Drinking Water Potability

Introduction

In our daily lives, drinking water comes from various sources including rivers, streams, and ground. However, not all drinking water is safe. It is important to ensure the potability which is affected by water's features and chemical context before potation.

Methods

Machine Learning is a subset of the field of Artificial Intelligence, it is a method of computer algorithms to interpret the data's repeating pattern, aiming to allow algorithms to learn patterns and gradually increase their data accuracy automatically. Machine Learning is widely used in many applications, for example, image recognition (especially in medical use in X-rays), social media suggesting content that may arouse user's interest, etc., Whereas Statistics and Statistical Modeling is the science of using types of statistical and mathematical methods to interpret data and make inference from it. Therefore, Machine Learning is strongly similar to Statistical modeling and related to the subject of Statistics: They are both targeted to learn from data and recognize the pattern. Moreover, a large proportion of Machine Learning concepts are drawn from Statistics and mathematics. Hence, our goal of this project is to create a Machine Learning classifying algorithm with background knowledge from Statistics.

Our project will be building a model using the "classification" method in machine learning. First, we find an appropriate water potability dataset which consists of 10 parameters from Kaggle. Secondly, the dataset is split into the training set and the testing set. Then, data are preprocessed by procedures like data cleaning, correlation, and featuring scaling. Finally, the roc curve and accuracy score of the four training models are compared, in order to find out the most suitable method for classifying water potability. Four training models, including the K-nearest neighbors (k-NN) algorithm classifier, random forest classifier, SVM classifier, and SGD classifier are used. By these classifiers, we are able to classify the substance, chemical element, and feature of water to ensure the water potability is safe or not for human consumption.

Data Collection

We used 3276 water samples in total and with 10 parameters for model training. The 10 parameters include pH values (an indicator of acidic or alkaline condition), Hardness (capacity of water to precipitate soap caused by Calcium and Magnesium), Solids (total dissolved solids), Chloramines (a major disinfectant), Sulfate (naturally occurring substances that are found in minerals, soil, and rocks), Conductivity (electrical conductivity), Organic Carbon (total organic

carbon), Trihalomethanes (chemicals which may be found in water that were treated by chlorine), Turbidity (a measure of light-emitting properties), and Potability (Indicator of safe water). There are 1278 passing examples and 1998 failing examples for potability. The response variable is Potability value, rest of the variable is explanatory variable. Our goal is to use the explanatory variable to classify the response variable.

The statistics of each parameter are as shown:

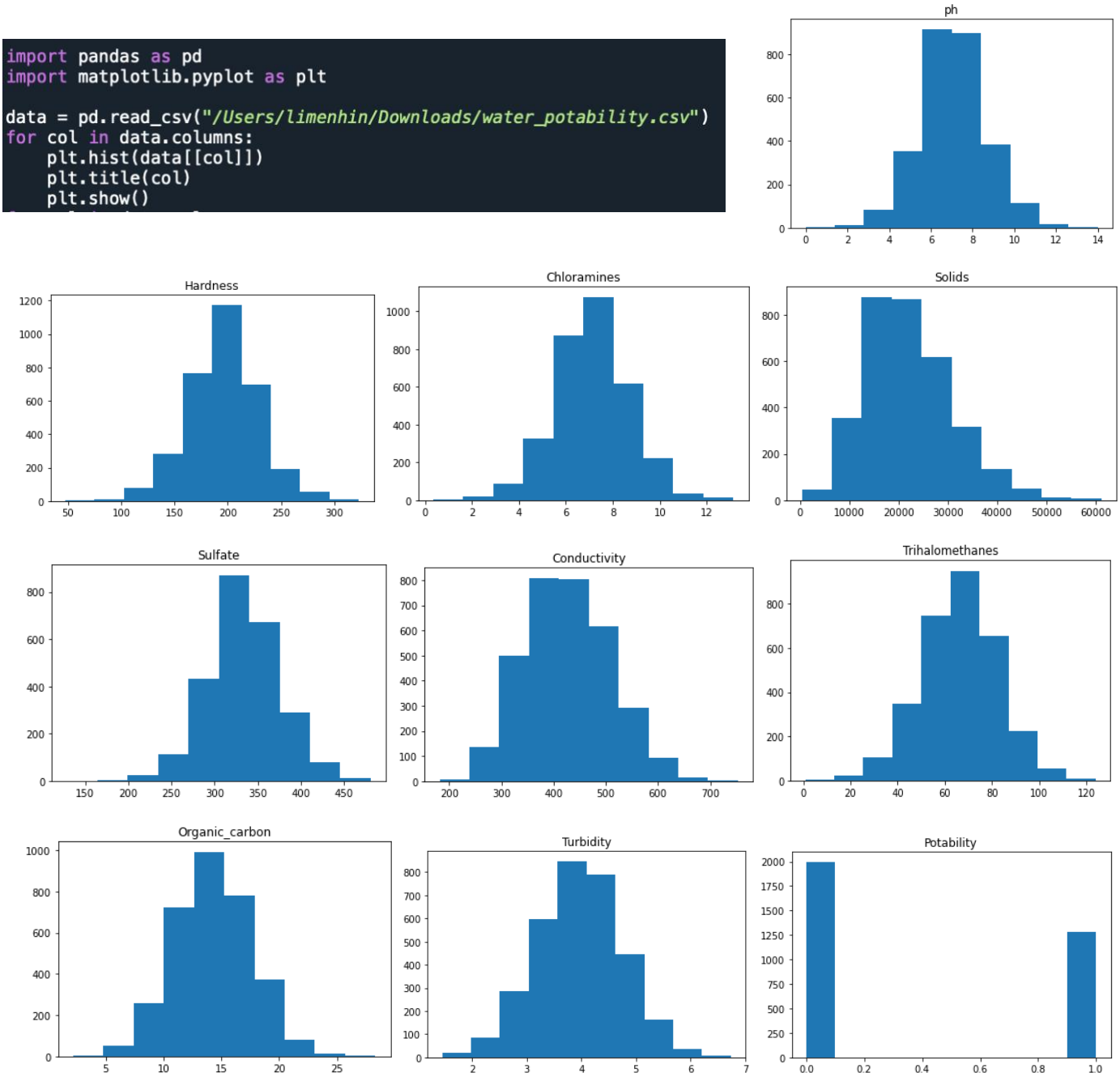
Parameter	Mean	SD	Minimum	Maximum
pH Values	7.078	1.497	0.000	14.000
Hardness	196.369	32.880	47.432	323.124
Solids	22014.100	8768.570	320.943	61227.200
Chloramines	7.122	1.583	0.352	13.127
Sulfate	333.607	37.458	129.000	481.031
Conductivity	426.205	80.824	181.484	753.343
Organic Carbon	14.285	3.308	2.200	28.300
Trihalomethanes	66.405	15.837	0.738	124.000
Turbidity	3.967	0.780	1.450	6.739

Data visualization

For a better understanding of each parameter, we used python to help plot the histogram.

```
import pandas as pd
import matplotlib.pyplot as plt

data = pd.read_csv("/Users/limenhin/Downloads/water_potability.csv")
for col in data.columns:
    plt.hist(data[[col]])
    plt.title(col)
    plt.show()
```



And we discovered that most of the parameters are bell-shaped, but the parameter solids and sulfate are having positive skewness and negative skewness respectively. Last but not least, our parameter potability, which will be our output variable later in different classification models.

Imbalanced Data

From the pie chart, we observed that this data set is quite balanced. The ratio between potable water and unsafe drinking water is about 4 to 6. It is close to 50%, therefore we do not need additional data transform.

Correlation Matrix

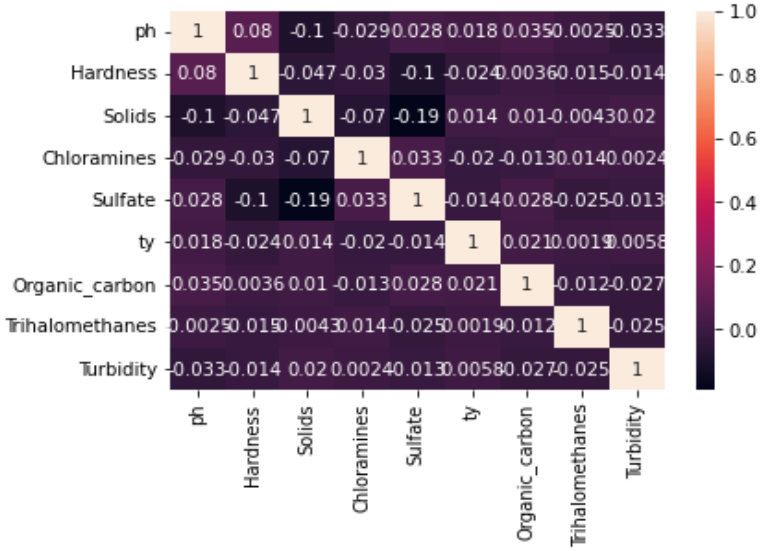
Since different classifiers of machine learning are sensitive to the dependency of different explanatory variables, if an explanatory variable is dependent or highly correlated to another, that means the explanatory variable provides redundant information to the classifier and gets an unfavorable result. Therefore, we would like to find a Correlation matrix to see the relationship of each explanatory variable. First, we import the data into python.

```
import numpy as np
import pandas as pd
import matplotlib as mpl
import matplotlib.pyplot as plt
data = pd.read_csv(r'D:\henry academic\water_potability.csv')
X = pd.read_csv(r'D:\henry academic\water_potability.csv')
del X["Potability"]
```

Here X is our explanatory variable, then we create a correlation matrix.

```
27  Y = data["Potability"] # define Categorical variables as Y
28
29  corrMatrix = X.corr() # creata a Correlation Matrix of x
30  print (corrMatrix)
```

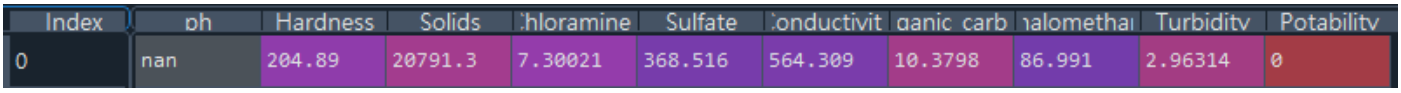
From the correlation matrix of the correlation, we can see the correlation of each explanatory variable is close to zero, therefore we assume all explanatory variables are not highly correlated and keep all explanatory variables into the classifier.



Data cleaning and transform

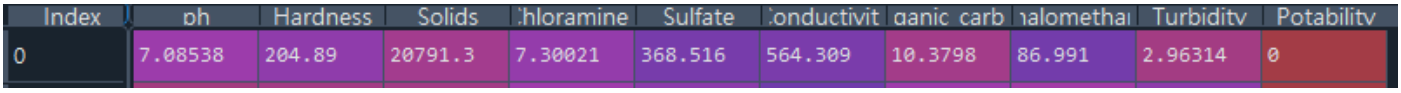
1. *Fill in missing data*

In this data set, several data are missing in the different parameter columns such as the pH value, Sulfate and Trihalomethanes. Since the classifier will not accept missing data, we would like to fill in the missing value by replacing the value by the mean value. First, separate the data by the "Potability" value. Group one, the "Potability" value of data equal to one, Group two "Potability" value of data equal to zero. Find the mean value of pH value, Sulfate and Trihalomethanes in each group. If the missing data belong to group one, fill in the missing value by filling in the corresponding column.



Index	ph	Hardness	Solids	hloramine	Sulfate	onductivit	ganic carb	alomethai	Turbidity	Potability
0	nan	204.89	20791.3	7.30021	368.516	564.309	10.3798	86.991	2.96314	0

For example, this data belongs to group two and it has a missing value in the ph value. Therefore, we fill in the mean value of the ph value in group two.



Index	ph	Hardness	Solids	hloramine	Sulfate	onductivit	ganic carb	alomethai	Turbidity	Potability
0	7.08538	204.89	20791.3	7.30021	368.516	564.309	10.3798	86.991	2.96314	0

```
df['ph'] = df['ph'].fillna(df.groupby(['Potability'])['ph'].transform('mean'))
df['Sulfate'] = df['Sulfate'].fillna(df.groupby(['Potability'])['Sulfate'].transform('mean'))
df['Trihalomethanes'] = df['Trihalomethanes'].fillna(df.groupby(['Potability'])['Trihalomethanes'].transform('mean'))
X = df.drop("Potability", axis=1)
Y = df["Potability"]
```

Here X is our explanatory variables with no missing value and Y is response variables

2. *Create train set and test set*

In machine learning, not all the data will be trained on some of the remaining data to test sets to test the accuracy of our model. Therefore we need to split our data to be a train set and test set from which we fit the data into the classifier. In this project, we reorder the data randomly then split 80% data into a train set and the remaining 20% into a test set.

```
def split_train_test(data, test_ratio): # define a function to split train set and test set
    np.random.seed(10)
    shuffled_indices = np.random.permutation(len(data))
    test_set_size = int(len(data) * test_ratio)
    test_indices = shuffled_indices[:test_set_size]
    train_indices = shuffled_indices[test_set_size:]
    return data.iloc[train_indices], data.iloc[test_indices]

train_set_X, test_set_X = split_train_test(X, 0.2) # find the train set and test set of X
train_set_Y, test_set_Y = split_train_test(Y, 0.2) # find the train set and test set of Y
```

3. *Scaling*

In machine learning, if the numerical attributes have different scales, the machine learning algorithms will not perform well. Therefore, we do feature scaling before we fit the train set into the machine learning algorithms. We observed a lot of outliers in different numerical attributes from the box and whisker diagram, so we use standardization to reduce the influence of outliers. Standardization=(value - mean value)/standard deviation

```
50  from sklearn.preprocessing import StandardScaler
51  Standardization_scaler = StandardScaler()
52  standardization_train_set_X = Standardization_scaler.fit_transform(train_set_X) # use standardization to rescaled the data
```

Classifying Models

1. *K-nearest neighbors (k-NN) algorithm classifier*

K-nearest neighbors (k-NN) algorithm classifier is a non-parametric classifier, and a supervised learning algorithm. The k-NN model predicts unknown data depending on its surrounding data, by selecting a specified number (k) neighbors and calculating distances from each data in the data set to the unknown data, k number of the closest data will be chosen and considered in classification.

2. *Random forest classifier*

Random forest classifier is a classifier method made by a few decision trees. The decisions are based on the criteria set initially to distinguish different types of data. The criteria in trees can be developed if algorithms learn more patterns in the data set.

3. *Support Vector Machine(SVM) classifier*

Support Vector Machine is a supervised machine learning algorithm that can be used for classification. Its objective is to find a hyperplane in an N-dimensional space that distinctly classifies the data points. To classify more data points, a plane that has a maximum margin is found by SVM. Like the distance between data points of two classes.

4. *Gaussian Processes Classifier*

Gaussian process classification which is a Bayesian method. Assuming the data are random variables that distribute as a multivariate normal distribution .The model consists of a mean function and a covariance matrix which is also called kernel. The similar data should have a similar target value calculated by kernels.The key is to select the suitable kernel for your own case.

Hyperparameter Tuning

Data analysts need to input the Hyperparameter into learning algorithms so that the algorithm can fit into data well. But unlike parameters, Hyperparameter are not optimized by the learning algorithms itself through the training, data analysts should find the optimized hyperparameter by testing different combinations of Hyperparameter. In this project, we will use the method of Grid Serah. Grid Serah, data analysts input a range of Hyperparameter and test the result of every single combination, compare their result by k fold cross validation. In K Fold Cross Validation, the algorithms will split the train set into K fold and do the validation for K time, every test will pick one fold as validation set and K-1 as train set. Every fold is treated as validation once time.

Below is the Hyperparameter used in this project,

- K-nearest neighbors (k-NN) algorithm classifier

  N_neighbour: Decide how many neighbors we take in the algorithm

  Weights: Check add weight to the data point is better or not

  Metric: find the distance metric will use

- Support Vector Machine (SVM)

  C: the penalty parameter, tell the SVM how much error is bearable

  Gamma: How far influences the calculation of plausible line of separation

- Random forest classifier

  N estimators: number of trees in forest

  Max features = max number of features for splitting node

  Max depth = max number of levels of decision tree

  Min sample split = min amount of data in a node before the node is split

- Gaussian Processes classifier (GPC)

  Kernels: The shape of the prior and posterior of the Gaussian Process

  1. Radial-basis function (RBF) kernel / Squared exponential kernel
  2. Matern kernel
  3. Dot-Product kernel

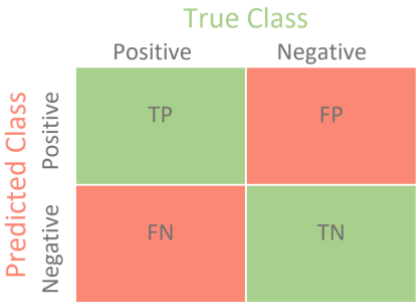
➔ Radial-basis function is applied, which has the parameter ℓ : the length-scale.

$$k(x_i, x_j) = \exp\left(-\frac{d(x_i, x_j)^2}{2l^2}\right)$$

Error Analysis

Confusion Matrix

In binary classifiers, our result can be treated as 4 types: True positive (TP), True Negative (TN), False Positive(FP), and False Negative(FN). Those 4 types are important criteria for us to evaluate the preform of the learning algorithms, so the confusion matrix helps us to contain the number of those data.



Also, In the confusion matrix we can find different scores: Precision, Recall, Accuracy, and F1 score to evaluate the learning algorithms.

- Precision (ratio of positive prediction to the total of positive prediction) = TP/TP+FP
- Recall (ratio of correct positive prediction to number of positive class) = TP/TP+FN
- Accuracy (number of correct predictions to total number of data) = TP/TP+TN+FP+FN
- F1 score (harmonic mean of Precision and Recall) = 2 *Precision*Recall/(Precision+Recall)

ROC curve

Receiver operating characteristic curve plots the true positive rate (TPR) against the false positive rate (FPR) For a set of thresholds. Threshold is a dividing line if the prediction score is greater than the threshold, the data identify as positive class vice versa. If the area under the curve (AOC) is close to one, it means the learning algorithm is fitting the model well. Therefore, we would like to have a classifier with AOC value as large as possible.

TPR = Recall; FPR = FP/FP+TN

Results for train set

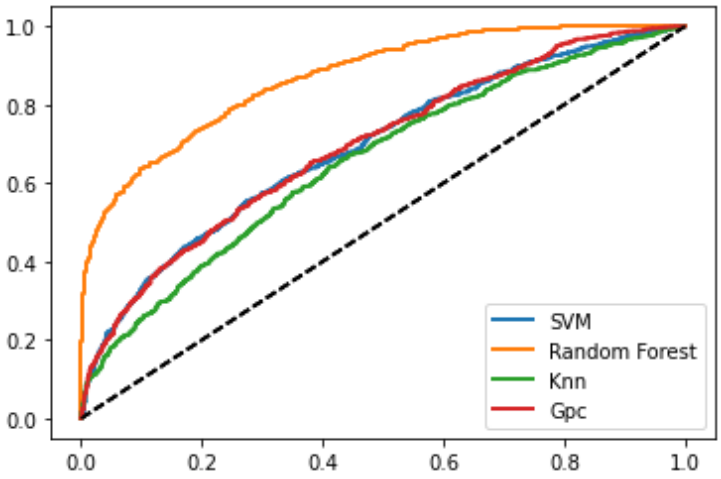
	Precision	Recall	Accuracy	F1 score	AOC
SVM	0.70588	0.27880	0.66997	0.39972	0.69006
Random forest	0.80194	0.63892	0.79550	0.71121	0.87001
KNN	0.625	0.22749	0.64174	0.33357	0.65445
GPC	0.66908	0.35818	0.67722	0.46658	0.69387

Random forest is the best model with all the highest scores, especially 0.80 accuracy and 0.87 AOC. Other models have similar results with more than 0.64 accuracy and 0.69 AOC.

Since our dataset has lots of outliers in different numerical attributes, random forest is robust to outliers and able to handle them automatically. Compared with other models, random forest is also less impacted by noise.

ROC curve

ROC curve shows that random forest is the best model that obtains the largest area under the curve. The Area of SVM and GPC are close to each other with similar performance. KNN model obtains the smallest area, which slightly performs worse than SVM and GPC.



Results for test set

	SVM	Random forest	KNN	GPC
True positive rate	0.191837	0.54693	0.130612	0.14947
True negative rate	0.946341	0.92682	0.946341	0.94410

Random forest has the highest true positive rate with 55%, but other models have below 20% true positive rate. However, all the models obtain great results with at least 92% in true negative rate.

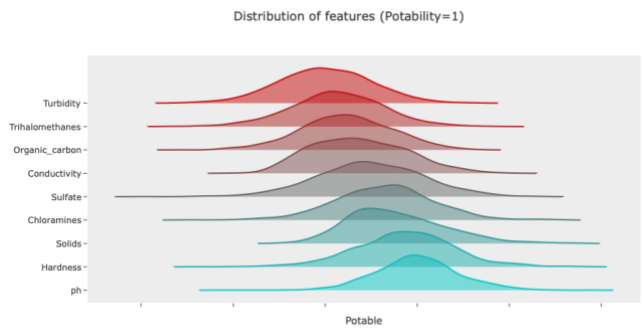
Irregular features of the dataset

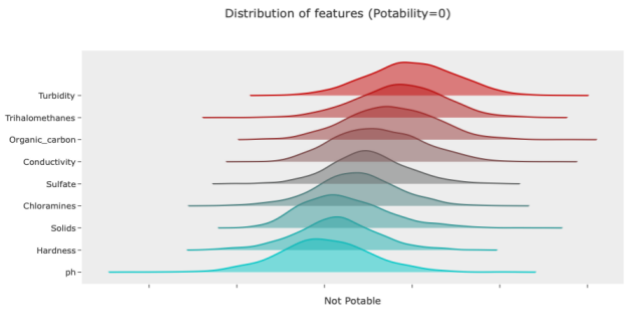
Rare phenomenon

There are lots of portable water samples with below 5 or more than 10 pH value. However, the maximum permissible limit of pH values suggested by the World Health Organization is from 6.5 to 8.5, which is different from our dataset (World Health Organization, 2007). Also, the average amount of organic carbon is 14 mg/L. In reality, it is generally less than 10 mg/L, such as 4 mg/L for source water and 2 mg/L for treated water (BC Victoria, 1998). The correlation between each of the parameters is low. It is believed that there is a relationship between conductivity and sulfate. However, since the dataset is from an invalid source from Kaggle, it is difficult to verify the data and get more information and background.

Normality of the data

Besides, after standardizing the raw data, then separating the data according to their portability, it is observed that most of the variables are well normal distribution, which is rare in reality.





However, after separating the standardized data according to their portability, it is observed some special and different patterns with their two sets of data. As for the data of portable water, the standardized turbidity is lower than the standardized pH value, but the data of not portable water has the opposite observation. Therefore, our models still work well to classify them according to their different features.

Limitation

First, since the dataset may not be the observed data set, it is not able to use our default mind and knowledge to treat the data. However, it is still a good example for beginners to try the binary classifier and handle the data cleaning part. Second, it is suggested to explore more data to increase the sample size to increase the accuracy of the model. Since the background of the dataset is invalid, some resampling methods, such as bootstrap, can also improve the performance. Third, because of the limitation of the specs of the computer, it is not able to tune out the most optimal hyperparameters of models. For example, as for the GPC, it takes more than an hour to find out the optimal kernel and its parameters. Fourth, trying different data cleaning methods can obtain a better performance, since our data cleaning method performs better than applying the KNN imputer and dropping rows with missing values already. Last, it is suggested to find the dataset from some credible sources, such as government websites, to ensure the quality of the data.

Conclusion

The results suggested that random forest is the best model, which obtains the highest score in most of the criteria, including accuracy 0.80, AOC 0.87, 55% true positive rate. Then, the performance of Support Vector Machine and Gaussian Processes Classifier is similar, and the K-nearest neighbors algorithm classifier slightly performs worse than SVM and GPC. However, all the models obtain great results with at least 92% in true negative rate. Overall, all models work well in classifying whether the water sample is potable or not, and the random forest is the most outstanding model.

References

Géron, A. (2019). Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems. 2nd Edition. O'Reilly Media.

Aditya Kadiwal(2021, April). Water quality. Kaggle. https://www.kaggle.com/adityakadiwal/water-potability

Jijo Abraham(2019, Sep) .A Beginner's Guide to K Nearest Neighbor(KNN) Algorithm With Code.Medium.https://medium.com/analytics-vidhya/a-beginners-guide-to-k-nearest-neighbor-knn-algorithm-with-code-5015ce8b227e

Gandhi, R. (2018, July 5). *Support Vector Machine - introduction to machine learning algorithms*. Medium. https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47

Koehrsen, W. (2019, December 10). *Hyperparameter Tuning the Random Forest in Python - Towards Data Science*.Medium.https://towardsdatascience.com/hyperparameter-tuning-the-random-forest-in-python-using-scikit-learn-28d2aa77dd74

SVM Hyperparameter Tuning using GridSearchCV. (2020, March 10). Velocity Business Solutions Limited. https://www.vebuso.com/2020/03/svm-hyperparameter-tuning-using-gridsearchcv/

A. (2020, April 25). *k-NN with Hyperparameter Tuning*. Kaggle. https://www.kaggle.com/arunimsamudra/k-nn-with-hyperparameter-tuning

Machine learning random forest algorithm - javatpoint. www.javatpoint.com. (n.d.). https://www.javatpoint.com/machine-learning-random-forest-algorithm

World Health Organization(2007). pH in Drinking-water. WHO Guidelines for Drinking-water Quality.https://www.who.int/water_sanitation_health/dwq/chemicals/ph_revised_2007_clean_version.pdfht

Victoria, BC(1998). Water Quality-Ambient water quality criteria for organic carbon in British Columbia.https://www2.gov.bc.ca/assets/gov/environment/air-land-water/water/waterquality/water-quality-guidelines/approved-wqgs/organic-carbon-tech.pdf

Emrearslan123. (2022, February 12). *Water potability prediction* . Kaggle. https://www.kaggle.com/emrearslan123/water-potability-prediction

Analytics Vidhya. (2020, July 20). *Knnimputer: Way to impute missing values*. https://www.analyticsvidhya.com/blog/2020/07/knnimputer-a-robust-way-to-impute-missing-values-using-scikit-learn/

Mandal, Kishal. (2021, November 26) EDA - A Categorical Approach. https://www.kaggle.com/kishalmandal/eda-a-categorical-approach/notebook