Prateek Kakkar, Nitharsan Sivakanthan, Liya LaPierre

DATA 5322 Final Project Report

June 8, 2022

Classification of Water Potability

Abstract

Human survival depends on the availability of water. Safe drinking water is both a basic need for good health and a basic human right. In many places of the world, fresh water is already in short supply. The availability of safe drinking water and sanitation is a global issue. Does the potability of water depend on the pH value, turbidity, hardness, or sediment present? Using the data from the Kaggle (Kadiwal, n.d.), this research demonstrates the use of unsupervised and supervised machine learning to determine the factors affecting the quality of water. The first model using logistic regression was able to classify the potability of water with an error rate of 39%. The models using Random Forest were able to classify potability with an error rate of 32.3% and 41.2%. The models using KNN resulted in test errors of 35.1% and 34.5%.

Introduction

Determining the potability of potential sources of drinking water is an essential tool for public health. However, there are many potential sources of contamination which necessitates the need for tests that measure many different things. Given that this is not always feasible due to cost, lack of access, etc., it is important to determine whether potability can be determined through a reduced set of measurements. To make this determination, we will be testing whether potability can be accurately predicted by various measurements of water quality using machine learning algorithms, and whether the models can be just as accurate with less measurements.

The dataset we are using was posted on the website Kaggle (Kadiwal, n.d.), and contains data on several measures of water quality for 3276 different bodies of water, and labels for whether the water is potable or not. There are measurements for pH, hardness (amount of dissolved calcium and magnesium), total dissolved solids (TDS), chloramines, sulfate, conductivity, total organic carbon (TOC), trihalomethanes (THMs), and turbidity (measures the amount of solids in water using light passing through). Several of these have desirable ranges for drinking water and are used to determine whether drinking water is potable (Environmental Protection Agency, n.d.). Using this data, we will be using machine learning algorithms to prepare the data and classify the potability of water using the various water quality measurements.

Background

Singular Value Decomposition (SVD)

One method we will be using to address missing values is called *singular value decomposition* (SVD). SVD is a matrix decomposition method in machine learning for reducing a matrix to make subsequent matrix calculations simpler. It factorizes a matrix which then decomposes it into three generic and familiar matrices. For example, a matrix A can be written as A=USV^T as described in Figure 1. Here we call the vectors in U the left singular vectors while the vectors in V the right singular vectors and diagonal values in the S (Sigma) matrix are known as singular values. The goal behind this method is that Matrix A transforms a set of orthogonal vectors V to another set of orthogonal vectors U with a scaling factor of s. So, s is called the singular value corresponding to the respective singular vectors U and V. It only works with numeric values and is effective in imputing the missing values using the underlying data correlation structures.

$$\underset{m \times n}{\overset{\mathrm{A}}{\begin{pmatrix} x_{11} & x_{12} & & x_{1n} \\ & & \ddots & \\ x_{m1} & & & x_{mn} \end{pmatrix}}} = \underset{m \times m}{\overset{\mathrm{U}}{\begin{pmatrix} u_{11} & & u_{m1} \\ & \ddots & \\ u_{1m} & & u_{mm} \end{pmatrix}}} \underset{m \times n}{\overset{\mathrm{S}}{\begin{pmatrix} \sigma_1 & & 0 \\ & \ddots \; \sigma_r \; \ddots & \\ 0 & & 0 \end{pmatrix}}} \underset{n \times n}{\overset{\mathrm{V^T}}{\begin{pmatrix} v_{11} & & v_{1n} \\ & \ddots & \\ v_{n1} & & v_{nn} \end{pmatrix}}}$$

Figure 1. SVD Matrix Decomposition. Source: (Hui, 2019)

Principal Component Analysis (PCA)

We also made use of principal component analysis (PCA) to discover any trends or important predictors that might allow us to reduce the number of predictors in any of our other models. Practically, this would be extremely helpful for communities that may not have the resources to perform all the tests corresponding to our predictors. PCA is a dimensionality reduction technique that allows us to approximate data by creating principal components that best describe that data. PCA computes principal components, Zn, using the following linear combination.

$$Z_n = \phi_{1n}x_1 + \phi_{2n}x_2 + \cdots + \phi_{p_n}x_P$$

Each $\phi\rho$ is a linear combination of features p that correspond to the direction of greatest variance and the observations x. Each principal component now gives us the location of observations in a reduced dimensional space. The principal components have a hierarchical ordering where the first principal component provides us with the direction of the greatest variance, and each following principal component provides the next greatest direction of variance orthogonal to the first principal component. We can use a small number of principal components that account for much of the variance in the data to understand and approximate the relationships present between our predictors. All that is needed to perform PCA is a completed matrix or sparse matrix where each column of the matrix is the relevant predictor variables, and each row of the matrix is a single observation.

Logistic Regression

We wanted to classify water portability based on all the features present in the data set. Logistic Regression seems an appropriate approach in problems where we need to predict the likelihood of events by looking at the prior observation of the data set. This method results in an extreme binary outcome with a logarithmic line distinguishing them. The main advantage of using this model is that it is less prone to overfitting in low dimensional datasets and is generally fast at classifying the unknown records. Without assuming distributions of classes, it provides inference about the importance of each feature. For binary classification, this model calculates the conditional probability P (Y|X) of the dependent variable Y, given independent variable(s) X. Where P (Y|X) is a sigmoid function applied to a linear combination of input features. As a result, it produces a logistic curve, where values are limited between 0 and 1, using the log-odds function as shown in figure 2.

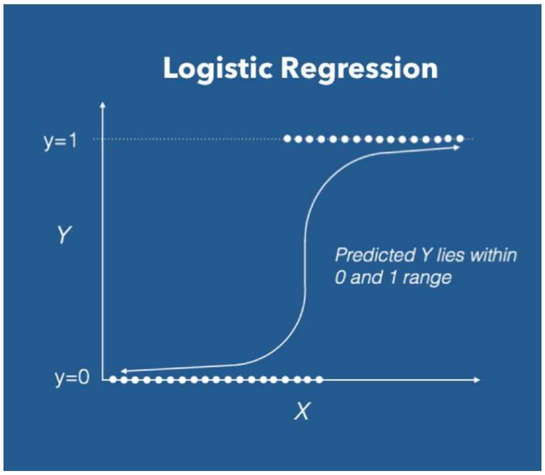


Figure 2. Logistic Regression. Source: (Rajput, 2018)

Random Forest Classification

Random Forest Classification is a decision tree method for classification. Random forest takes an ensemble approach to decision trees. This means the algorithm creates many decision trees which it considers when developing the best decision tree. The unique part of the random forest algorithm is the way it creates its ensembles. Each tree is created by taking a sample of m of the possible p predictors. This approach allows the algorithm to search for more possible trees compared to other approaches that may only consider the best predictor at each step of the tree. To run the random forest classification algorithm, we must provide the predictor variables, the response variables, and the number m of the possible predictors to sample for each tree.

K-Nearest Neighbors Classification

We also wanted to explore a classification algorithm which works off the assumption that similar points can be found near one another. With our comparatively small and properly labelled dataset, KNN seems to be a wise pick. K-Nearest Neighbors is a non-parametric, supervised learning classifier, which uses proximity to make classifications or predictions about the grouping of an individual data point. It standardizes each variable to prevent different units of variables from influencing the results. It is one of the easiest to implement with only one parameter to tune. Here, the parameter K needs to be tuned to get the best results. K is the number of nearest neighbors to be considered to classify the new data points. It generally uses Euclidian distance, but other methods can also be configured to calculate the smallest distance.

Methodology

Data Preparation

To prepare the data for use in our models, we needed to address missing values; many of the bodies of water were not tested for pH, sulfates, or THMs. We were able to impute the missing values through SVD. Now that the matrix was complete, we were able to run PCA on the data to produce a lower dimension version.

Models

For each model we tested, we trained the model on the data with imputed missing values. First, we split the data so that 80% could be used to train our models, and the other 20% was reserved to test model accuracy. We then trained a logistic regression model to predict water potability and tested the model's accuracy with our reserved test data. Next, we used Random Forest classifiers to predict water potability. The Random Forest classifiers were tuned to find the best parameter values. We tuned on the number of m predictors to sample, as well as the number of trees created. The first Random Forest model used all the predictors, and we also created a limited model with only pH, sulfate, and chloramines as predictors. Again, we tested these models on our reserved training data. Finally, we trained a K-Nearest Neighbors model to do the same classification problem as the others. We first used cross-validation to determine the value of k that had the highest accuracy and then trained a model using that value.

Results

PCA

We conducted PCA in order to see if the data could be reduced in dimensionality, and to determine which variables account for greater variability in the dataset. The results of the PCA indicate that we would need 8 principal components to capture at least 90% of the variation in the data, which is only slightly lower dimension than the dataset itself with 9 variables. Therefore, we decided not to use the principal components when training models. However, we did learn that the first principal component (the component accounting for the most variability in the dataset) had high weightings for the pH, sulfate, and conductivity variables.

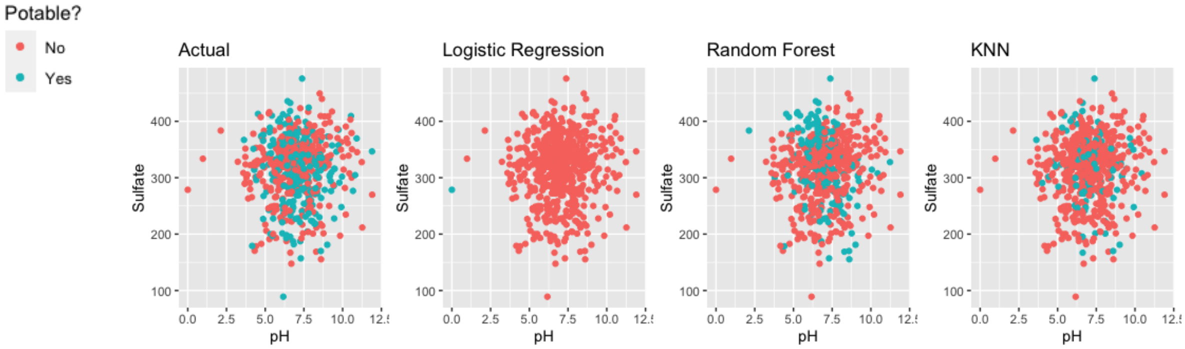


Figure 3. Scatterplots for Actual Potability and Predicted Potability.

Classification- Logistic Regression

First, we trained a logistic regression model to predict water potability. Only solids was significantly predictive of potability, *p* = 0.047. The model had a test error rate of 38.90% or an accuracy of 61.10%. As you can see in the second plot of Figure 3 above and the confusion matrix in Figure 4 below, the logistic regression classifier mostly predicted the samples to be not potable, except for one point. However, because it is classifying almost everything as non-potable, the specificity for this model is extremely high (99.8%). However, because it is almost exclusively predicting samples as not potable, it's not a very useful model because we do want to identify some sources of drinking water.

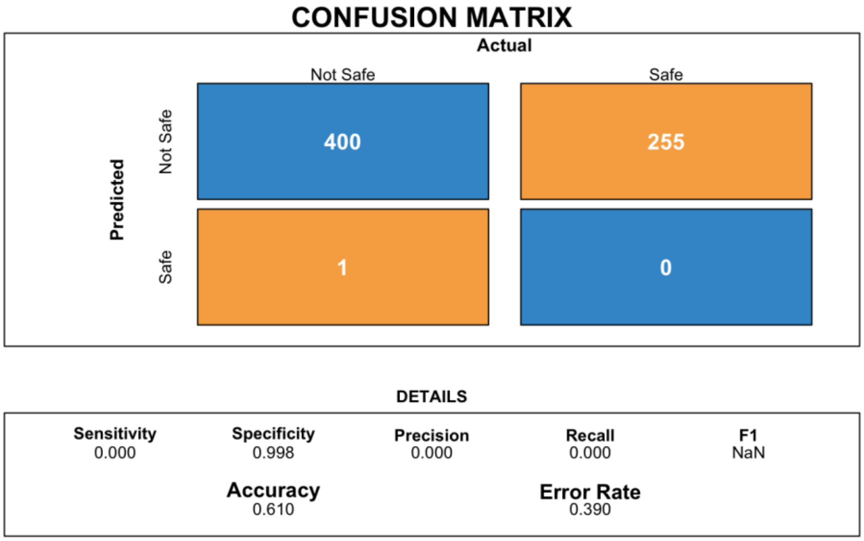


Figure 4. Confusion Matrix and Performance Statistics for Logistic Regression

Classification- Random Forest Classifier

To find the ideal values for parameters of our Random Forest classifiers, we tuned the models for the values of mtry (number of variables selected for each tree) and ntrees (number of trees). The results for the tuning can be seen in Table 1. The best model used mtry of 4 and ntrees of 300. The confusion matrix for the predictions of the best model on the testing data can be seen in Figure 5. As we can see, the accuracy for this model is higher than the logistic regression model (67.7%). Additionally, as you can see in the third plot of Figure 3, this model better resembles the actual data in its classifications compared with the logistic regression model. In this context, we would like to be able to perform well with predicting whether water is not potable. The measure of specificity shows that for approximately 89% of the samples that were not potable, this model correctly predicted it to be not potable.

mtry	ntrees	error
2	300	0.34389
3	300	0.33855
4	300	0.33092
2	400	0.33626
3	400	0.34504
4	400	0.33550
2	500	0.33893
3	500	0.33664
4	500	0.34046

Table 1. Tuning Results for Random Forest Classifier.

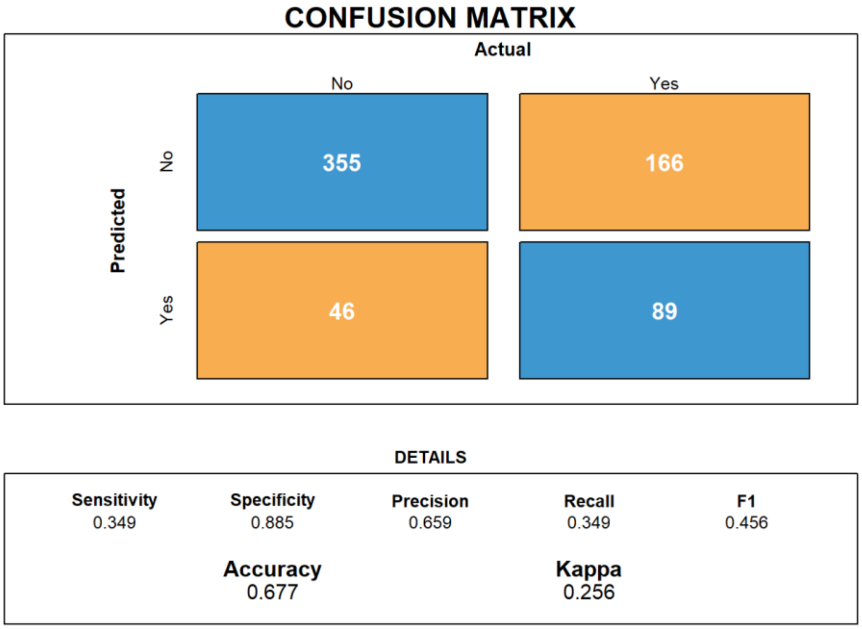
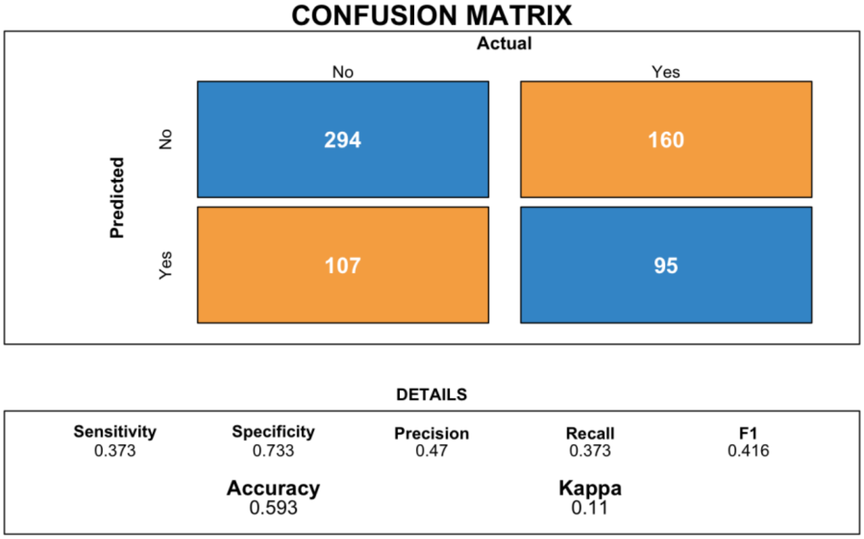


Figure 5. Confusion Matrix and Performance Statistics for Random Forest Algorithm #1

We also want to the accuracy of predicting potability using the most important variables found from PCA. Next, we trained the random forest classifier only using pH, Sulfate, and Conductivity as predictors. The confusion matrix and performance statistics for the results of the best model on the testing data can be seen in Figure 6. This model performs poorer compared to the model with all predictors, however it still has a high specificity.



CONFUSION MATRIX

Predicted \ Actual	No	Yes
No	294	160
Yes	107	95

DETAILS

Sensitivity	Specificity	Precision	Recall	F1
0.373	0.733	0.47	0.373	0.416

Accuracy	Kappa
0.593	0.11

Figure 6. Confusion Matrix and Performance Statistics for Random Forest Algorithm #2

Classification – K-Nearest Neighbors

For our KNN model, we tuned the model with respect to the value of k. We found that the model with the highest cross-validated accuracy had a k value of 21 (see Figure 7 below). The test accuracy for this model was 64.94%, and therefore a test error of 35.06%. As you can see in Figure 8, we obtained higher specificity with this model compared to the others, accurately classifying 90.5% of the non-potable samples in the dataset.

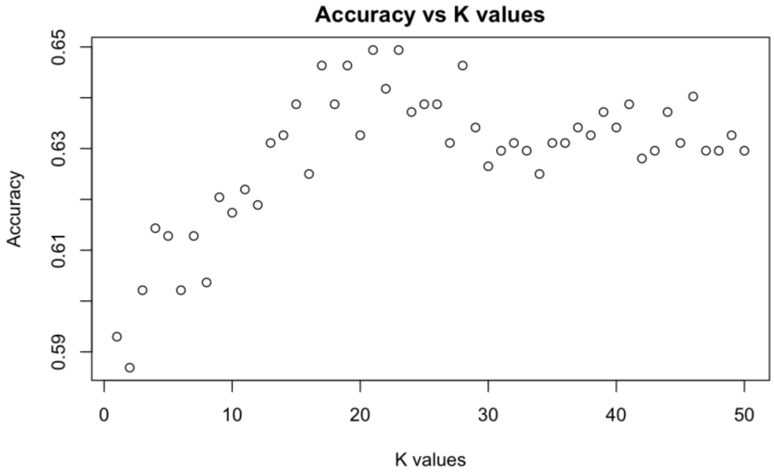
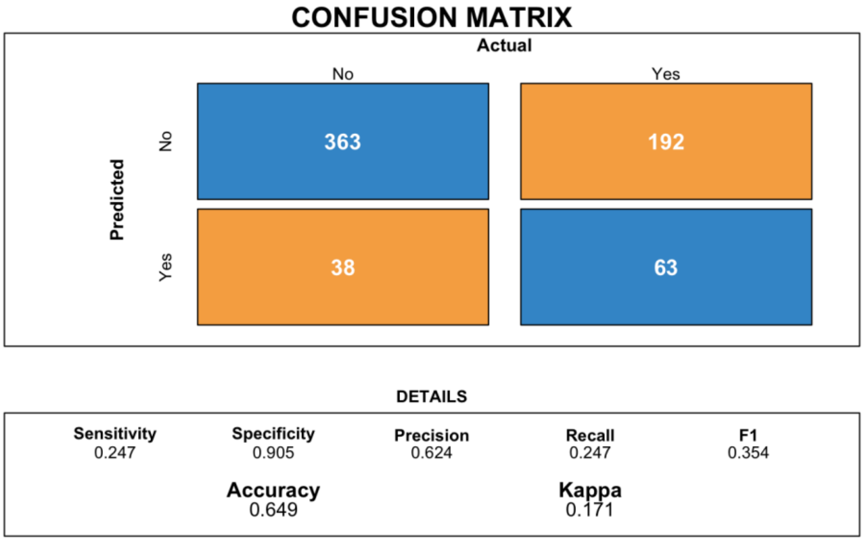


Figure 7. KNN Cross-Validated Accuracy by K Value.



CONFUSION MATRIX

		Actual	
		No	Yes
Predicted	No	363	192
	Yes	38	63

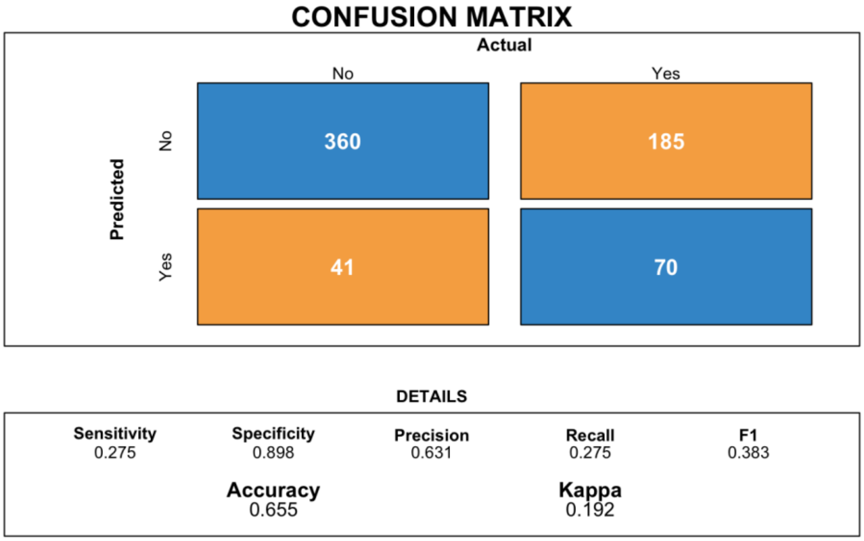
DETAILS

Sensitivity	Specificity	Precision	Recall	F1
0.247	0.905	0.624	0.247	0.354

Accuracy	Kappa
0.649	0.171

Figure 8. Confusion Matrix and Performance Statistics for K-Nearest Neighbors Classifier #1

Next, we will train KNN using fewer predictors. Again, we choose the predictors that appeared most important from the PCA analysis, pH, Sulfate, and Conductivity. See Figure 9 for a confusion matrix of the results of the best model on the testing data. This model provides us with very good results considering we only use three predictors. In terms of specificity, it outperforms the best random forest classifier model and is nearly equivalent to the KNN model with all predictors.



CONFUSION MATRIX

		Actual	
		No	Yes
Predicted	No	360	185
	Yes	41	70

DETAILS

Sensitivity	Specificity	Precision	Recall	F1
0.275	0.898	0.631	0.275	0.383

Accuracy	Kappa
0.655	0.192

Figure 9. Confusion Matrix and Performance Statistics for K-Nearest Neighbors Classifier #2

Discussion

The results of this research show that we can predict, with moderate accuracy, the potability of water using several machine learning algorithms. Of the algorithms tested, we found that K-Nearest Neighbors performed the best at accurately predicting non-potable water samples. The KNN model with all the predictors had a testing specificity rate of 90.5%, and the KNN model with only three predictors had a slightly lower specificity of 89.8%. This shows that these KNN models were very accurate at identifying the samples that were not potable, which is what is important from a public health perspective. Additionally, the models with only three of the nine water quality measurements as predictors still performed very similarly to the full models, suggesting that we could reduce the number of tests done on a water sample and still be able to determine whether it is potable or not.

It is unclear how "potability" was determined for this dataset. Drinking water potability (in the United States) is determined by levels of many more contaminants than what is contained in our dataset (Environmental Protection Agency, n.d.). For instance, bacterial levels are an important indicator used in determining water potability, but it is not contained in the dataset we used. Additionally, the class sizes in our dataset were uneven; 61% of the water samples were non-potable and 39% were potable. Future research could consider using undersampling or oversampling to make the class sizes more equal when training models. Another thing to note is that model parameters were tuned based on total accuracy in classification.

References

Environmental Protection Agency. (n.d.) National Primary Drinking Water Regulations. *EPA.gov.* https://www.epa.gov/sites/default/files/2016-06/documents/npwdr_complete_table.pdf

Kadiwal, Aditya. (n.d.) Water Quality. *Kaggle*. https://www.kaggle.com/datasets/adityakadiwal/water-potability?resource=download