INSTITUTE OF HIGHER EDUCATION AND RESEARCH
(Declared as Deemed - to - be - University under section 3 of UGC Act 1956)

173, Agaram Road, Selaiyur, Chennai-600073

BHARATH INSTITUTE OF SCIENCE & TECHNOLOGY

Department of Computer Science & Engineering

WATER QUALITY ANALYSIS

TERM PAPER

SUBCODE: U18PRCS6P1

Name: MOTUPALLY SRIKARACHARYA

Year: III **Semester: VI** **Batch: 2019-2023** **Section: CSE-L**

University Register No. **U19CN142**

JUNE 2022

Bharath
INSTITUTE OF HIGHER EDUCATION AND RESEARCH
(Declared as Deemed - to - be - University under section 3 of UGC Act 1956)

173, Agaram Road, Selaiyur, Chennai-600073

Department of Computer Science & Engineering

SUBCODE: U18PRCS6P1

Name: MOTUPALLY SRIKARACHARYA

Year: III **Semester: VI** **Batch: 2019-2023** **Section: CSE-L**

University Register No. **U19CN142**

Certified that this is the bonafide record of work done by the above student in the **TERM PAPER (SUB CODE: U18PRCS6P1)** during **June 2022.**

Signature of Faculty-In-Charge **Signature of Head of Department**

Submitted for the practical Examination held on ……………………………………
at Bharath Institute of Higher Education & Research.

Signature of Internal Examiner **Signature of External Examiner**

TABLE OF CONTENT

ABSTRACT

A Regression algorithm is used to assign predefined classes to test instances for evaluation (or) future instances to an application. This study presents a Regression model using Random Forest Algorithm to analyze water quality. Water quality is very important in ensuring citizens can get to drink clean water. Application of Random Forest as an Ensemble Techniques to predict clean water based on the water quality parameters can ease the work of the laboratory technologist by predicting which water samples should proceed to the next step of the analysis. Regression using Random Forest was applied to predict the clean and not clean water. The analysis of water Hardness, solids, Turbidity, pH level, Sulfate, and conductivity can play a major role in assessing water quality. Nowadays Most Diseases are caused by Using Water To avoid Those diseases We are Implementing This Model.

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Supervised learning is a machine learning algorithm that receives a feature vector and the target pattern as an input to build a model. The model can be used to recognize new patterns and assign a target to them. Applications of supervised learning include classification and Regression, Unsupervised learning is a machine learning algorithm that only receives the feature vector as an input, and its task is to find similar groups of items with comparable features. The essential application of unsupervised learning is clustering, such as determining the distribution of data items within a multidimensional space of given data.

Regression is an instance of supervised learning that includes a training phase to create a model (Regressor). Its task is to predict the class of items in a data set using a certain model of a Regression.
The model is constructed using already-labeled. The model is constructed using already-labeled items of similar data sets. This step allows Regression techniques to be considered as a supervised machine learning method. **Ensemble Techniques** are used to predict High accuracy using **hyperparameter tuning** and not only the regressor, but the Classifier algorithm also uses ensemble Techniques

1.2 Scope of the Project:

This analysis helps so many people from so many diseases which occur with water pollution in different areas of the world.

CHAPTER 2

LITERATURE SURVEY

[1] Jiang, J.; Tang, S.; Han, D.; Fu, G.; Solomatin, D.; Zheng, Y.

A comprehensive review on the design and optimization of surface water quality monitoring networks. Environ. Model. Soft. 2020, 132, 104792. The Neural Network of Machine Learning is based on Random Forest Regression to obtain a proper solution to address the problem of changes in the quality of drinking water.

Regression is an important problem in machine learning. It has been widely applied in many real-world applications examples as Food testing, loan prediction, and checking online fraud payments. To build a Regression, a user first needs to collect a set of training examples/instances that are labeled with predefined classes. A Regression algorithm is then applied to the training data to build a Regressor that is subsequently employed to assign the predefined classes to test instances (for evaluation) or future instances (for application).

[2] Zhou, Z.H. Ensemble learning. In Machine Learning; Springer: Berlin/Heidelberg, Germany, 2021; pp. 181–210.

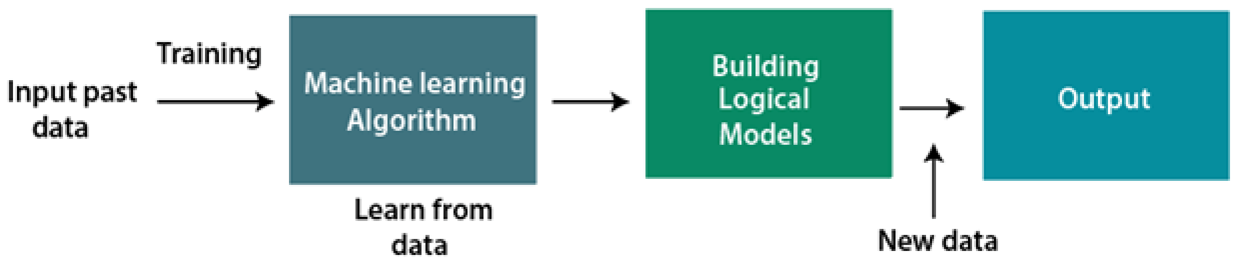
Random forest is a tree-based algorithm that involves the building of several trees and combining them with the output to improve the generalization ability of the model. This method of combining trees is known as an ensemble Technique where we can increase the accuracy of the model. The ensemble is nothing but a combination of weak learners (individual trees) to produce a strong learner of the given data. The bagging Algorithm is used to create random samples. Take data set D1 is given for n rows and m columns, and new data set D2 is created for sampling n cases at random with replacement from the original data to Predict a Machine Learning model. From dataset D1,1/3rd of rows are left out and are known as Out of Bag samples. Then, a new dataset D2 is trained to this model, and Out of Bag samples is used to determine an unbiased estimate of the error. Out of m columns, M << m columns are selected at each node in the data set. The M columns are selected at random. Usually, the default choice of M is m/3 for the regression tree and M is sqrt(m) for the classification tree. Unlike a tree, no pruning takes place in a random forest i.e; each tree is grown fully. For indecision trees, pruning is a method to avoid overfitting. Pruning means selecting a subtree that leads to the lowest test error rate. Cross-validation is used to determine the test error rate of a subtree. Several trees are grown and the final prediction is obtained by averaging or voting.

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

This research was based on unsupervised learning. The significance of this paper was to find new methods for Water Analysis and to increase the accuracy of results. The data set for this paper is based on real-life transactional data by a large European company and personal details in data are kept confidential and safe. The accuracy of an algorithm is around 70%. Thus, the accuracy of the results obtained from these methods is less when compared with the proposed system. A comprehensive understanding of the quality of a water sample can be helpful for us to solve the problem of Water Quality Analysis. The work provides a comprehensive discussion of the challenges and problems of water.



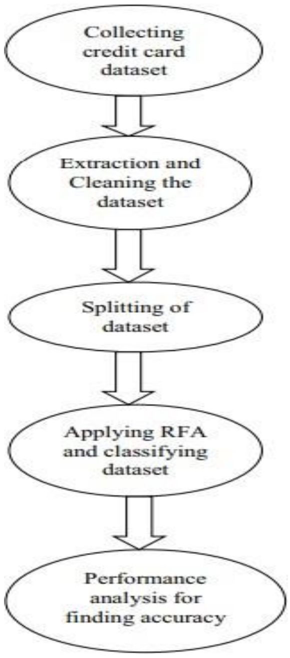
3.1 EXISTING SYSTEM DIAGRAM

3.2 PROPOSED SYSTEM

In the proposed system, we use RFA for the classification and regression of the dataset. First, we will collect the Water Quality dataset and analysis will be done on the collected dataset. After the analysis of the dataset then cleaning of the dataset is required. Generally, in any dataset there will be many duplicates and null values will be present, so to remove all those duplicates and null values cleaning process is required. Then we must split the dataset into two categories a trained dataset and a testing dataset for comparing and analyzing the dataset. After dividing the dataset, we must apply the RFA where this algorithm will give us better accuracy about the Water Quality Measures. By applying the RFA, the dataset will be classified into four categories which will be obtained in the form of a confusion matrix. In this analysis, the accuracy of Water Quality Analysis can be obtained which will be finally represented in the form of a graphical representation.

3.2.1 RANDOM FOREST ALGORITHM:

A random forest is also called a random decision forest which is used for classification, regression, and other tasks that are performed by constructing multiple decision trees. This RFA is based on supervised learning and the major advantage of this algorithm is that it can be used for both classification and regression. RFA gives you better accuracy when compared with all other existing systems and this is the most used algorithm. In this paper, the use of RFA in credit card fraud detection can give you an accuracy of about 90 to 95%.



.

3.2 EXISTING SYSTEM

3.2.2 DISADVANTAGES OF THE PROPOSED SYSTEM

01.It takes so much time to run the data set.
02.Decision tree has low accuracy.
03.Data has more null values.

CHAPTER 4

SYSTEM DESIGN

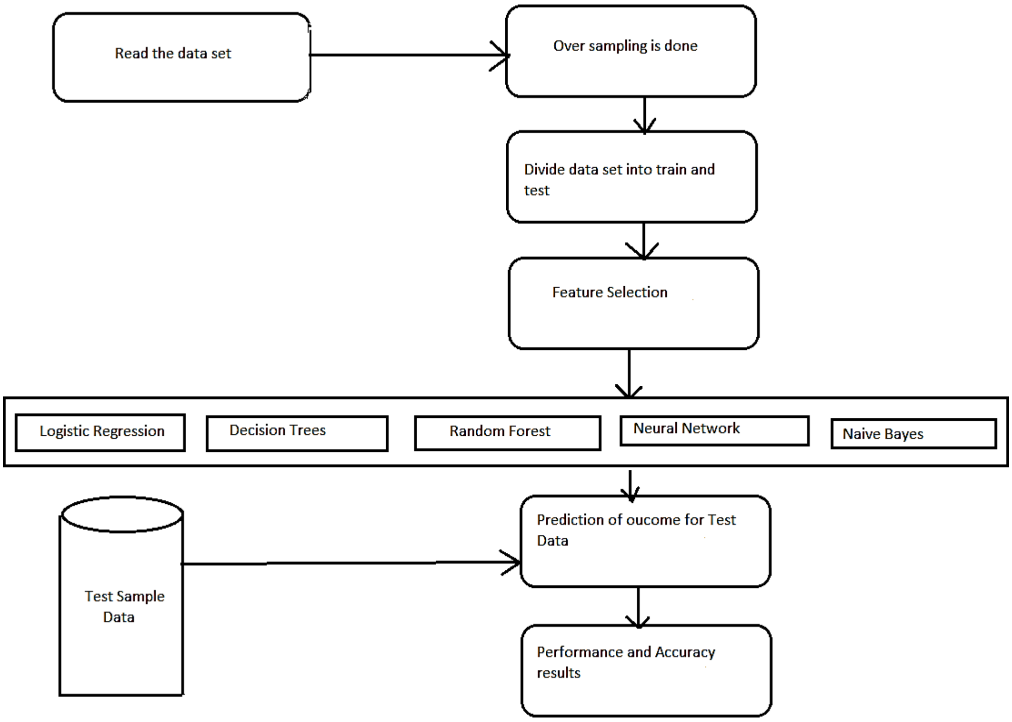


Fig 4.1 ARCHITECTURE DIAGRAM

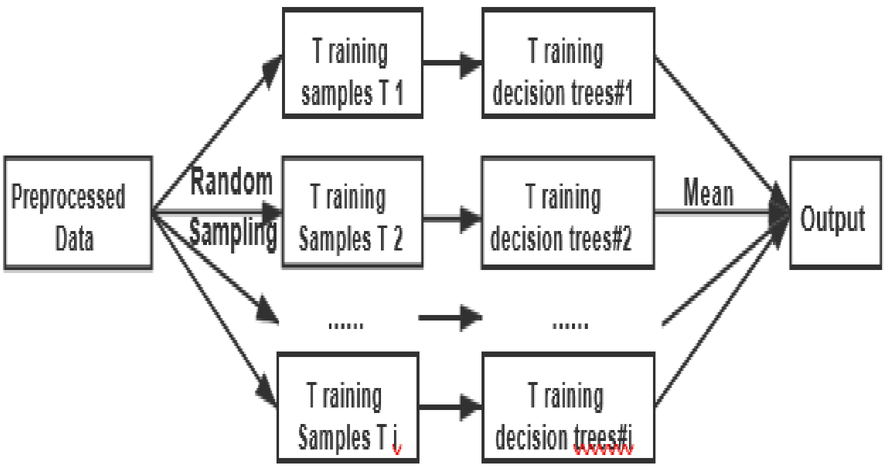


Fig 4.2 TRAINING ALGORITHM

CHAPTER 5

5.1 MODULES DESCRIPTION

5.1.1 EXPLORATORY DATA ANALYSIS

5.1.2 DATA CLEANING

5.1.3 DATA PREPROCESSING

5.1.4 SPLITTING THE DATA

5.1.5 PREDICTING THE ACCURACY

5.1.1 MODULE NAME: EXPLORATORY DATA ANALYSIS

Exploratory Data Analysis In this module we will first collect all the Water Quality analysis dataset and store it in a database. Then we will perform some descriptive analysis of the dataset.

5.1.2 MODULE NAME: DATA CLEANING

Data Cleaning Is the next step, after analyzing the dataset then we have to clean the data. In this cleaning process, all the duplicate values and null values that are present in the dataset will be removed and moved to further process.

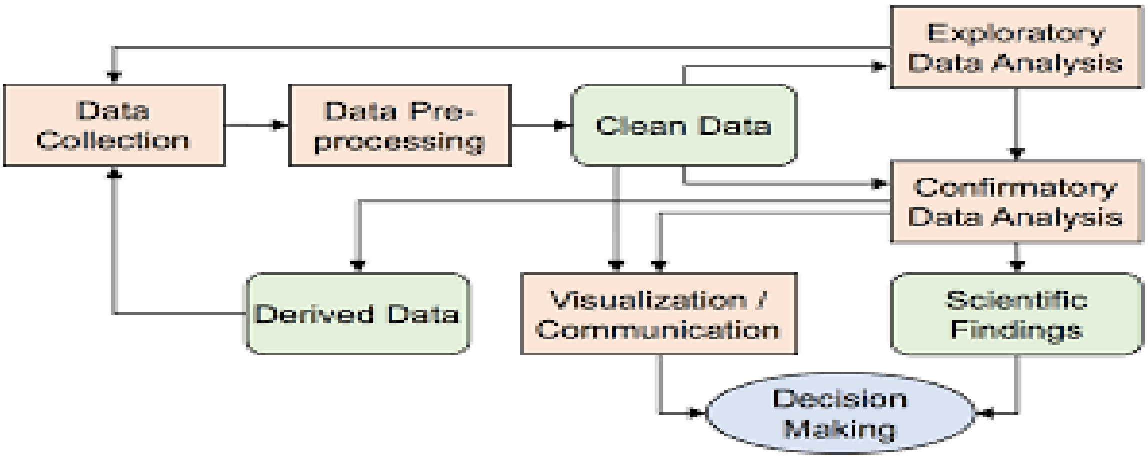
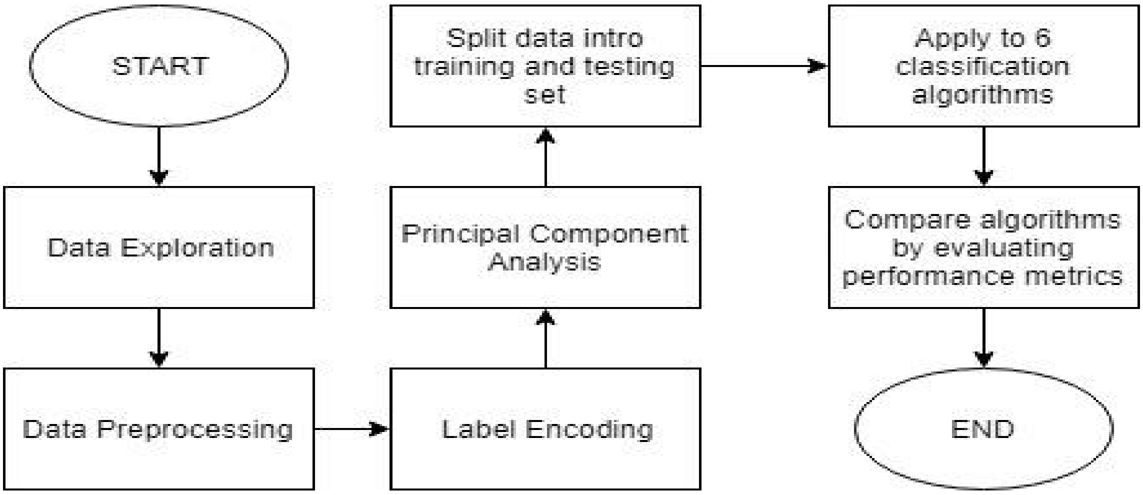


Fig 5.1.2 EXPLORATORY DATA ANALYSIS

5.1.3 MODULE NAME: DATA PREPROCESSING

Preprocessing of dataset In this module, the cleaned dataset will be preprocessed where the dataset will be divided based on the Given Data.



5.1.3 DATA PREPROCESSING

5.1.4 MODULE NAME: SPLITTING OF DATA

Dataset Partition In this module first the dataset will be divided into two partitions a trained dataset and a testing dataset. After the data partitions, the Random Forest Algorithm is applied. After applying RFA finally, a confusion matrix is obtained.

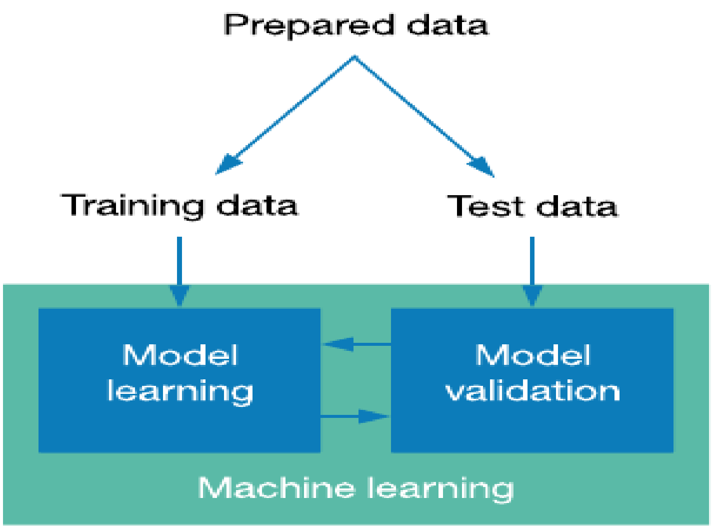


Fig 5.1.4 SPLITTING OF DATA

5.1.5 MODULE NAME: PREDICTING THE ACCURACY

Evaluation Now the resultant data obtained in the form of a confusion matrix can be evaluated by using graphical representation which gives better accuracy

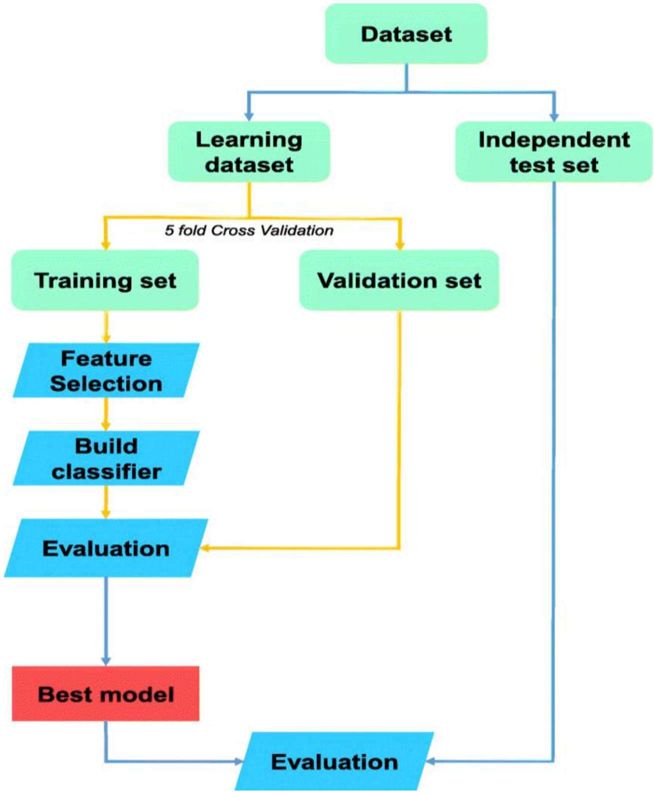


Fig 5.1.5 PREDICTING THE ACCURACY

CHAPTER 6

SYSTEM REQUIREMENTS

SOFTWARE REQUIREMENTS

PROGRAMMING LANGUAGE: PYTHON

IDE: GOOGLE COLAB

TECHNOLOGIES USED: MACHINE LEARNING

HARDWARE REQUIREMENTS

Processor - i5-10th Gen

Speed - 1.6 GHz

RAM - 8 GB

Hard Disk - 512 GB

CHAPTER 7

7.1 CONCLUSION

This study implemented the water quality model using the Random Forest technique. The analysis of water Alkalinity, pH level, and conductivity can play a major role in assessing water quality. Although random forest obtains good results on small set data, some problems exist, such as imbalanced data. Our future work will focus on solving these problems. The algorithm of the random forest itself should be improved. For example, the voting mechanism assumes that each of the base classifiers has equal weight, but some of them may be more important than others. Therefore, we also try to make some improvements to this algorithm. By using the Random Forest algorithm we got an accuracy of 70% and to enhance that accuracy we have used ensemble techniques. Finally, we have predicted high accuracy when compared to other algorithm models.

7.2 FUTURE SCOPE

This section shows the details and results of the experiments. Firstly, a performance comparison is made on the same subset. Then we explore the relation between a model's performance and the ratio of legal and fraud transactions in a subset. Finally, it shows the performances of models on a much bigger dataset, which is more closed to the actual result.

CHAPTER 8

Appendix

SAMPLE CODE

Importing The Libraries

In [95]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import re
from sklearn.ensemble import ExtraTreesRegressor
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import RandomizedSearchCV
from sklearn.metrics import r2_score
```

Importing The Dataset

In [96]:
```
df_water = pd.read_csv("/content/drive/MyDrive/MINOR PROJECT/water_potability.csv")
```

In [97]:
```
df_water.head()
```

Out[97]:

	ph	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic_carbon	Trihalomethanes	Turbidity	Potability
0	NaN	204.890455	20791.318981	7.300212	368.516441	564.308654	10.379783	86.990970	2.963135	0
1	3.716080	129.422921	18630.057858	6.635246	NaN	592.885359	15.180013	56.329076	4.500656	0
2	8.099124	224.236259	19909.541732	9.275884	NaN	418.606213	16.868637	66.420093	3.055934	0
3	8.316766	214.373394	22018.417441	8.059332	356.886136	363.266516	18.436524	100.341674	4.628771	0
4	9.092223	181.101509	17978.986339	6.546600	310.135738	398.410813	11.558279	31.997993	4.075075	0

Exploratory Data Analysis (DE, DC, DM, DV)

In [99]:
```
df_water.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3276 entries, 0 to 3275
Data columns (total 10 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   ph               2785 non-null   float64
 1   Hardness         3276 non-null   float64
 2   Solids           3276 non-null   float64
 3   Chloramines      3276 non-null   float64
 4   Sulfate          2495 non-null   float64
 5   Conductivity     3276 non-null   float64
 6   Organic_carbon   3276 non-null   float64
 7   Trihalomethanes  3114 non-null   float64
 8   Turbidity        3276 non-null   float64
 9   Potability       3276 non-null   int64  
dtypes: float64(9), int64(1)
memory usage: 256.1 KB
```

In [100…
```
df_water.shape
```

Out[100…
```
(3276, 10)
```

In [101…
```
df_water.isnull().sum()
```

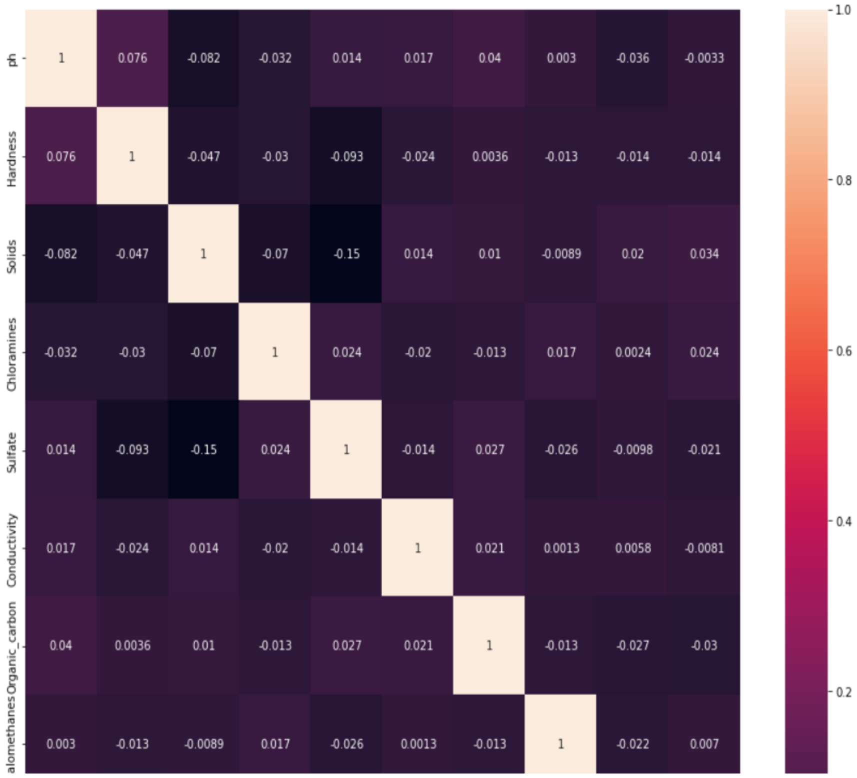
Out[101…
```
ph                 491
Hardness             0
Solids               0
Chloramines          0
Sulfate            781
Conductivity         0
Organic_carbon       0
Trihalomethanes    162
Turbidity            0
Potability           0
dtype: int64
```

In [123…

```
plt.figure(figsize=(15,15))
sns.heatmap(df_water.corr(),annot=True)
```

Out[123…

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f419da4d450>
```



Feature Selection

In [127…

```
model = ExtraTreesRegressor()
feat_imp = model.fit(X,y)
```

In [128…

```
feat_imp.feature_importances_
```

Out[128…

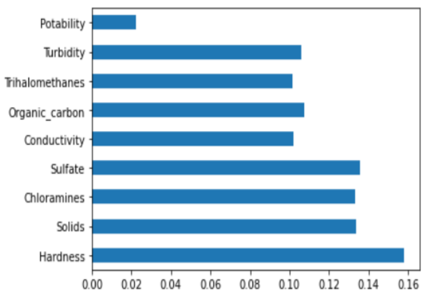
```
array([0.15788927, 0.13353025, 0.13334933, 0.13592055, 0.10185571,
       0.10727123, 0.10173808, 0.10596975, 0.02247582])
```

In [129…

```
imp = pd.Series(feat_imp.feature_importances_,index=X.columns)
imp.plot(kind='barh')
```

Out[129…

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f419d6627d0>
```



Splitting The Data

In [131…

```
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.2, random_state=0)
```

In [132…

```
X_train
```

Out[132…

	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic_carbon	Trihalomethanes	Turbidity	Potability
2128	228.735924	35343.628580	4.346608	333.775777	526.112381	14.930982	46.780508	2.798158	0
1519	210.732854	13671.416030	8.546187	418.470551	352.252328	10.353659	45.304007	3.364891	1
40	233.858996	11703.923907	4.599388	309.039320	349.399633	18.338893	42.677465	3.510004	0
1151	227.007086	7323.302301	7.490508	326.695199	412.896404	12.906730	68.748918	2.010537	1
2404	217.372780	25175.754158	9.883946	329.174454	394.054835	20.277571	85.840258	2.615257	1
...	...	...	...	...	...	...	...	...	...
835	183.362713	17259.852302	4.610245	335.626443	452.995293	9.700906	80.537065	2.496343	0
3264	239.269481	20526.666156	6.349561	341.256362	403.617560	18.963707	63.846319	4.390702	1
1653	191.841801	15176.290678	5.661663	333.775777	471.047129	15.438287	56.532387	3.829784	0
2607	233.300759	23673.100606	8.407497	333.775777	232.613624	18.459408	60.993590	5.040461	0
2732	160.915815	13943.244974	8.399730	380.768478	344.154228	15.208691	75.575056	4.141552	1

2620 rows × 9 columns

In [133…

```
X_test
```

Out[133…

	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic_carbon	Trihalomethanes	Turbidity	Potability
2017	217.266472	38184.469574	7.254122	311.910224	281.069203	13.027921	78.582094	4.430750	1
2533	179.805992	23793.031358	5.332099	333.198191	461.530446	13.557381	60.571241	4.145807	0
589	180.893036	17705.608616	6.223312	350.195253	447.937123	10.461025	32.074863	3.999125	0
482	178.922858	18476.619166	8.226228	334.889911	518.043369	10.638798	63.157489	3.861956	0

Model Selection

In [134…

```
regressor = RandomForestRegressor()
```

Hyper Parameter Tuning

In [135…

```
n_estimators = [int(i) for i in np.linspace(start=100, stop=1200, num=12)]

max_features =  ['auto', 'sqrt']

max_depth = [int(i) for i in np.linspace(start=5, stop=30, num=6)]

min_samples_leaf = [1, 2, 5, 10]
```

In [136…

```
random_grid = {'n_estimators':n_estimators,
               'max_features':max_features,
               'max_depth':max_depth,
               'min_samples_leaf':min_samples_leaf}
print(random_grid)
```

```
{'n_estimators': [100, 200, 300, 400, 500, 600, 700, 800, 900, 1000, 1100, 1200], 'max_features': ['auto', 'sqrt'], 'max_depth': [5, 10, 15, 20, 25, 3
0], 'min_samples_leaf': [1, 2, 5, 10]}
```

In [137…

```
rf_regressor = RandomizedSearchCV(estimator=regressor,
                                  param_distributions=random_grid,
                                  scoring='neg_mean_squared_error',
                                  cv=5,
                                  verbose = 2,
                                  random_state = 42,
                                  n_jobs = 1)
```

Training The Model

In [138…

```
rf_regressor.fit(X_train, y_train)
```

Training The Model

In [138…

```
rf_regressor.fit(X_train, y_train)
```

```
Fitting 5 folds for each of 10 candidates, totalling 50 fits
[CV] END max_depth=15, max_features=auto, min_samples_leaf=10, n_estimators=700; total time=   7.2s
[CV] END max_depth=15, max_features=auto, min_samples_leaf=10, n_estimators=700; total time=   7.0s
[CV] END max_depth=15, max_features=auto, min_samples_leaf=10, n_estimators=700; total time=   7.0s
[CV] END max_depth=15, max_features=auto, min_samples_leaf=10, n_estimators=700; total time=   7.0s
[CV] END max_depth=15, max_features=auto, min_samples_leaf=10, n_estimators=700; total time=   7.1s
[CV] END max_depth=10, max_features=auto, min_samples_leaf=2, n_estimators=1100; total time=  11.7s
[CV] END max_depth=10, max_features=auto, min_samples_leaf=2, n_estimators=1100; total time=  12.2s
[CV] END max_depth=10, max_features=auto, min_samples_leaf=2, n_estimators=1100; total time=  11.6s
[CV] END max_depth=10, max_features=auto, min_samples_leaf=2, n_estimators=1100; total time=  11.7s
[CV] END max_depth=10, max_features=auto, min_samples_leaf=2, n_estimators=1100; total time=  13.2s
[CV] END max_depth=20, max_features=sqrt, min_samples_leaf=1, n_estimators=1100; total time=   7.5s
[CV] END max_depth=20, max_features=sqrt, min_samples_leaf=1, n_estimators=1100; total time=   7.5s
[CV] END max_depth=20, max_features=sqrt, min_samples_leaf=1, n_estimators=1100; total time=   7.6s
[CV] END max_depth=20, max_features=sqrt, min_samples_leaf=1, n_estimators=1100; total time=   7.5s
[CV] END max_depth=20, max_features=sqrt, min_samples_leaf=1, n_estimators=1100; total time=   7.5s
[CV] END max_depth=30, max_features=auto, min_samples_leaf=2, n_estimators=700; total time=  10.5s
[CV] END max_depth=30, max_features=auto, min_samples_leaf=2, n_estimators=700; total time=  10.1s
[CV] END max_depth=30, max_features=auto, min_samples_leaf=2, n_estimators=700; total time=  10.1s
[CV] END max_depth=30, max_features=auto, min_samples_leaf=2, n_estimators=700; total time=  10.3s
[CV] END max_depth=30, max_features=auto, min_samples_leaf=2, n_estimators=700; total time=  10.2s
[CV] END max_depth=25, max_features=auto, min_samples_leaf=2, n_estimators=700; total time=  10.4s
[CV] END max_depth=25, max_features=auto, min_samples_leaf=2, n_estimators=700; total time=  10.1s
[CV] END max_depth=25, max_features=auto, min_samples_leaf=2, n_estimators=700; total time=  16.6s
[CV] END max_depth=25, max_features=auto, min_samples_leaf=2, n_estimators=700; total time=  14.6s
[CV] END max_depth=25, max_features=auto, min_samples_leaf=2, n_estimators=700; total time=  15.7s
[CV] END max_depth=25, max_features=auto, min_samples_leaf=2, n_estimators=500; total time=   7.4s
[CV] END max_depth=25, max_features=auto, min_samples_leaf=2, n_estimators=500; total time=   7.1s
[CV] END max_depth=25, max_features=auto, min_samples_leaf=2, n_estimators=500; total time=   7.2s
[CV] END max_depth=25, max_features=auto, min_samples_leaf=2, n_estimators=500; total time=   7.3s
[CV] END max_depth=25, max_features=auto, min_samples_leaf=2, n_estimators=500; total time=   7.3s
[CV] END max_depth=15, max_features=sqrt, min_samples_leaf=2, n_estimators=800; total time=   4.6s
[CV] END max_depth=15, max_features=sqrt, min_samples_leaf=2, n_estimators=800; total time=   4.5s
[CV] END max_depth=15, max_features=sqrt, min_samples_leaf=2, n_estimators=800; total time=   4.5s
[CV] END max_depth=15, max_features=sqrt, min_samples_leaf=2, n_estimators=800; total time=   4.6s
[CV] END max_depth=15, max_features=sqrt, min_samples_leaf=2, n_estimators=800; total time=   4.6s
[CV] END max_depth=5, max_features=auto, min_samples_leaf=5, n_estimators=600; total time=   4.0s
[CV] END max_depth=5, max_features=auto, min_samples_leaf=5, n_estimators=600; total time=   3.9s
[CV] END max_depth=5, max_features=auto, min_samples_leaf=5, n_estimators=600; total time=   3.9s
```

Predicting The Model

In [140…

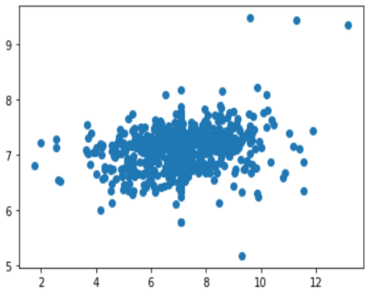
```
y_pred = rf_regressor.predict(X_test)
```

In [141…

```
plt.scatter(y_test, y_pred)
```

Out[141…

```
<matplotlib.collections.PathCollection at 0x7f419d91a590>
```



In [142…

```
finaldf = pd.DataFrame({"Actual":y_test, "Predicted":y_pred})
```

In [143…

```
finaldf
```

Out[143…

	Actual	Predicted
2017	8.111953	7.100015
2533	6.768060	6.706596
589	7.080795	7.050122
482	7.705711	6.762511
2620	5.830542	6.672074
...	...	...

2620	5.830542	6.672074
...	...	...
2802	5.230454	6.480071
1120	7.444914	7.368170
601	7.080795	7.335074
2381	4.945695	6.617878
98	7.205559	6.839946

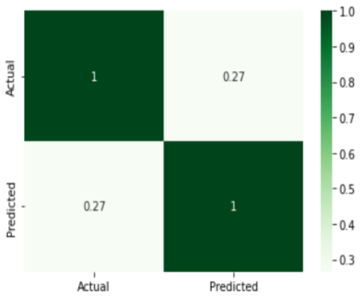
656 rows × 2 columns

In [144…

```
sns.heatmap(finaldf.corr(),annot=True, cmap='Greens')
```

Out[144…

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f419da5e650>
```



Perfomance/Accuracy

In [147…

```
r2_score(y_test, y_pred)
```

Out[147…

```
0.07047462866991772
```

CHAPTER 9

REFERENCES

1. Jiang, J.; Tang, S.; Han, D.; Fu, G.; Solomatin, D.; Zheng, Y. A comprehensive review on the design and optimization of surface water quality monitoring networks. Environ. Model. Soft. 2020, 132, 104792.

2. Zhou, Y.; Yu, D.; Yang, Q.; Pan, S.; Gai, Y.; Cheng, W.; Liu, X.; Tang, S. Variations of Water Transparency and Impact Factors in the Bohai and Yellow Seas from Satellite Observations. Remote Sens. 2021, 13, 514.

3.Mateo Pérez, V.; Mesa Fernández, J.M.; Villanueva Balsera, J.; Alonso Álvarez, C. A Random Forest Model for the Prediction of FOG Content in Inlet Wastewater from Urban WWTPs. Water 2021, 13, 1237.

4. Zhou, Z.H. Ensemble learning. In Machine Learning; Springer: Berlin/Heidelberg, Germany, 2021; pp. 181–210

5. Karami, J.; Alimohammadi, A.; Seifouri, T. Water quality analysis using a variable consistency dominance-based rough set approach. Comput. Environ. Urban Syst. 2014, 43, 25–33.

CHAPTER 10

PLAGIARISM REPORT

Fig 12.1

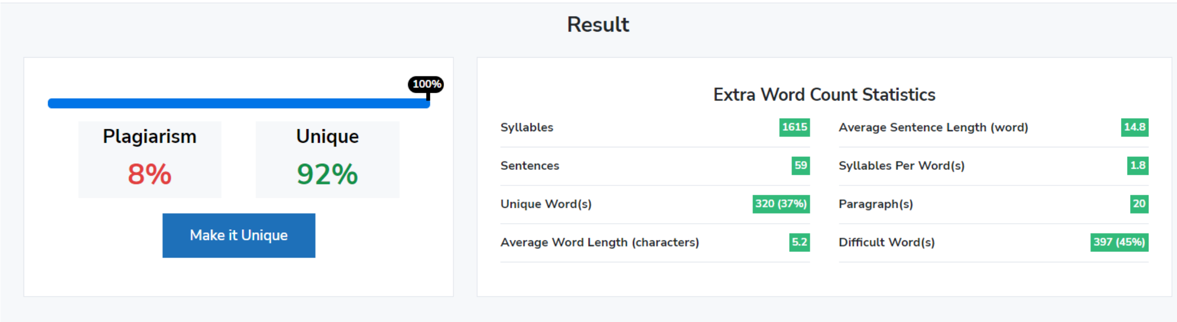


Fig 12.2