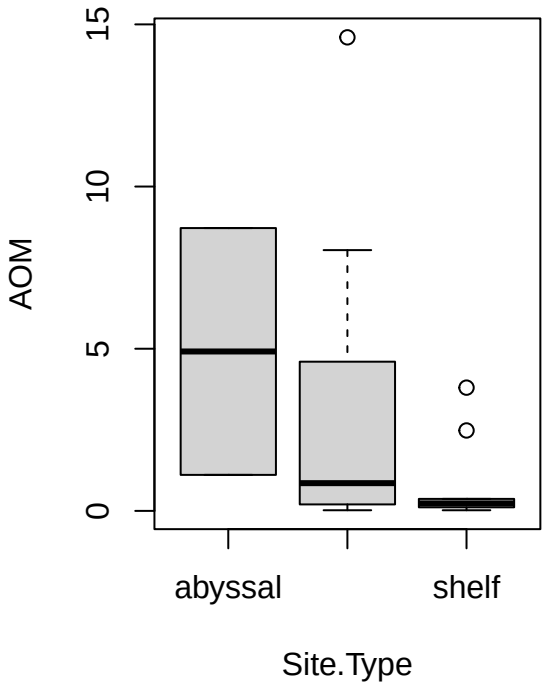
```
line_data <- data[predict(rf.class,newdata=data,type='class')=='Yes',]
line_data <- line_data[line_data$AOM>0,]
# remove outliers
quarter25 <- quantile(line_data$AOM,0.25)
quarter75 <- quantile(line_data$AOM,0.75)
iqr <- quarter75 - quarter25
line_data <- line_data[line_data$AOM<=quarter75+1.5*iqr &
                       line_data$AOM>=quarter25-1.5*iqr,]

# training & testing sets split
set.seed(4996)
sample_line <- sample.int(nrow(line_data),floor(.50*nrow(line_data)),replace=F)
train_line <- line_data[sample_line, ]
test_line <- line_data[-sample_line, ]
test_line <- test_line[-c(6,7), ]
```

EDA

Qualitative

```
par(mfrow=c(1,2))
boxplot(AOM~Sed.Depth,line_data)
boxplot(AOM~Site.Type,line_data)
```

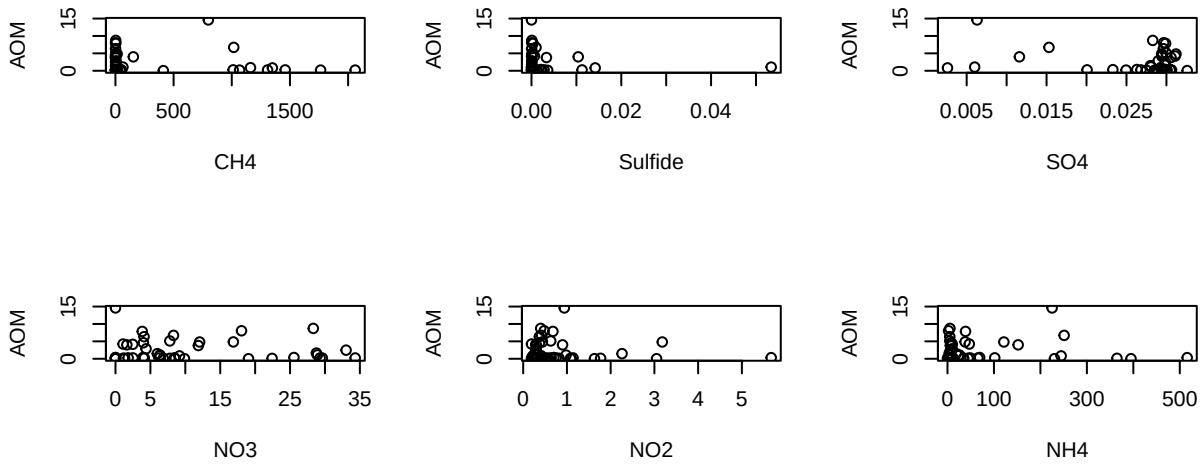




With outliers removed, higher AOM levels seem to appear in deeper sea levels. AOM level also varies drastically across the three site types.

Qualitative

```
par(mfrow=c(3,3))
plot(AOM~CH4,line_data)
plot(AOM~Sulfide,line_data)
plot(AOM~SO4,line_data)
plot(AOM~NO3,line_data)
plot(AOM~NO2,line_data)
plot(AOM~NH4,line_data)
```

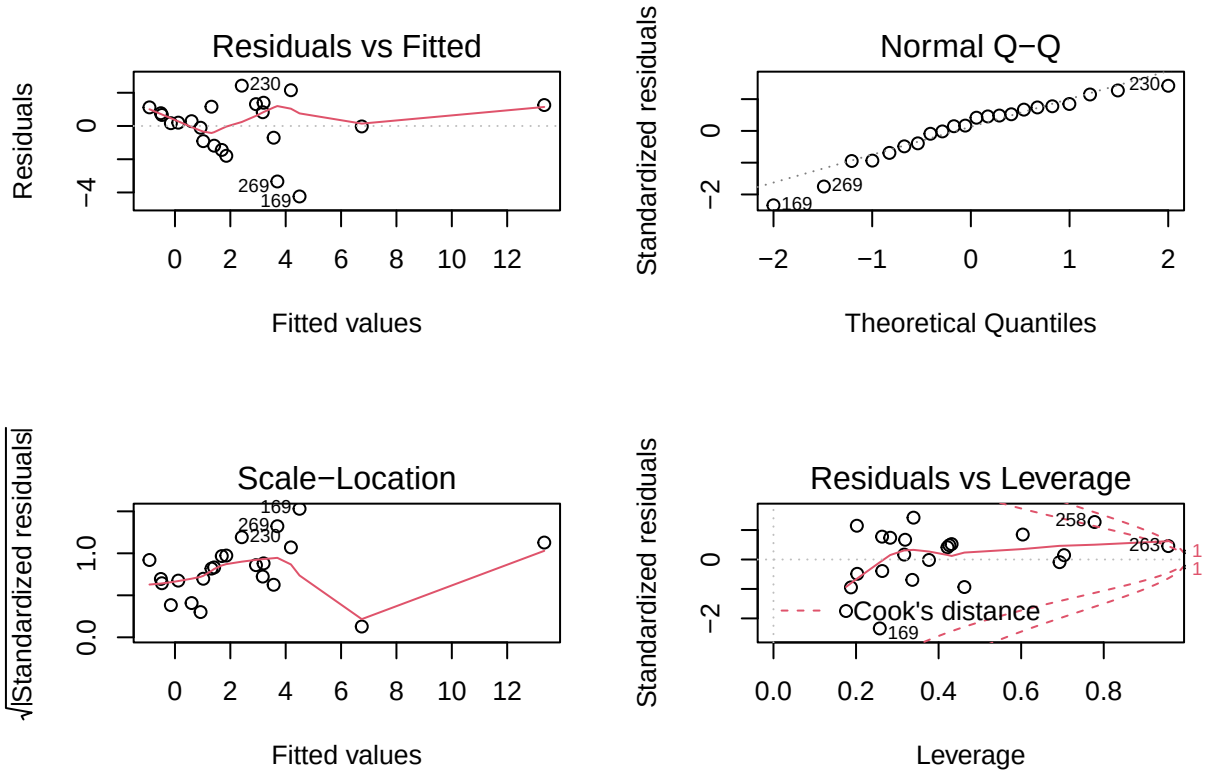


There does not seem to be apparent associations between AOM level and any of the predictors.

Linear Regression

Model Assumption

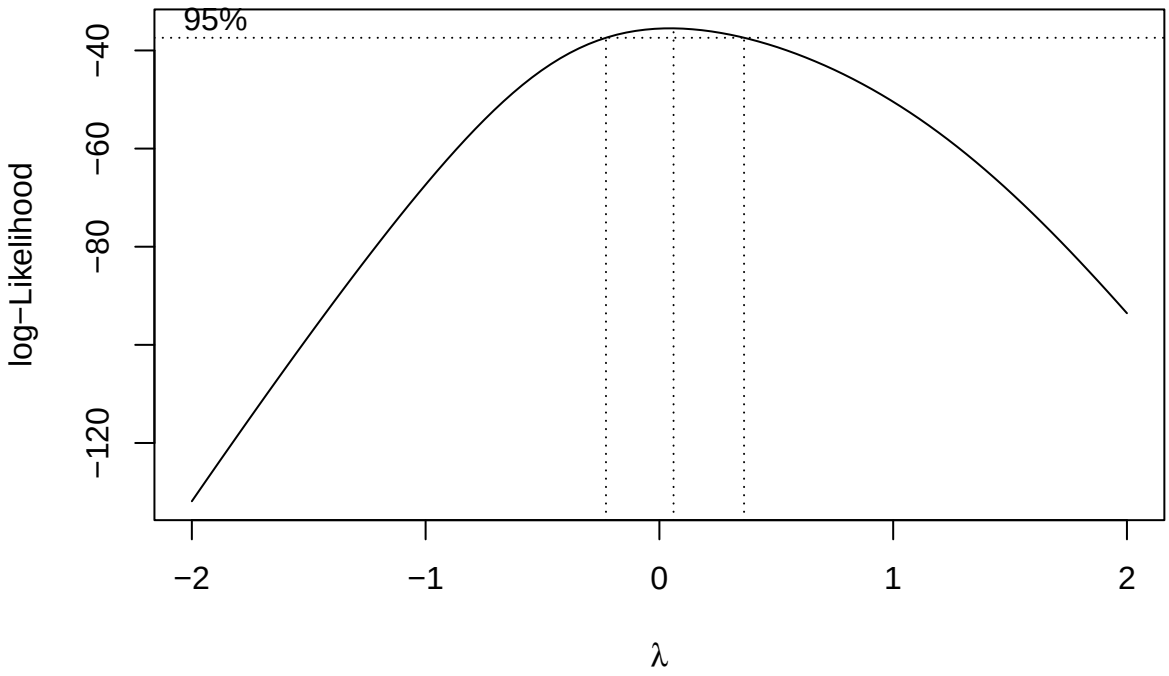
```
lm_result<-lm(AOM~., data=train_line)
par(mfrow=c(2,2))
plot(lm_result)
```



Our data does not pass the linear regression model assumptions. The residuals follow along the horizontal line with minor curvature. The normality assumption is good, following the 45 degree line. However, there seems to be an apparent curvature in the scale-location plot and seems to be a lot of outliers as shown in the residual-leverage plot. We can consider transforming our data to better meet the model assumptions.

Evalue Need for Transformation

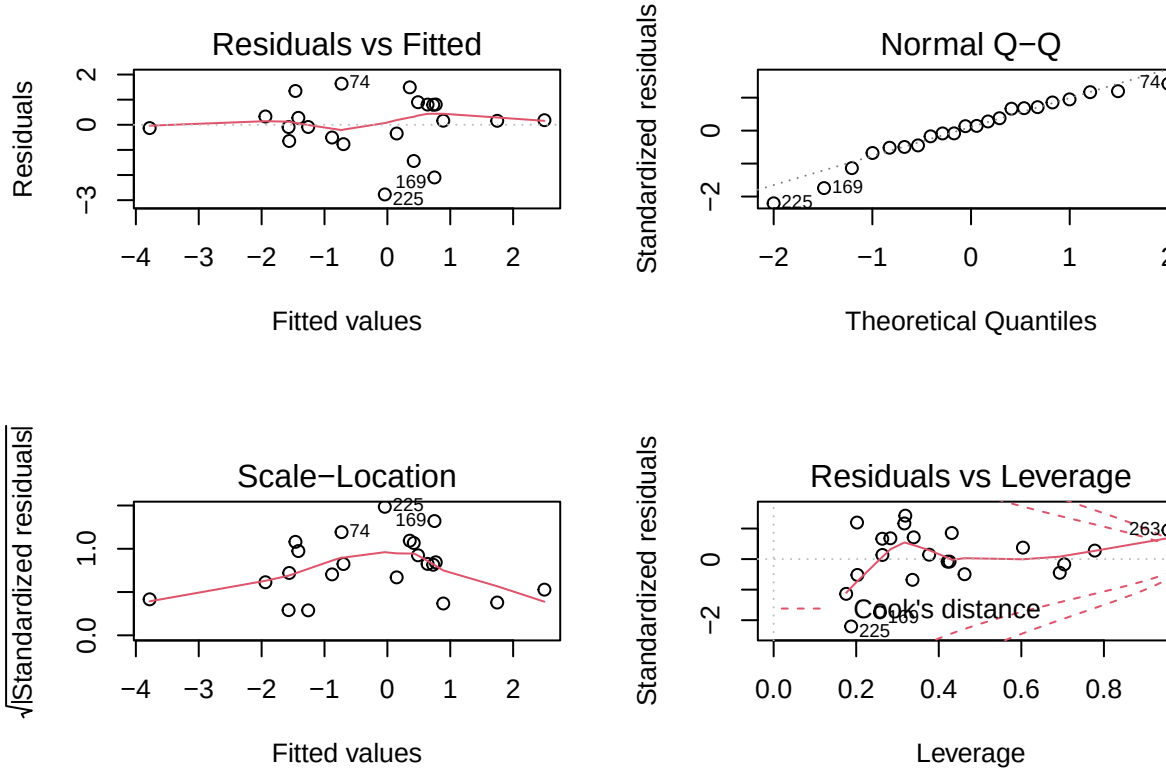
```
par(mfrow=c(1,1))
MASS::boxcox(lm_result)
```



As shown in the boxcox plot, with a lambda of around 0, we would need a log transformation.

Log-Linear Regression Model

```
log_result<-lm(log(AOM)~., data=train_line)
par(mfrow=c(2,2))
plot(log_result)
```



Model Assumption

After taking a log transformation, our model better meets the regression assumptions.

```
summary(log_result)
```

Model Summary

```
##
## Call:
## lm(formula = log(AOM) ~ ., data = train_line)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.7727 -0.4668  0.1599  0.8029  1.6366
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)       4.229e+00  2.477e+00   1.707   0.1115
## Sed.Depthshallow -3.528e-01  8.251e-01  -0.428   0.6759
## Site.Typeshelf   -9.011e-01  9.820e-01  -0.918   0.3755
## CH4              -1.668e-03  6.324e-04  -2.637   0.0205 *
## Sulfide          -1.931e+02  1.252e+02  -1.542   0.1470
## SO4              -1.039e+02  8.100e+01  -1.283   0.2218
## NO3              -8.329e-03  3.449e-02  -0.242   0.8129
## NO2              -1.183e+00  4.706e-01  -2.514   0.0259 *
```

```
## NH4              7.640e-03  5.960e-03   1.282   0.2223
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.395 on 13 degrees of freedom
## Multiple R-squared:  0.622,  Adjusted R-squared:  0.3894
## F-statistic: 2.674 on 8 and 13 DF,  p-value: 0.05578
```

With a p-value slightly larger than 0.05, we fail to reject the null. This model is not sufficient in predicting AOM level and we consider reducing the model to better fit the prediction.

Stepwise Selection

```
stepAIC(log_result,direction = 'backward')
```

```
## Start:  AIC=21.08
## log(AOM) ~ Sed.Depth + Site.Type + CH4 + Sulfide + SO4 + NO3 +
##     NO2 + NH4
##
##             Df Sum of Sq    RSS    AIC
## - NO3        1    0.1135 25.419 19.178
## - Sed.Depth  1    0.3560 25.662 19.387
## - Site.Type  1    1.6392 26.945 20.460
## <none>                  25.306 21.080
## - NH4        1    3.1986 28.504 21.698
## - SO4        1    3.2056 28.511 21.704
## - Sulfide    1    4.6301 29.936 22.776
## - NO2        1   12.2999 37.605 27.794
## - CH4        1   13.5365 38.842 28.506
##
## Step:  AIC=19.18
## log(AOM) ~ Sed.Depth + Site.Type + CH4 + Sulfide + SO4 + NO2 +
##     NH4
##
##             Df Sum of Sq    RSS    AIC
## - Sed.Depth  1    0.4735 25.893 17.584
## <none>                  25.419 19.178
## - Site.Type  1    2.4420 27.861 19.196
## - NH4        1    3.1837 28.603 19.774
## - SO4        1    3.5134 28.933 20.026
## - Sulfide    1    4.6086 30.028 20.844
## - NO2        1   12.2563 37.675 25.835
## - CH4        1   14.0806 39.500 26.876
##
## Step:  AIC=17.58
## log(AOM) ~ Site.Type + CH4 + Sulfide + SO4 + NO2 + NH4
##
##             Df Sum of Sq    RSS    AIC
## <none>                  25.893 17.584
## - NH4        1    2.9231 28.816 17.937
## - Site.Type  1    3.4940 29.387 18.369
## - SO4        1    3.5786 29.471 18.432
```

```
## - Sulfide    1    5.2885 31.181 19.673
## - NO2        1   12.2577 38.150 24.111
## - CH4        1   13.6263 39.519 24.886

##
## Call:
## lm(formula = log(AOM) ~ Site.Type + CH4 + Sulfide + SO4 + NO2 +
##     NH4, data = train_line)
##
## Coefficients:
##    (Intercept)  Site.Typeshelf             CH4         Sulfide             SO4
##      4.048e+00      -1.137e+00      -1.642e-03      -2.033e+02      -1.082e+02
##            NO2             NH4
##     -1.164e+00       7.242e-03
```

Site type, CH4, Sulfide, SO4, NO2, and NH4 are retained after stepwise selection. We will further evaluate the reduced model with the predictors as mentioned.

Evaluate Reduced Model

```
better_log_result<-lm(log(AOM)~Site.Type+CH4+Sulfide+SO4+NO2+NH4,data=train_line)
summary(better_log_result)
```

Model Summary

```
##
## Call:
## lm(formula = log(AOM) ~ Site.Type + CH4 + Sulfide + SO4 + NO2 +
##     NH4, data = train_line)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.87291 -0.59746  0.07376  0.87500  1.54975
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     4.048e+00  2.288e+00   1.769   0.0971 .
## Site.Typeshelf -1.137e+00  7.990e-01  -1.423   0.1753
## CH4            -1.642e-03  5.843e-04  -2.810   0.0132 *
## Sulfide        -2.033e+02  1.161e+02  -1.750   0.1005
## SO4            -1.082e+02  7.515e+01  -1.440   0.1705
## NO2            -1.164e+00  4.367e-01  -2.665   0.0177 *
## NH4             7.242e-03  5.565e-03   1.301   0.2128
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.314 on 15 degrees of freedom
## Multiple R-squared:  0.6132, Adjusted R-squared:  0.4585
## F-statistic: 3.964 on 6 and 15 DF,  p-value: 0.01416
```

With a p-value much less than 0.05, we reject the null and our model can sufficiently provide information about AOM level. However, there are only two significant variables (CH4 and NO2 level), so we further compare between the model selected by stepwise and the reduced model with the two significant variables as predictors.

```
reduced_log_result <- lm(log(AOM)~CH4+NO2,data=train_line)
anova(reduced_log_result,better_log_result)
```

Model Comparison (Stepwie vs. Stepwise-Reduced)

```
## Analysis of Variance Table
##
## Model 1: log(AOM) ~ CH4 + NO2
## Model 2: log(AOM) ~ Site.Type + CH4 + Sulfide + SO4 + NO2 + NH4
##   Res.Df    RSS Df Sum of Sq      F  Pr(>F)
## 1     19 59.430
## 2     15 25.893  4    33.538 4.8572 0.01029 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

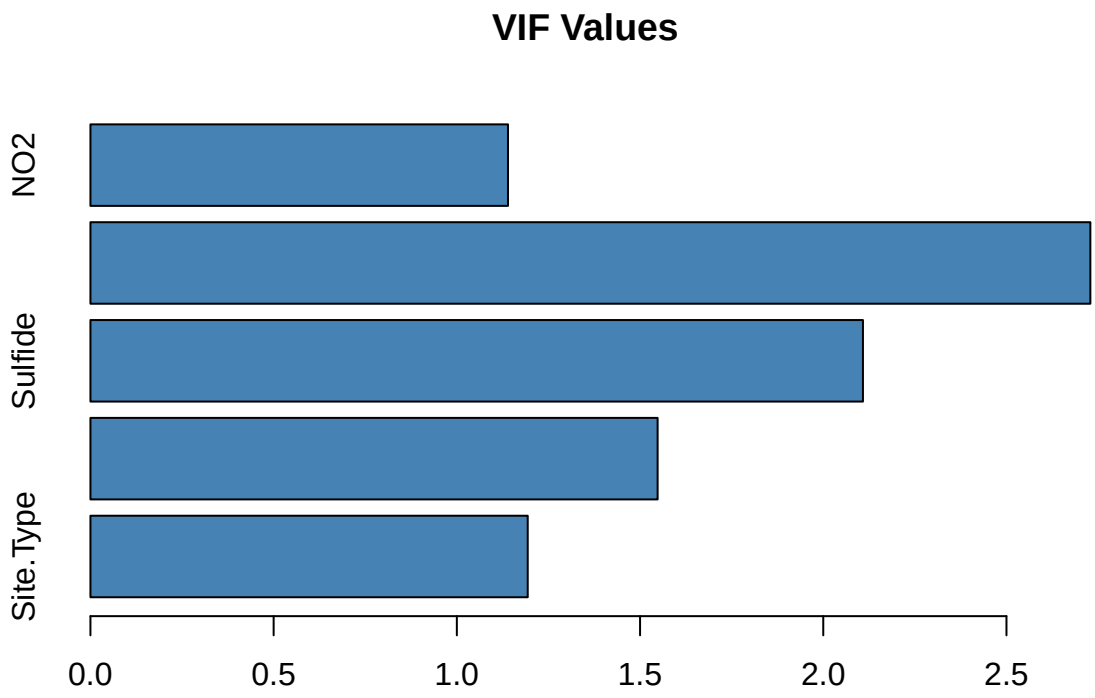
With a p-value less than 0.05, we reject the null and move forward with the model suggested by backwards selection.

```
VIF_check <- lm(AOM~Site.Type+CH4+Sulfide+SO4+NO2+CH4,data=train_line)
car::vif(VIF_check)
```

VIF & Multicolinearity Check

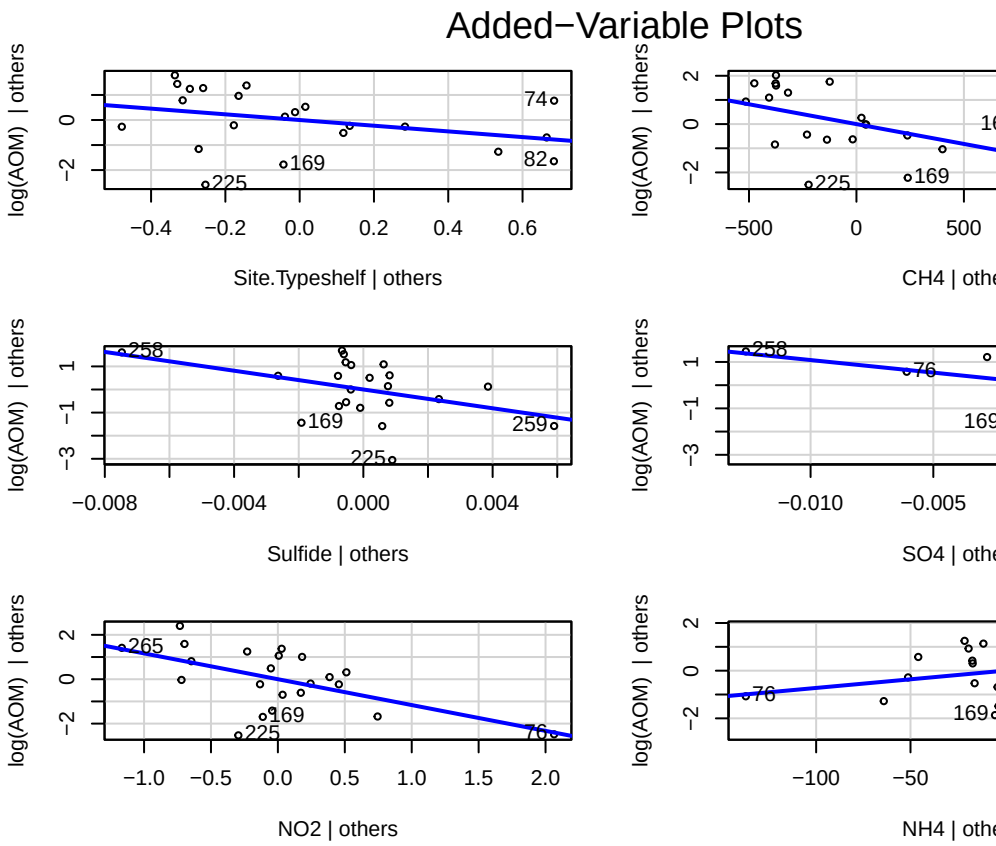
```
## Site.Type       CH4   Sulfide       SO4       NO2
##  1.193420  1.547942  2.108286  2.728946  1.139531
```

```
#create horizontal bar chart to display each VIF value
barplot(vif(VIF_check), main = "VIF Values", horiz = TRUE, col = "steelblue")

#add vertical line at 5
#greater than 5 indicating prescence of multicolinearity
abline(v = 5, lwd = 3, lty = 2)
```



VIF score is good for all of the quantitative factors indicating there is no multicolinearity.

```
avPlots(better_log_result)
```



Model Graphical Interpretation

It seems like linear regression is not the best option since the line does not fit all the points well. Many points just vertically spread out at the sides of the line.

```
pred<-predict.lm(better_log_result,test_line)
mse.reg<-mean((log(test_line$AOM)-pred)^2)
mse.reg
```

Test MSE

```
## [1] 9.808248
```

A test MSE of around 9.8 is pretty low, but we will continue with further optimization, such as Ridge regression.

Ridge Regression

```
x <- model.matrix(AOM~.,data=line_data)[,-1]
y <- line_data$AOM
##split data
set.seed(4996)
sample.data.ridge<-sample.int(nrow(line_data), floor(.50*nrow(line_data)), replace = F)
```

```
x.train<-x[sample.data.ridge,]
x.test<-x[-sample.data.ridge,]
y.train<-log(y[sample.data.ridge])
y.test<-log(y[-sample.data.ridge])
##use CV to find optimal lambda based on training set
set.seed(4996)
cv.out.ridge<-glmnet::cv.glmnet(x.train,y.train,alpha=0,nfolds=5,thresh = 1e-23)
bestlam.ridge<-cv.out.ridge$lambda.min
```

```
##fit ridge regression using training data and bestlam
ridge.mod<-glmnet::glmnet(x.train,y.train,alpha=0,lambda=bestlam.ridge,nfolds=5,thresh = 1e-23)
##Test MSE with best lambda
ridge.pred<-predict(ridge.mod,newx=x.test)
mse.ridge <- mean((ridge.pred-y.test)^2)
mse.ridge
```

Test MSE

```
## [1] 5.034566
```

The test MSE drastically reduces after fitting ridge regression, dropping from 8 to 5. Since we are focusing on prediction, we are not worried about interpreting the coefficients.

Lasso Regression

```
##use CV to find optimal lambda based on training set
set.seed(4996)
cv.out.lasso<-glmnet::cv.glmnet(x.train,y.train,alpha=1,nfolds=5,thresh = 1e-23)
bestlam.lasso<-cv.out.lasso$lambda.min
##fit lasso regression using training data and bestlam
lasso.mod<-glmnet::glmnet(x.train,y.train,alpha=1,lambda=bestlam.lasso,nfolds=5,thresh = 1e-23)
##Test MSE with best lambda
lasso.pred<-predict(lasso.mod,newx=x.test)
mse.lasso <- mean((lasso.pred-y.test)^2)
mse.lasso
```

Test MSE

```
## [1] 5.781116
```

Lasso regression's test MSE is slightly higher than Ridge's, so we move forward using Ridge regression model.

Regression Tree

Recursive Binary Splitting

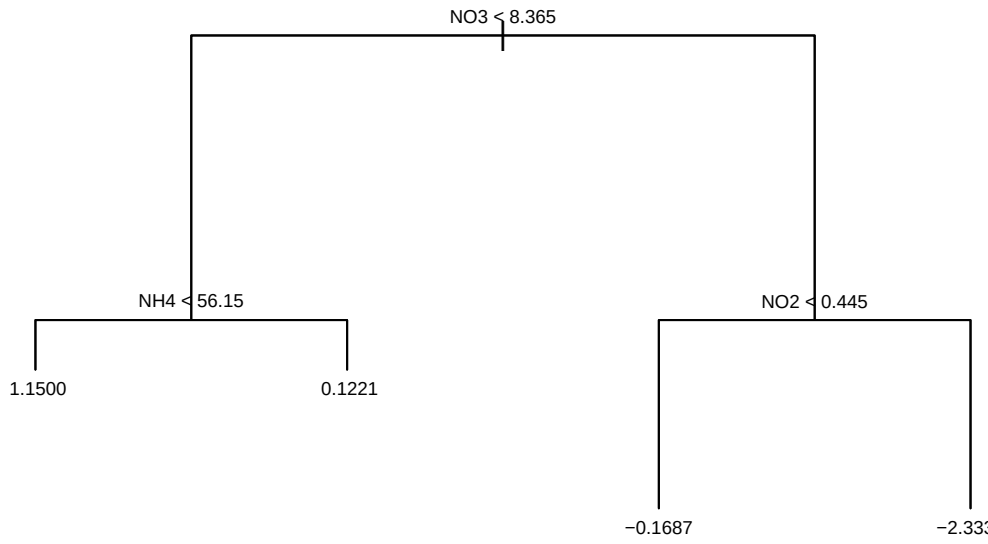
```
tree.reg <- tree(log(AOM)~., data=train_line)
summary(tree.reg)
```

Model Summary

```
##
## Regression tree:
## tree(formula = log(AOM) ~ ., data = train_line)
## Variables actually used in tree construction:
## [1] "NO3" "NH4" "NO2"
## Number of terminal nodes:  4
## Residual mean deviance:  1.914 = 34.44 / 18
## Distribution of residuals:
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -1.73200 -1.26600  0.04811  0.00000  0.97630  2.55900
```

NO3, NH4, NO2 are retained in our tree model, with 4 terminal nodes.

```
plot(tree.reg)
text(tree.reg, cex=0.6)
```



Graphical Summary
NO3 seems to be the most important variable in our recursive binary tree, then NH4 and NO2.

```
pred.tree.reg <- predict(tree.reg, newdata=test_line)
mse.tree.reg <- mean((log(test_line$AOM)-pred.tree.reg)^2)
mse.tree.reg
```

Test MSE

```
## [1] 7.344205
```

The test MSE is around 7, which does not perform better than Ridge regression. However, we will try out some optimization methods.

Pruning

```
set.seed(4996)
cv.class.reg<-tree::cv.tree(tree.reg, K=10)
trees.num.class.reg<-cv.class.reg$size[which.min(cv.class.reg$dev)]
trees.num.class.reg
```

```
## [1] 1
```

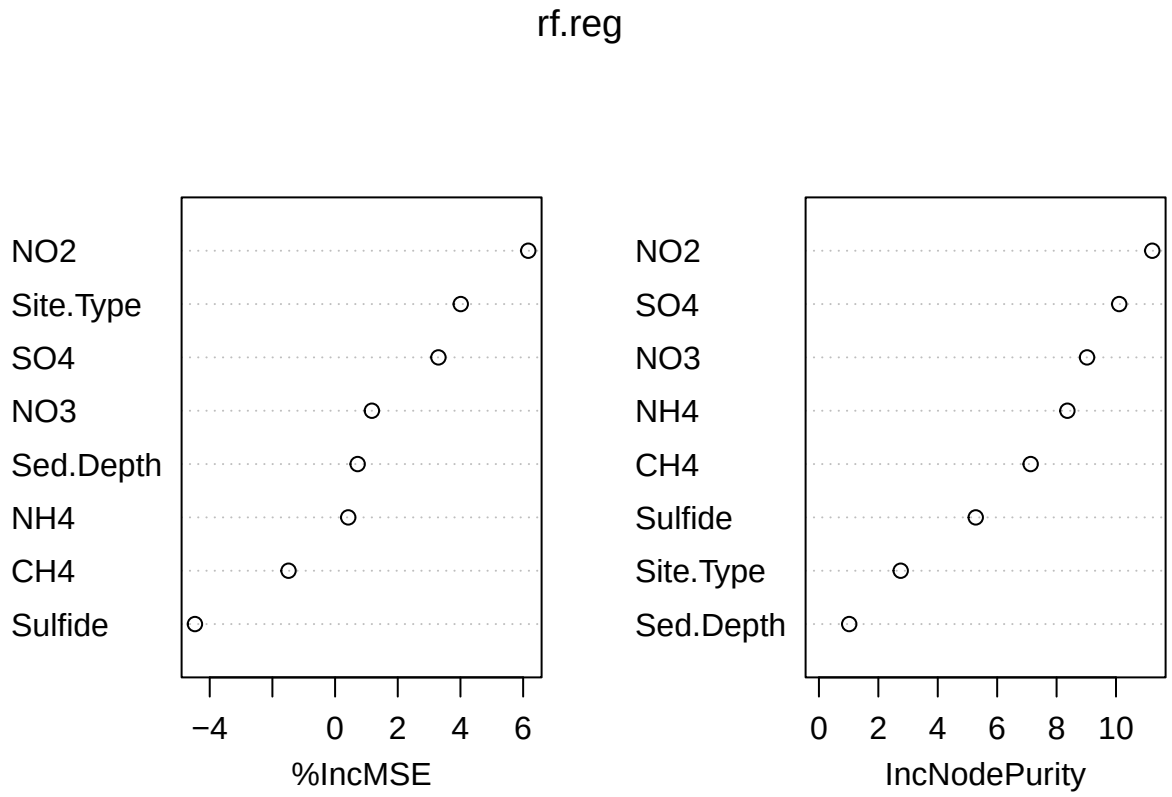
It seems like we are pruning the tree to retain only one terminal node, which is not desirable so we would not continue with that.

Random Forest

```
set.seed(4996)
size <- floor((ncol(train_line)-1)/3)
rf.reg <- randomForest::randomForest(log(AOM)~., data=train_line,mtry=size, importance=TRUE)
randomForest::importance(rf.reg)
```

```
##              %IncMSE IncNodePurity
## Sed.Depth  0.7199456      1.019730
## Site.Type  4.0089635      2.752327
## CH4       -1.4852176      7.127112
## Sulfide   -4.4718600      5.277682
## SO4        3.2975003     10.108069
## NO3        1.1742230      9.024898
## NO2        6.1636287     11.220406
## NH4        0.4200047      8.363384
```

```
randomForest::varImpPlot(rf.reg)
```



The random forest model selects NO2, NO3, and SO4 as the important predictors, which is slightly different from what we observed in the recursive binary tree model.

```
pred.rf.reg <- predict(rf.reg, newdata=test_line)
mse.rf.reg <- mean((log(test_line$AOM)-pred.rf.reg)^2)
mse.rf.reg
```

Test MSE

```
## [1] 4.821596
```

The MSE drops drastically, close to Ridge regression's performance.

Conclusion

```
reg.matrix <- data.frame(linear=mse.reg,
                         ridge=mse.ridge,
                         lasso=mse.lasso,
                         tree=mse.tree.reg,
                         random.forest=mse.rf.reg)
reg.matrix
```

```
##     linear    ridge    lasso     tree random.forest
## 1 9.808248 5.034566 5.781116 7.344205      4.821596
```

Ridge and random forest have the lowest MSE, which means they both are the most effective in predicting AOM level. However, since they each select different sets of variables in the model, we do not know for sure which can be more effective. This is partially due to the massive variation contained in our data set.

Final Remarks

To recap, we first explored various classification models and found random forest to be the best in predicting AOM presence. Then, we use that model to predict on our data and find the corresponding observations. Using these data points, we fit regression models and eventually find ridge regression and random forest regression model to be the best at predicting AOM level.

There are challenges involved, such as cleaning the data due to various levels contained in certain variables and figuring how to deal with the outliers. Also, under complex real-life data sets like the one we use, comparing between vast statistical models becomes increasingly crucial.