A comparison of Random Forest(RF) and Naïve Bayes (NB) on predicting water potability.

Morteza Layegh Mirhosseini
Student ID: 220003166

Brief description and motivation of the problem

- It is quite common to use different machine learning methods for water potability classification[1].
- Machine learning methods help us to avoid time-consuming and expensive experiments for the classification of water potability[2].
- In this project, we compare two supervised machine learning approaches, Random Forests and Naive Bayes, for a classification of different water samples.
- Our main goal is to compare and improve the performance of each model to predict whether a sample of water is drinkable or not.

Initial analysis of the dataset including basic statistics

- The dataset is Water Quality from the Kaggle website.
- The dataset has 3276 rows and 10 columns; there are 9 independent variables (ratio) as predictors and one label column which contains binary values: 0 for Not potable water, 1 for potable water
- Table 1 provides information about the predictors with max, min, mean and standard deviation.
- The pie chart, Figure 1, gives information about the percentage of the observation for each target value. As we can see, the dataset is fairly unbalanced regarding our target label with 61% of the observation being non-potable water and 30 percent of the observation being potable.
- The correlation heatmap, figure 2, shows the correlation between different predictors. we can see that, there is a very low correlation between features.
- Finally, the histograms give insights into the distribution of each feature with regard to different target labels.

Figure 1: The Percentage of each Class in the dataset

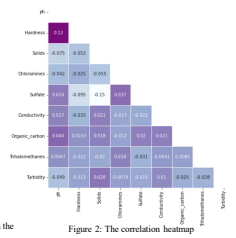

Figure 2: The correlation heatmap

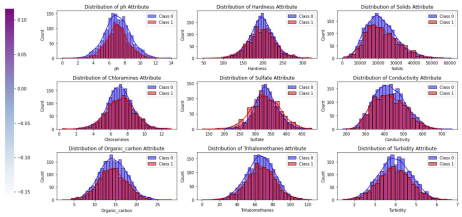

Figure 3: Distributions of all features with regard to different target labels

Table 1: Feature characteristics including mean, max, min, and std

Predictors	Mean	Max	Min	Std
Ph	7.08	14.00	0	1.59
Hardness	196.36	323.12	47.43	32.87
Solids	22014.09	61227.19	320.94	8768.57
Chloramine	7.12	13.12	0.35	1.58
Sulfate	333.77	481.03	129.00	41.41
Conductivity	426.20	753.34	181.48	80.82
Organic Carbon	14.28	28.3	2.20	3.30
Trihalomethanes	66.39	124.00	0.73	16.17
Turbidity	3.96	6.73	1.45	0.78

Hypothesis Statement

- We expect that the Random Forest model performs better in comparison to the Naïve Bayes model.
- Training and testing time is likely to be much higher for Random forest.
- We expect to see better performance for each model after applying the Synthetic Minority Oversampling Technique (SMOTE) to address the imbalance classes problem.

Description of the choice of training and evaluation methodology

- Split the dataset into a training set (80%) and a test set (20%) in order to evaluate the model's generalization error.
- Preprocess the training and test sets separately to avoid information leakage. This may include filling null values, balancing the dataset if necessary, and standardizing the data.
- Use 10-fold cross-validation on the training set to evaluate the model's performance for a given set of hyperparameters. Calculate the average performance across all folds using metrics such as accuracy, precision, recall, and AUC (area under the curve).
- Optimize the model using grid search to tune the hyperparameters, select features, and correct class imbalance if necessary.
- Choose the best set of hyperparameters, and best features based on the model's performance and retrain the model using the entire training set.
- Use the final model to predict the target in the test set and compare the predictions with the true labels to evaluate the model's performance on unseen data.
- Repeat the above steps using another machine learning method for comparison.
- Compare and contrast the performance of each machine learning model using the relevant performance metrics.

The choice of parameters and experimental results

Random Forest:

- We used MATLAB's *Tree bagger* to fit the model.
- We use random grid search to find the best features.
- We employed a cross-validation grid search to find the optimal number of trees and the number of predictors to sample at each node.

Random Forest Results:

- The grid search found that 300 trees and 3 predictors were the best numbers to sample at each node.
- The average accuracy across all folds for the best model was about 71%, and it took 9.6 seconds to train the final model.
- The training time increased as the number of trees and predictors increased as shown in figures 4 and figure 5.
- The final model's prediction time was 1.1 seconds.
- A ROC curve was also generated to evaluate the model's performance.

Naive Bayes:

- We used MATLAB's *fitcnb* to fit the model.
- We used the chi-square method for feature selection to select the top 5 important features.
- Grid search was used to find the best prior probability of each class and the best distribution to use for the class-conditional probabilities.

Naïve Bayes Results:

- The grid search found that the best distribution to use for the class-conditional probabilities is normal, and the best prior is empirical.
- The average accuracy across all folds for the best model was 55%, and it took 0.0042 to train the final model.
- The prediction time of the best model was 0.0013 seconds
- A ROC curve was also generated to evaluate the model's performance.

Table 2: Classification Results

NB	Measure	RF
0.0042	Train time(s)	9.6
0.0013	Predict time(s)	1.1
0.55	Cross Validation Accuracy	0.71
0.61	Test Accuracy	0.64
0.60	Precision	0.64
0.66	Recall	0.61
0.63	F1 score	0.63
0.64	AUC	0.70

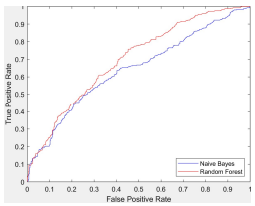

Figure 8: ROC Curves For Random Forest and Naïve Bayes

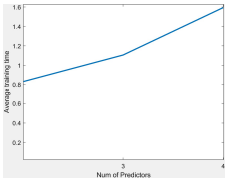

Figure 4: Avg. training time vs. number of predictors

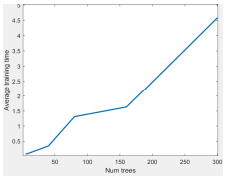

Figure 5: Avg. training time vs. number of trees

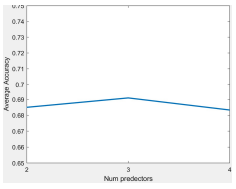

Figure 6: Avg. 10-fold cross-validation accuracy vs number of Predictors

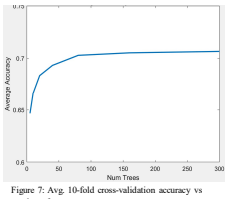

Figure 7: Avg. 10-fold cross-validation accuracy vs number of tress

A brief summary of the two machine learning methods with their pros and cons

Naïve Bayes

- The Naïve Bayes classifier assigns the most probable class to an observation with a feature vector[3].
- This classifier simplifies the learning process to a great extent by assuming that features are independent with respect to a class, a "naive" assumption"[3].
- It estimates the likelihood of each feature given each class, which represents the probability of a particular feature value occurring given a particular class label.
- A decision rule e.g. maximum posterior or MAP decision Rule combined with Naïve Bayes chooses the most probable class label [4].
- In order to estimate the prior probability there are different approaches but often we assume equiprobable classes or we calculate each class probability from training data [4].
- Despite the "naive" assumption, this classifier can compete with more sophisticated methods and provide accurate results in many practical applications, including text classification, and medical diagnosis[3]

Advantages:

- Easy to understand
- Compared to many other machine learning methods, this method is faster to train a model and predict classes[5].
- This method gives reliable results even if we only have a small dataset[5].

Disadvantages:

- Naïve Bayes makes a strong assumption of independence of class features. Such characteristics of a dataset's features are extremely rare to come across in the real world[5].
- The "Zero Conditional Probability Problem" is one of Naive Bayes's disadvantages. This problem eliminates all the information in other probabilities. To solve this issue, some sample correction methods exist, such as "Laplacian Correction"[5].
- It does not provide good results when we have continuous features compared to when we have discrete features.

Random Forest

- A Random Forest model is the ensemble of different base models which are usually decision trees[6].
- These decision trees are usually trained on different subsets of data[6].
- Random forest method makes a decision about the class of an observation based on the majority votes of its decision trees[6].

Advantages

- The concept of the model is easy to understand and explain.
- Random Forest benefits from the additional efficiency and accuracy of a group of models[2].
- It can measure the importance of each feature. The feature with the highest importance will do the best job of splitting the data with regard to the target label[7].

Disadvantages

- In terms of having categorical variables with different discrete values, Random Forest can be biased toward those values with the most frequency[2].
- Training a Random forest model can be computationally expensive. This is mainly a major issue when we have a large dataset.

Analysis and critical evaluation of results

- In our analysis, we initially trained a Gaussian Naïve Bayes model using unbalanced data. While the model achieved an accuracy of 60%, we observed a poor performance in predicting true negatives upon examining the confusion matrix. Given that the proportion of positive and negative samples in the dataset was not equal (60% to 40%), it is possible that the model's accuracy was largely influenced by the class imbalance. As such, we implemented the Synthetic Minority Oversampling Technique (SMOTE) to balance the data and address the issue of class imbalance. After balancing the data, the accuracy of the model decreased by approximately 3%, however, the model demonstrated an improved ability to predict the negative class.
- The implementation of SMOTE to correct the class imbalance also reduced the Random Forest accuracy by a few percent. However, it improved precision and recall scores.
- Feature selection using the chi-square test slightly improved the performance (validation accuracy) of the Naïve Bayes model by approximately 3%. However, the accuracy of the Naïve Bayes model is still relatively low. After using grid search cross-validation to find the best hyperparameters, we obtained the final trained Naïve Bayes model, results can be seen in table 2.
- We trained our Random Forest model using all available features in the dataset at first, but the recall score was quite low. Therefore, we employed the chi-square method for feature selection in an effort to improve performance. However, the results indicated that using the chi-square method did not improve the model's performance. As a next step, we employed a random search method to identify the best features for our model[8]. This approach is computationally expensive, but since our dataset is relatively small, containing only nine features, it was possible to implement. The features identified by the random search were pH, hardness, solids, chloramine, and sulfate. We then utilized grid search with 10-fold cross-validation to identify the optimal hyperparameters, resulting in the final trained Random Forest model.
- Our analysis showed that neither the Random Forest nor the Naïve Bayes model performed well on our dataset. Despite attempting various feature selections and using grid search for hyperparameter tuning, these techniques did not significantly improve the performance of the models. One possible reason for poor performances could be that the distribution of both classes (0 and 1) is very similar across each feature(see Figure 3), making it difficult for the models to properly learn how to distinguish between the two classes.
- Despite of the low performance of both models, the results of our experiments support our hypothesis. The Random Forest model performs slightly better compared to the Naïve Bayes model on both the validation and test phases. The only performance metric in which the Naïve Bayes model outperformed the Random Forest model was recall, which suggests that the Naïve Bayes model is better at identifying all relevant instances in the data, even if it is not generally as accurate as the Random Forest model.
- It is notable that the average validation accuracy of the Random Forest (RF) model was significantly higher than that of the Naïve Bayes (NB) model, with 71% and 55% respectively(Table2). However, this difference was less significant when the models were evaluated on the test set, with the Random Forest model achieving an accuracy of approximately 64% and the NB model achieving an accuracy of 61%. These results suggest that the Random Forest model may have slightly better generalization performance compared to the NB model, although the difference is relatively small.
- Another reason for the difference between validation accuracy and test accuracy could be that the model is overfitted. However, since this difference is not significant, we cannot say for sure whether this is the case.
- Figure 8 shows that the Random Forest ROC curve is plotted above the Naïve Bayes Roc Curve. This means that the Random Forest model has a higher true positive rate (TPR) and a lower false positive rate (FPR) at various classification thresholds compared to the Naïve Bayes model. This suggests that the Random Forest model is a better classifier as it is able to correctly classify more positive cases while also minimizing the number of negative cases that are incorrectly classified as positive.
- The training time for the final Random Forest model was significantly longer than that of the Naïve Bayes model, taking 9.6 seconds compared to 0.0042 seconds. Similarly, the prediction time for the Random Forest model was significantly longer than that of the Naïve Bayes model, with 1.1 seconds compared to 0.0013 seconds. This suggests that the Random Forest model is more computationally intensive in both training and prediction stages compared to the Naïve Bayes model. We expected these results in our hypothesis statement. These results could be mainly because Random Forest is a more complex model than Naïve Bayes. Also, the results show us that the Naïve Bayes model is a good choice if we want to get results fast.
- Figure 7 presents the results of hyperparameter tuning for a Random Forest model. The model's average 10-fold validation accuracy was measured as the number of trees in the model varied. It can be seen that after around 80 trees, the average validation accuracy did not improve significantly. Despite this, we selected 300 trees in order to maximize the validation accuracy. Figure 6 shows the average 10-fold validation accuracy of the model as the number of predictors was changed. The results indicate that the number of predictors does not significantly impact the model's performance. In conclusion, the number of trees in the Random Forest model appears to have a stronger effect on its accuracy compared to the number of predictors.
- Our analysis of the Naïve Bayes model showed that it performed better on the test set compared to the validation set(see Table 2). This may be due to the imbalance of the test set, as opposed to the balanced validation set. Additionally, this observation could be a result of the random splitting of the dataset into test and train sets. Overall, we were unable to identify another explanation for this observation.

Lesson learned

- To ensure that our model generalizes well to the test data, we should normalize the test data based on the minimum and maximum values from the training set. If we normalize the test set without considering the range of data in the training set, we may change the distribution of the test set, making it significantly different from the training set. This can reduce the test accuracy of our model and prevent it from generalizing well to new data. Therefore, it is important to assume that the training and test sets have similar distributions and to normalize the test data accordingly.
- We should not resample the test set for balance because it would change the distribution of the test set and result in a misleading evaluation of results.

Further work

- The effects of dealing with outliers can be evaluated on both models.
- The effects of other hyperparameters for the Random Forest model can be evaluated.
- Gathering more data may help to improve the performance of the models.
- The effects of correcting imbalanced data using techniques such as oversampling and undersampling can also be considered.
- Feature selection using other methods could help the model to train with the most relevant features and potentially provide better results.
- The performance metrics could be improved by using other machine learning algorithms, such as logistic regression and k-nearest neighbor.

References

1. Zhu, M., et al., *A review of the application of machine learning in water quality evaluation*. Eco-Environment & Health, 2022.
2. Ahmed, U., et al., *Efficient water quality prediction using supervised machine learning*. Water, 2019. **11**(11): p. 2210.
3. Rish, I. *An empirical study of the naive Bayes classifier*. in *IJCAI 2001 workshop on empirical methods in artificial intelligence*. 2001.
4. Bishop, C.M. and N.M. Nasrabadi, *Pattern recognition and machine learning*. Vol. 4. 2006: Springer.
5. Kaviani, P. and S. Dhotre, *Short survey on naive bayes algorithm*. International Journal of Advance Engineering and Research Development, 2017. **4**(11): p. 607-611.
6. Nasir, N., et al., *Water quality classification using machine learning algorithms*. Journal of Water Process Engineering, 2022. **48**: p. 102920
7. Prajwala, T., *A comparative study on decision tree and random forest using R tool*. International journal of advanced research in computer and communication engineering, 2015. **4**(1): p. 196-199.
8. Zheng, A., *Evaluating machine learning models: a beginner's guide to key concepts and pitfalls*. 2015: O'Reilly Media.