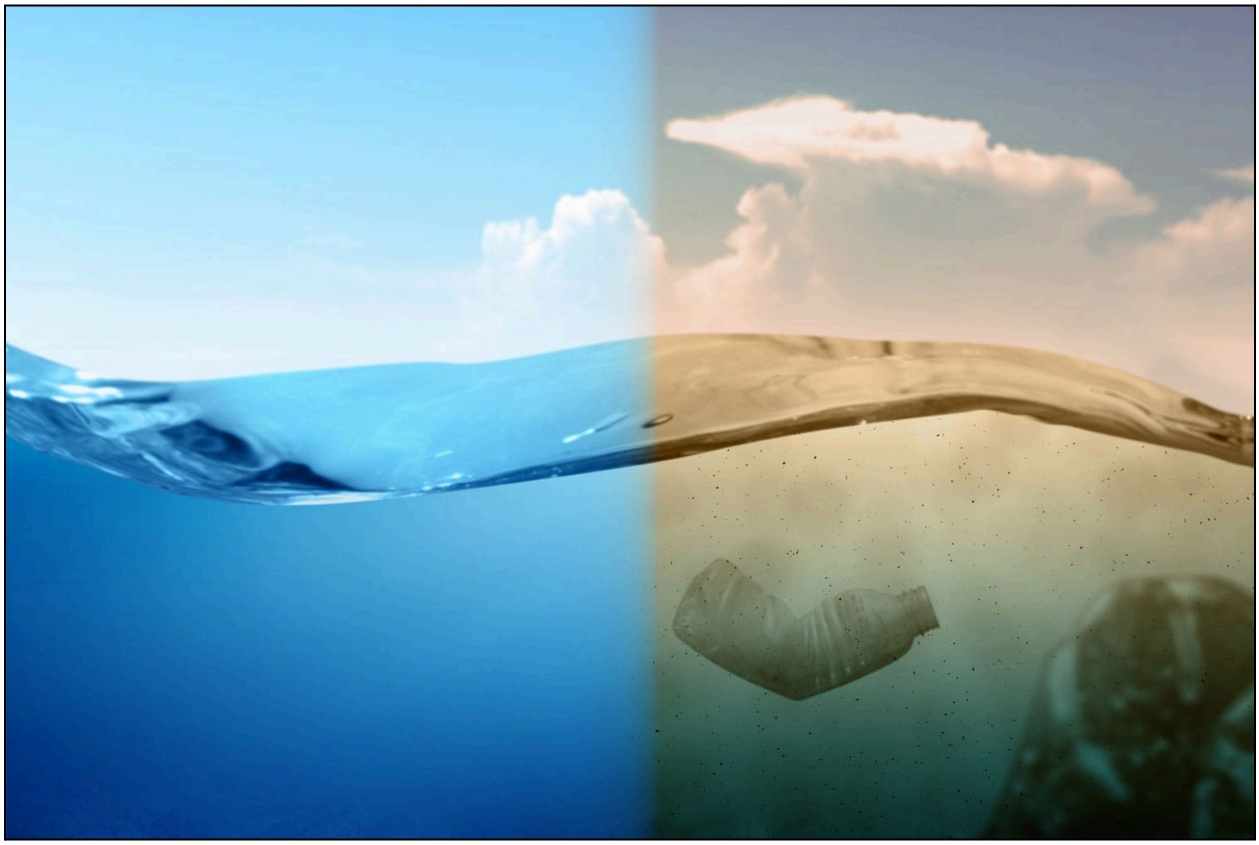
Key Determinants of Water Potability

Group 25:

Maximillian Chalitsios, Alfredo Ruiz, Zahra Bukhari, Meenakshi Vinod, Abdul Rafay Khan

Spring 2024



Introduction

Context

Water potability, a category defined as the safety of water for human consumption, ensures that water meets established safety standards and is free from contamination. According to the Water Education Foundation, contaminated water can pose significant health risks, making it crucial to ensure water potability to prevent the spread of diseases.

Goal

For this project, we are using the *Water Quality* dataset from kaggle which contains 2011 observations and 10 different variables. By comparing the results of Multivariate Logistic Regression and Random Forest models, the goal is to accurately answer:

Which variables are most significant in determining water potability?

Description of Variables

Response

- *Potability*: 1 is safe for human consumption, 0 is not safe.

Predictors

- *pH*: pH of 1. water (0 to 14).
- *Hardness*: Capacity of water to precipitate soap in mg/L.
- *Solids*: Total dissolved solids in ppm.
- *Chloramines*: Amount of Chloramines in ppm.
- *Sulfate*: Amount of Sulfates dissolved in mg/L.
- *Conductivity*: Electrical conductivity of water in µS/cm.
- *Organic_carbon*: Amount of organic carbon in ppm.
- *Trihalomethanes*: Amount of Trihalomethanes in µg/L.
- *Turbidity*: Measure of light emitting property of water in NTU.

Models and Methods

Data Cleaning

The first task was to clean our data. We used Mean Imputation to replace the NULL or N/A values with the mean of that column. Using R to do this with the following code:

```
install.packages("dplyr")
library(dplyr)
```

```
> water_potability <- dplyr::mutate(water_potability, dplyr::across(where(is.numeric), ~ifelse(is.na(.), mean(., na.rm = TRUE), .)))
> head(water_potability)
# A tibble: 6 × 10
     ph Hardness Solids Chloramines Sulfate Conductivity Organic_carbon Trihalomethanes Turbidity Potability
  <dbl>    <dbl>  <dbl>       <dbl>   <dbl>        <dbl>          <dbl>           <dbl>     <dbl> <fct>
1  7.08     205. 20791.        7.30    369.         564.           10.4            87.0      2.96 0
2  3.72     129. 18630.        6.64    334.         593.           15.2            56.3      4.50 0
```

This cleaned up our data to ensure both models used the same data for training and testing.

Logistic Regression **(Alfredo, Abdul Rafay, Maximillian)**

Model formula:

$$p[x] = \frac{exp^{B_0+B_1X_1+\ldots+B_nX_n}}{1 + exp^{B_0+B_1X_1+\ldots+B_nX_n}}$$

The Logistic Regression Model was chosen due to the fact that they are computationally inexpensive to train, which makes it viable for problems where computational scalability and efficiency are required. Logistic regression is specifically designed for binary classification tasks. It models the logit-transformed probability as a linear relationship with the predictor variables, making it a natural choice for problems with binary outcomes. To fit the model, we did some data preprocessing by splitting the data: 80% of the data in the training set and 20% of the data in the testing set as shown below:

```
> library(caret)
Loading required package: ggplot2
Loading required package: lattice
> splitIndex <- createDataPartition(water_potability$Potability, p = 0.8, list = FALSE)
> train <- water_potability[splitIndex, ]
> test <- water_potability[-splitIndex, ]
```

This is where we fit the model:

```
> water_potability_model <- glm(Potability ~ ., data = train_set, family = binomial)
```

And this was the resulting summary:

```
> summary(water_potability_model)

Call:
glm(formula = Potability ~ ., family = binomial, data = train_set)

Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)     -0.4482088  0.0401351 -11.167   <2e-16 ***
ph               0.0435476  0.0409537   1.063   0.2876
Hardness        -0.0276777  0.0405107  -0.683   0.4945
Solids           0.0955571  0.0405458   2.357   0.0184 *
Chloramines      0.0518379  0.0404616   1.281   0.2001
Sulfate         -0.0009098  0.0405768  -0.022   0.9821
Conductivity    -0.0057848  0.0400520  -0.144   0.8852
Organic_carbon  -0.0449807  0.0399525  -1.126   0.2602
Trihalomethanes  0.0107402  0.0402420   0.267   0.7896
Turbidity        0.0157270  0.0399686   0.393   0.6940
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 3507.3  on 2621  degrees of freedom
Residual deviance: 3497.7  on 2612  degrees of freedom
AIC: 3517.7
```

We can quickly see that we don’t have much significance amongst our predictors, with Solids showing some significance. We also notice that our AIC is very high as well, which correlates that our model's predictors thus far don’t have enough significance in predicting potability. We noticed that the dataset we selected had already separated our findings between those that are Not Potable, and those Potable indicated with a 0 or 1. We decided to shuffle our dataset to potentially remove any bias in our training/testing split of 80/20.

```
> set.seed(123)
> shuffled_data <- water_potability[sample(1:nrow(water_potability)), ]
> train_index <- sample(1:nrow(shuffled_data), size = floor(0.8 * nrow(shuffled_data)))
>
> train_set <- shuffled_data[train_index, ]
> test_set <- shuffled_data[-train_index, ]
> water_potability_model <- glm(Potability ~ ., data = train_set, family = binomial)
```

After utilizing our shuffled data to recreate the water potability model, we got the following summary:

```
> summary(water_potability_model)

Call:
glm(formula = Potability ~ ., family = binomial, data = train_set)

Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)     -4.504e-01  6.822e-01  -0.660   0.5091
ph              -2.444e-03  2.706e-02  -0.090   0.9280
Hardness        -6.982e-04  1.224e-03  -0.570   0.5685
Solids           8.253e-06  4.601e-06   1.794   0.0729 .
Chloramines      2.186e-02  2.539e-02   0.861   0.3893
Sulfate         -2.324e-06  1.124e-03  -0.002   0.9984
Conductivity    -1.962e-05  4.927e-04  -0.040   0.9682
Organic_carbon  -2.244e-02  1.234e-02  -1.818   0.0691 .
Trihalomethanes  7.931e-04  2.551e-03   0.311   0.7559
Turbidity        2.480e-02  5.165e-02   0.480   0.6312
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 3506.2  on 2619  degrees of freedom
Residual deviance: 3498.0  on 2610  degrees of freedom
AIC: 3518

Number of Fisher Scoring iterations: 4
```

From the new summary, we can see that there's still some significance between the Solids, albeit its p-value went up which reduced the significance it had. We also see that the Organic Carbon predictor demonstrates some significance this time as well. With this, we also see that our AIC decreased, suggesting our model is still well away from being good at

predicting water potability. In order to see if we can improve our AIC, we decided to refit our model to just use the predictors that showed significance, and thus we got the summary below:

```
> water_potability_model <- glm(Potability ~ Solids + Organic_carbon, data = train_set, family = binomial)
> summary(water_potability_model)

Call:
glm(formula = Potability ~ Solids + Organic_carbon, family = binomial,
    data = train_set)

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)    -2.979e-01  2.044e-01  -1.458   0.1449
Solids          8.132e-06  4.498e-06   1.808   0.0706 .
Organic_carbon -2.291e-02  1.230e-02  -1.862   0.0626 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 3506.2  on 2619  degrees of freedom
Residual deviance: 3499.5  on 2617  degrees of freedom
AIC: 3505.5

Number of Fisher Scoring iterations: 4
```

While our AIC did improve, it was a very marginal improvement from what we had previously. We also generated a confusion matrix for this last model. Still, the results were largely disappointing, and only compounded by the fact that our model is not equipped with the predictors needed to predict water potability.

While we tried our best with many different approaches (going even beyond what we've shared here), we've come to realize our dataset had some issues in being what we had hoped would be excellent data to create a water potability prediction model. Despite that, this was a huge learning experience for us since we got to see firsthand what may happen with a real-world dataset. We also realize that our question could've been potentially more related to predicting levels of specific contaminants as that may have been a potential question for this problem specifically.

Random Forest Model **(Meenakshi, Zahra)**

The Random Forest Model was chosen for this analysis due to its ability to handle both numeric and categorical predictors, making it well-suited for datasets with a mix of variable types like the *Water Quality* dataset. It also excels in capturing complex relationships and interactions between predictors, with feature importance scores, which is essential in understanding the factors influencing water potability.

Additionally, the Random Forest Model is robust to overfitting, as it builds multiple decision trees and combines their predictions, reducing the risk of modeling noise or outliers. However, the Random Forest Model can be computationally intensive, especially with large datasets or many predictors, which may increase training time.

Interpretability of the model can also be a challenge, as the combination of multiple trees makes it less straightforward to interpret individual tree decisions compared to simpler models like linear regression. Despite these drawbacks, the strengths of Random Forest Model in handling complex data structures and providing robust predictions make it a suitable choice for this analysis.

Model formula:
Potability ~ pH + Hardness + Solids + Chloramines + Sulfate + Conductivity + Organic_Carbon + Trihalomethanes + Turbidity

We first made our preliminary Random Forest Model with all 10 variables as seen below.

```
water_data$Potability <- as.factor(water_data$Potability)

library(randomForest)

# Preliminary Random Forest Model with all variables
preliminary.RFModel <- randomForest(Potability ~ ph + Hardness + Solids
                                    + Chloramines + Sulfate + Conductivity
                                    + Organic_carbon + Trihalomethanes
                                    + Turbidity
                                    , data = water_data,
                                    mtry = sqrt(9), importance = TRUE)

preliminary.RFModel
```

Output:

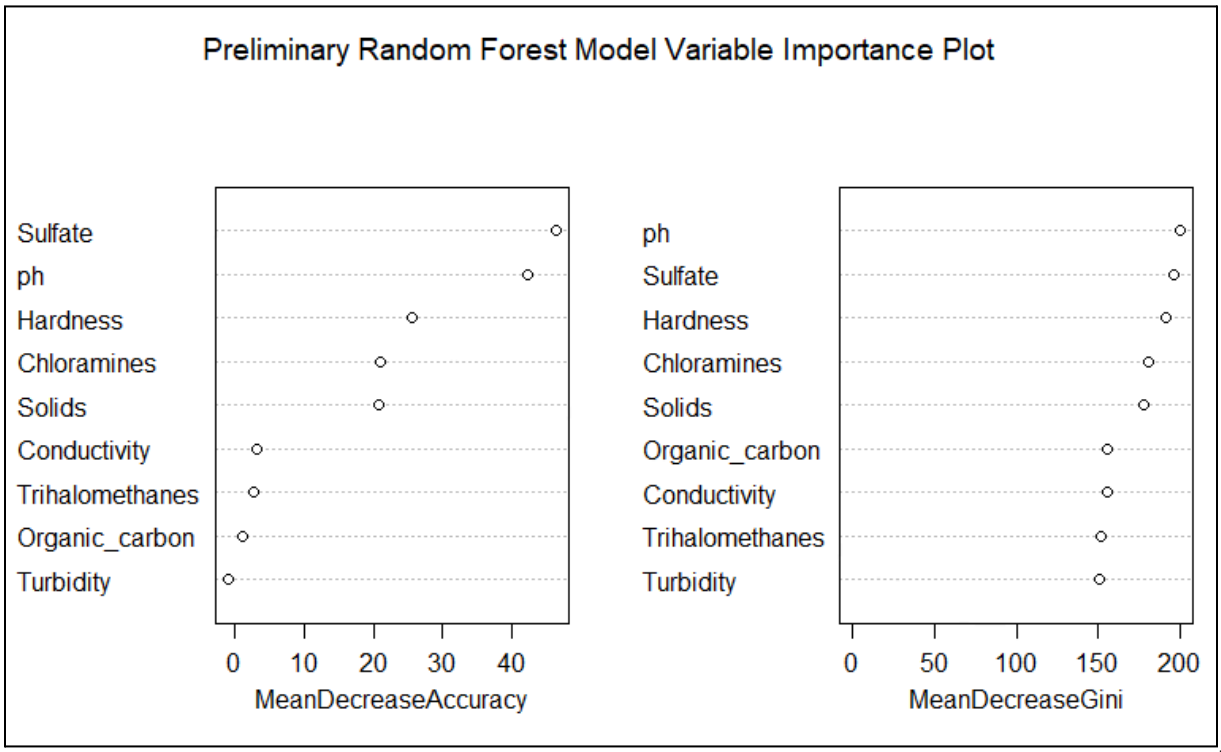
```
# Call:
# randomForest(formula = Potability ~ ph + Hardness + Solids +
#                                     Chloramines + Sulfate +
#                                     Conductivity + Organic_carbon +
#                                     Trihalomethanes + Turbidity,
#                                     data = water_data, mtry = sqrt(10)
#                                     , importance = TRUE)
# Type of random forest: classification
# Number of trees: 500
# No. of variables tried at each split: 3

# OOB estimate of  error rate: 32.23%
# Confusion matrix:
#      0   1 class.error
# 0 1758 240   0.1201201
# 1  816 462   0.6384977
```

The preliminary model has 500 trees and at each split 3 variables are tried. We have an OOB(Out of Bag) estimate error of 32.23 percent. The OOB is calculated by using samples that were not included in the training of the individual trees in order to estimate the model's performance on unseen data.

We then plotted a variable importance plot to visualize the importance of each variable in determining water potability. This plot helps us identify which variables have the most significant impact on the model's performance.

```
varImpPlot(preliminary.RFModel,main="Preliminary Random Forest Model Variable Importance Plot")
```



The MeanDecreaseAccuracy plot shows that Sulfate and pH are the most important variables, as their removal would significantly decrease the prediction accuracy. On the other hand, variables like Conductivity, Trihalomethanes, Organic_carbon, and Turbidity have a relatively low impact on prediction accuracy.

In the MeanDecreaseGini plot, pH, Sulfate, and Hardness appear to be the most important variables, as they have the highest values, indicating that splitting on these variables would result in the greatest decrease in node impurity.

While the MeanDecreaseAccuracy plot suggests that Conductivity, Trihalomethanes, Organic_carbon, and Turbidity have minimal impact, the MeanDecreaseGini plot indicates that none of the variables can be considered completely insignificant.

Based on the information provided in the plots, it can be inferred that pH and Sulfate are the most significant variables for the Random Forest Model, as they are important in both MeanDecreaseAccuracy and MeanDecreaseGini. Hardness, while not as important for prediction accuracy, contributes to reducing node impurity. The remaining variables, although less influential, cannot be excluded from the model based on the provided information.

Therefore, we proceeded to train the model with a randomized 80:20 split, 80% training, and 20% testing, with *all* predictors in the model formula.

```
# Set the seed for reproducibility
set.seed(123)

# Create a vector to store error rates
error.rate = rep(0, 10)
```

```
# Perform 10-fold cross-validation
for(i in 1:10) {
  set.seed(i)

# Create indices for the training set
index = sample.int(n = nrow(water_data),
                   size = floor(0.8 * nrow(water_data)),
                   replace = FALSE)

# Create the training and test sets
train = water_data[index, ]
test = water_data[-index, ]

# Build the random forest model
result.rf = randomForest(Potability ~ ph + Hardness + Solids +
                         Chloramines + Sulfate + Conductivity
                         + Organic_carbon + Trihalomethanes + Turbidity,
                         data = train,
                         mtry = sqrt(10),
                         importance = TRUE)

# Make predictions on the test set
yhat.rf = predict(result.rf, newdata = test)

# Create confusion matrix
con.matrix = table(yhat.rf, test$Potability)

# Calculate error rate
error.rate[i] = (con.matrix[1, 2] + con.matrix[2, 1]) / sum(con.matrix)
}
```

```
# Print error rates
error.rate
# 0.3262195 0.3185976 0.3109756 0.3094512 0.3521341
# 0.3003049 0.3048780 0.3246951 0.3125000 0.3216463

# Print mean error rate
mean(error.rate)
# 0.3181402
```

We got an average error rate of 31.81 percent across 10 different iterations of training and testing, ranging from 30.03 to 35.21 percent.

Results **(All group members)**

As shown in our findings, the logistic regression model we created didn't provide any conclusive results as far as being able to predict water potability. Most of our predictors showed no significance, and the ones that did show any significance were very weak. Overall, we observed that our logistic regression model was not equipped with the necessary predictors for answering which factors were key determinants of water potability.

The Random Forest Model was similarly disappointing, though it still provided a bit better results than the logistic regression model. Random Forests provide more flexibility, and can capture complex relationships within the data, ultimately this was shown in the results compared to the logistic regression model. Through feature selection, we found that the strongest variables were ph, sulfate, and hardness, but we did not have enough information to suggest that the other variables were completely insignificant.

In conclusion, neither model provided strong insights into the key determinants of water potability. We suspect that the dataset might have been flawed, or the inclusion of additional variables would help build more accurate predictive models.

Bibliography (References)

Kadiwal, A. (2021, April 25). *Water quality*. Kaggle.

https://www.kaggle.com/datasets/adityakadiwal/water-potability

Potable water. Water Education Foundation. (2023).

https://www.watereducation.org/aquapedia-background/potable-water

Radečić, D. (n.d.). *Imputation in R: Top 3 ways for imputing missing data*. Appsilon.

https://www.appsilon.com/post/imputation-in-r

Professional Product

GitHub Repository