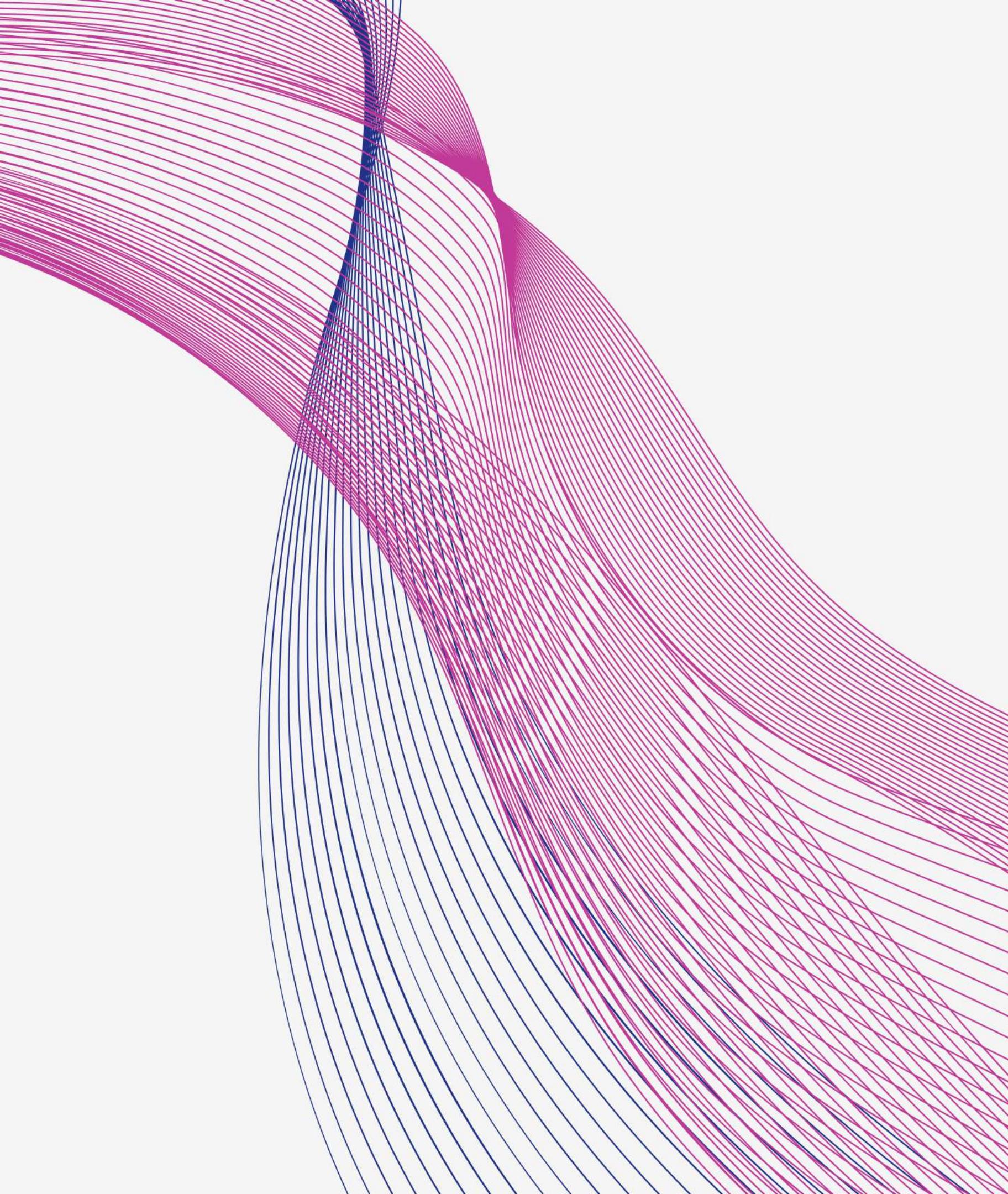
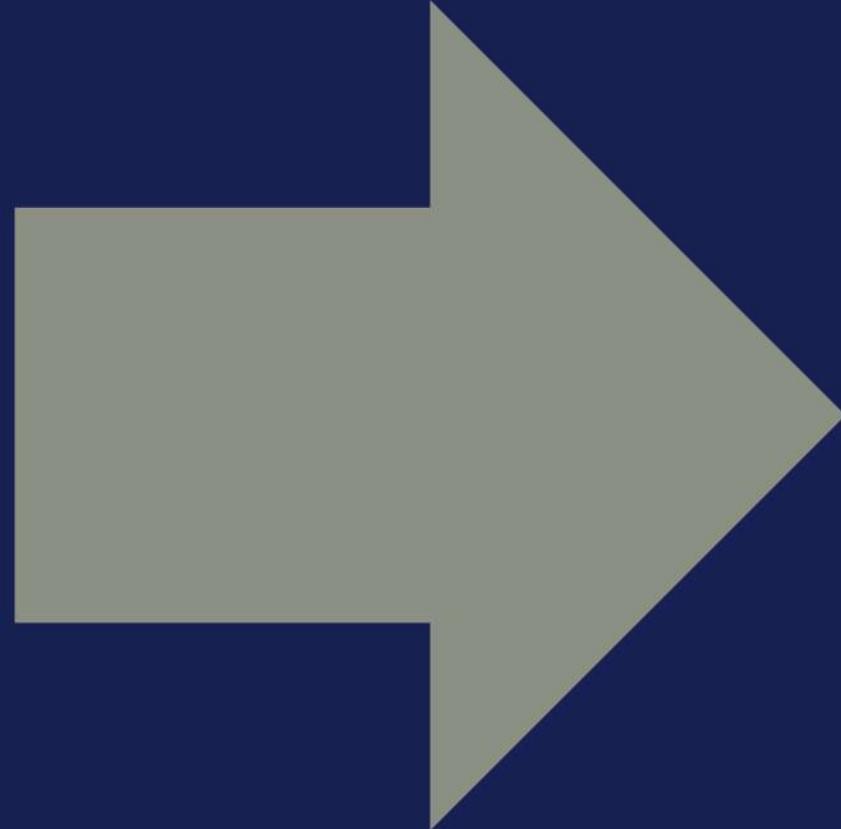


WATER QUALITY ANALYSIS





IBM COGNOS



Edit More Filter Link Unlink Copy PropertiesAnalytics More Filters Fields Properties

Selected sources /

water_potability.csv + :

Search

Navigation paths +

water_potability.csv

ph

Hardness

Solids

Chloramines

Sulfate

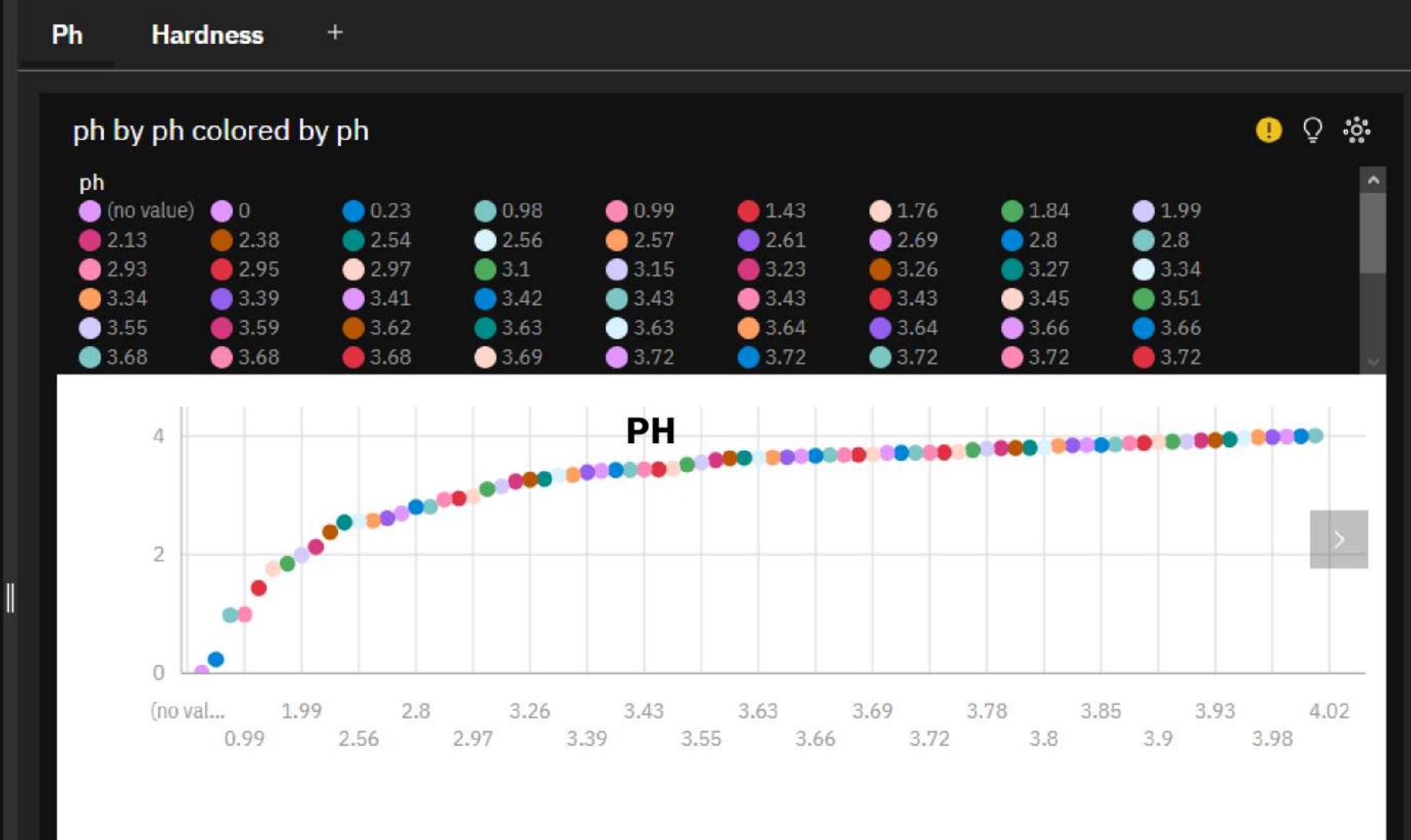
Conductivity

Organic_carbon

Trihalomethanes

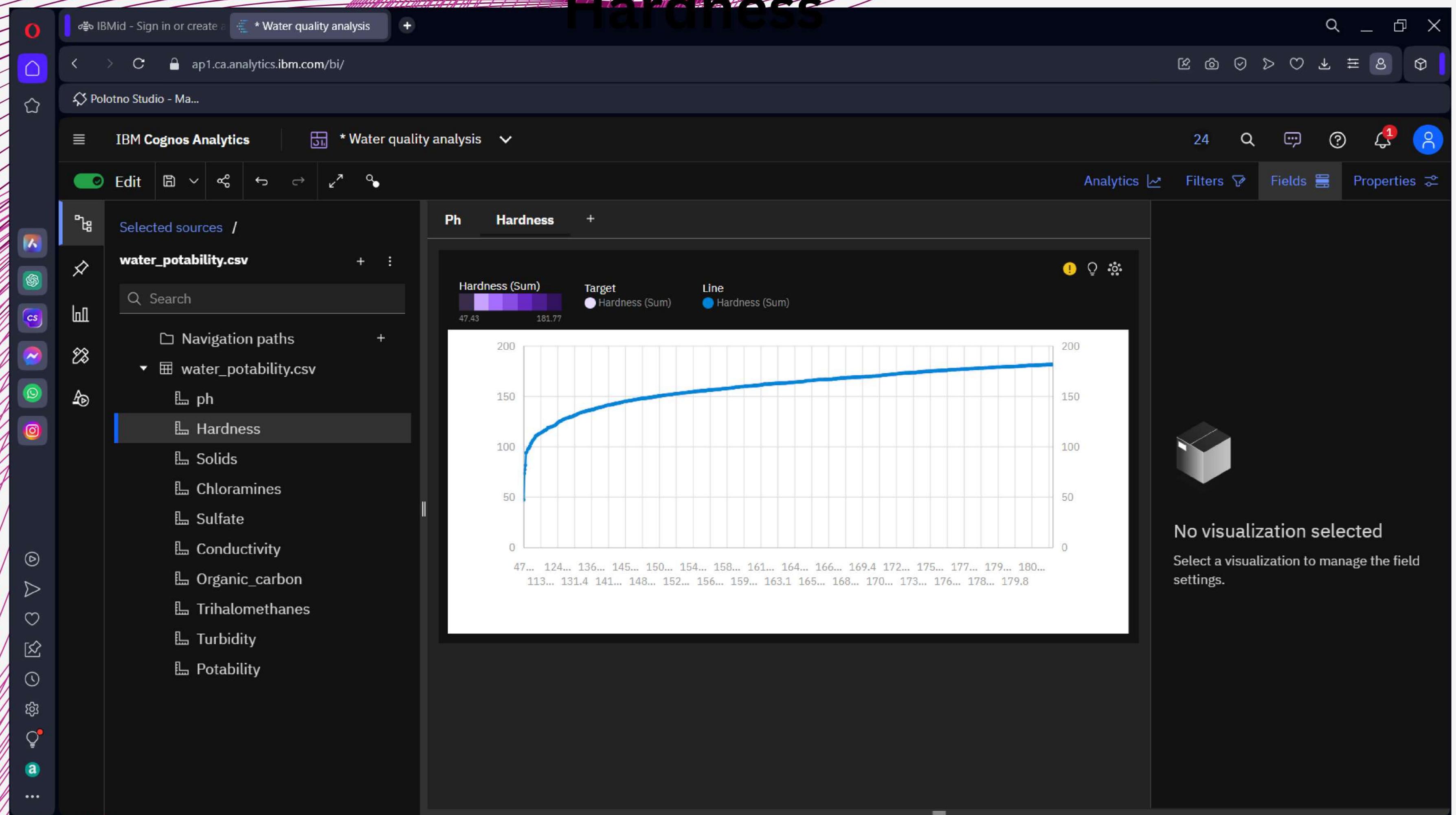
Turbidity

Potability



No visualization selected

Select a visualization to manage the field settings.



Hardness & solids

Mi IBM * Water quality analysis +

Water quality analysis

Polotno Studio - Ma...

IBM Cognos Analytics * Water quality analysis

Edit | Analytics | Filters | Fields | Properties

Selected sources /

water_potability.csv

Search

Navigation paths

water_potability.csv

ph

Hardness

Solids

Chloramines

Sulfate

Conductivity

Organic_carbon

Trihalomethanes

Turbidity

Potability

Ph hardness & solids +

Hardness by Hardness

! ? ☰ Solids to Solids with line width Solids

7,845.47 13,404.25
.18,257.57 .19,116.01 .19,871.79
.18,344.68 .27,884.43 .24,752.07
.18,908.56 .21,536.22 .15,288.21 .31,561.96
.24,856.63 .21,410.32 .28,049.65 .44,506.44
.21,174.41 .22,018.42 .28,544.62 .23,318.19 .28,873.38
.24,683.72 .32,653.91 .23,564.58 .26,736.09 .7,035.13
.30,565.58 .19,620.55 .15,193.41 .35,216.39 .38,200.8 .21,829
.20,687.87 .25,456.27 .27,150.4 .29,368.67 .19,646.23 .14,101.87 .19,909
.40,710.52 .9,615.83 .30,646.82 .24,656.24 .20,868.33 .14,285.58 .19,82
.12,285.89 .13,990.51 .28,608.05 .25,840
.21,568.43 .20,434.11 .34,864.43 .22,452.09 .14,904.9
.28,897.49 .9,381.24 .41,839.46 .18,767.66 .26,244.0
.27,492.31 .27,141.36 .20,571.85 .28,827.36 .24,468.05 .17,16
.25,508.39 .27,355.93 .21,167.5 .36,957.79 .9,898.02 .28,062.64
.37,962.17 .15,249.62 .16,312.15 .31,461.2 .17,358.8
.39,742.97 .39,742.97 .22,824.7 .15,130.15

No visualization selected
Select a visualization to manage the field settings.

Mi IBM * Water quality analysis +

C ap1.ca.analytics.ibm.com/bi/ Polotno Studio - Ma...

IBM Cognos Analytics * Water quality analysis 24 ? 1

Edit Analytics Filters Fields Properties

Selected sources /

water_potability.csv + :

Search

Navigation paths +

water_potability.csv

- ph
- Hardness
- Solids
- Chloramines
- Sulfate
- Conductivity
- Organic_carbon
- Trihalomethanes
- Turbidity
- Potability

Ph hardness & solids Tab 1 +

Chloramines and Chloramines with Chloramines and Chloramines for Chloramines

Sulfate

Chloramines (Sum) Chloramines (Sum)

0.35 6.36 0.35 6.36

1.39 2.39 2.48 2.62 2.74 2.86
0.35 1.92 2.46 2.56 2.65 2.79 2.98

No visualization selected
Select a visualization to manage the field settings.

The screenshot shows the IBM Cognos Analytics interface. On the left, there's a sidebar with various icons and a list of selected sources under 'water_potability.csv'. The main area displays a visualization titled 'Chloramines and Chloramines with Chloramines and Chloramines for Chloramines' and 'Sulfate'. It includes two color scales for 'Chloramines (Sum)' and 'Chloramines (Sum)' with ranges from 0.35 to 6.36. Below these are numerical values: 1.39, 2.39, 2.48, 2.62, 2.74, 2.86, 0.35, 1.92, 2.46, 2.56, 2.65, 2.79, 2.98. To the right, there's a large circular visualization with a grid of purple dots. A message at the bottom right says 'No visualization selected' and 'Select a visualization to manage the field settings.'

Conductivity & Organic_carbon

Mi IBM * Water quality analysis IBM Naanmudhalvan Que +

ap1.ca.analytics.ibm.com/bi/ Polotno Studio - Ma... IBM Cognos Analytics * Water quality analysis Edit Analytics Filters Fields Properties

Selected sources / water_potability.csv + : Search

Navigation paths +

- water_potability.csv
 - ph
 - Hardness
 - Solids
 - Chloramines
 - Sulfate
 - Conductivity
 - Organic_carbon
 - Trihalomethanes
 - Turbidity
 - Potability

Ph hardness & solids chloramines & sulfate Tab 1 +

Conductivity by Conductivity colored by Conductivity

Conductivity

181.48	201.62	210.32	217.36	232.61
233.91	235.04	245.86	247.92	251.02
252.97	254.39	254.39	256.3	257.01
257.7	257.96	258.88	259.64	259.96
260.53	260.57	261.44	263.77	264.28

Organic_carbon by Organic_carbon

No visualization selected
Select a visualization to manage the field settings.

Turbidity & Potability

IBM Cognos Analytics | Water quality analysis

Selected sources / water_potability.csv

Ph hardness & solids chloramines & sulfate conductivity & organic_carbon Tab 1 +

Trihalomethanes by Trihalomethanes with points for Trihalomethanes ! ? ⓘ Potability by Potability ⓘ

Trihalomethanes (...

8.18 124

150
100
50
0

0 20 40 60 80 100 120 140

Potability
0 1

1,278

No visualization selected
Select a visualization to manage the field settings.

Water quality analysis

IBM Naan udh... ue

ap1.ca.analytics.ibm.com/bi/

Polotno Studio - Ma...

IBM Cognos Analytics

Edit

Analytics

Filters

Fields

Properties

24

Search

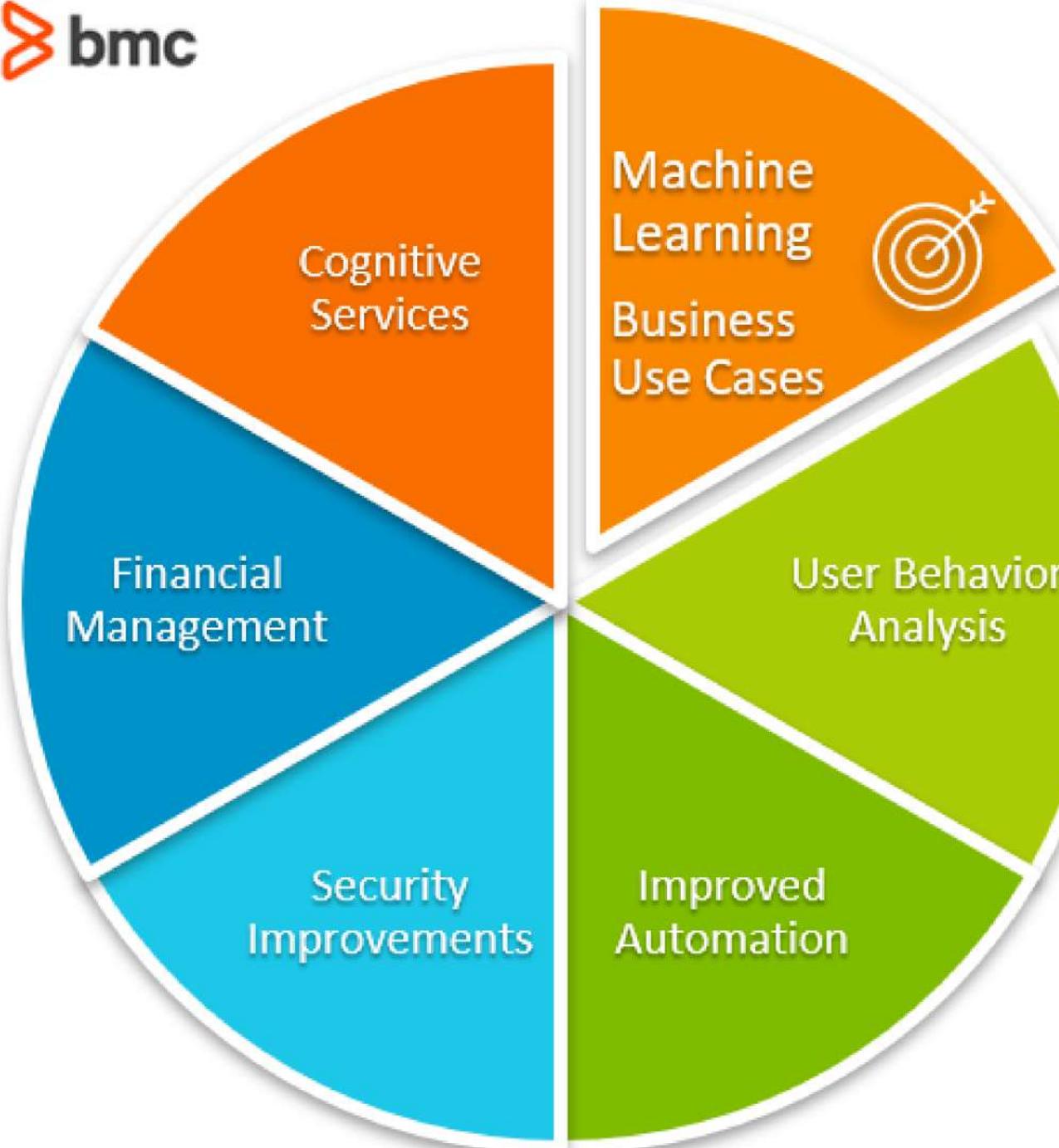
Navigation paths

water_potability.csv

- ph
- Hardness
- Solids
- Chloramines
- Sulfate
- Conductivity
- Organic_carbon
- Trihalomethanes
- Turbidity
- Potability

Search

...



IMPLEMENTATION

Analyzing water quality using Machine Learning (ML) algorithms involves leveraging computational techniques to process and interpret water quality data, predict parameters, and detect anomalies. Here's a step-by-step approach to implementing water quality analysis using ML algorithms:

MACHINE LEARNING ALGORITHMS: THAT WE ARE GOING TO USE

- Logistic Regression
- Decision Tree
- Random Forest
- K-Nearest Neighbours
- Support Vector Machine

WHOLE DATA SET REQUIRED



Data Preprocessing:

Handle missing values, outliers, and any inconsistencies in the dataset.
Perform feature scaling, normalization, or transformation as needed.



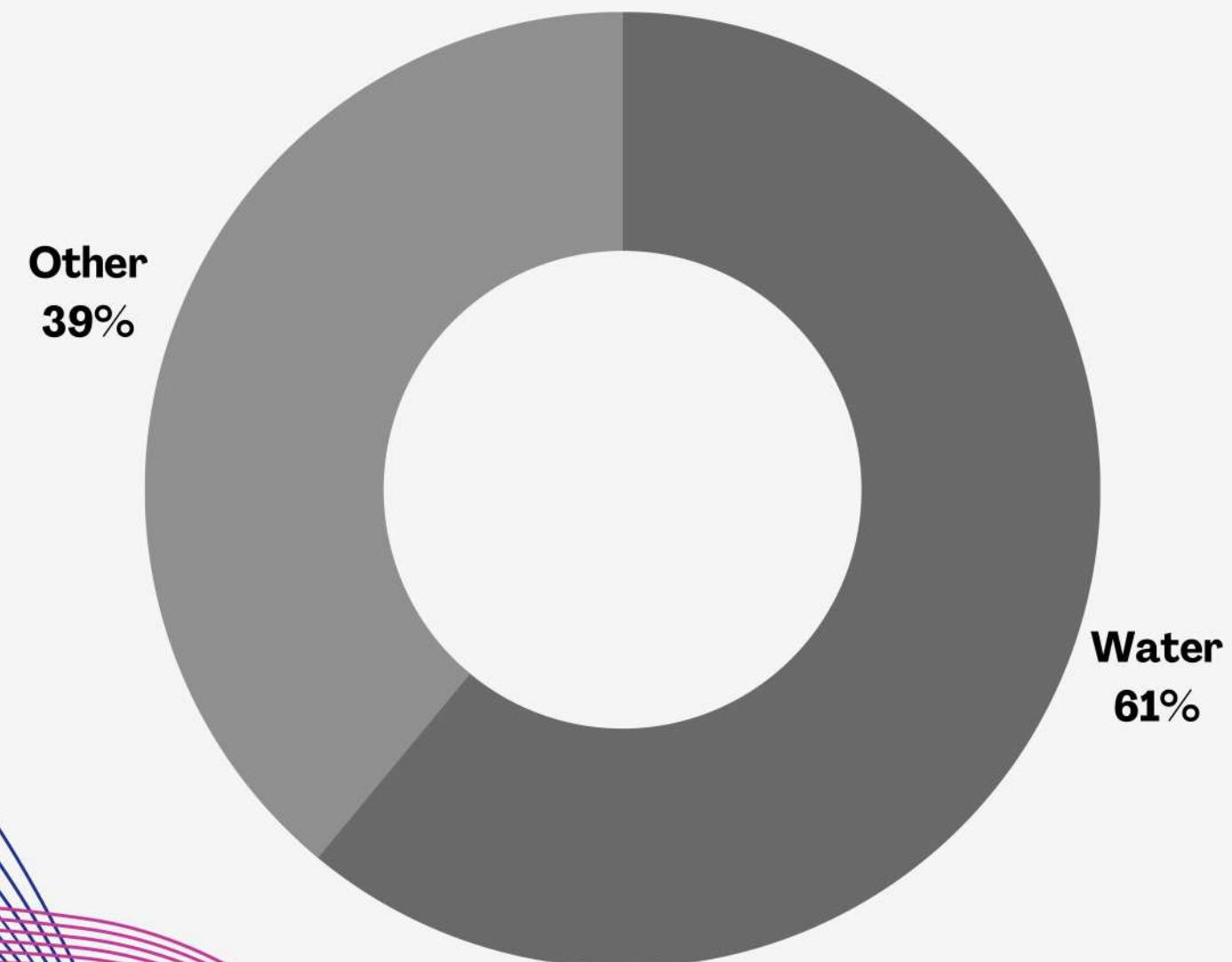
Feature Selection:

Identify relevant features that contribute to the water quality analysis using techniques like correlation analysis, feature importance, or domain knowledge.



Model Training:

Split the dataset into training and testing sets.
Train the selected ML models using the training set and validate them using the testing set.

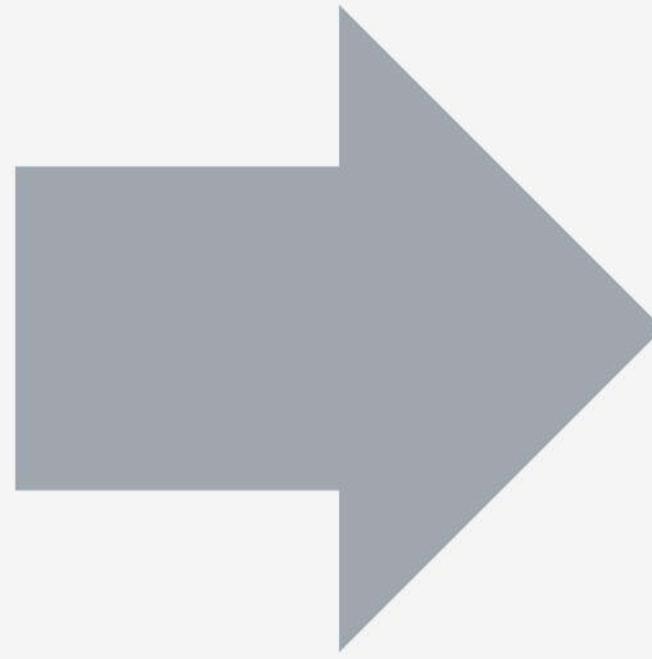


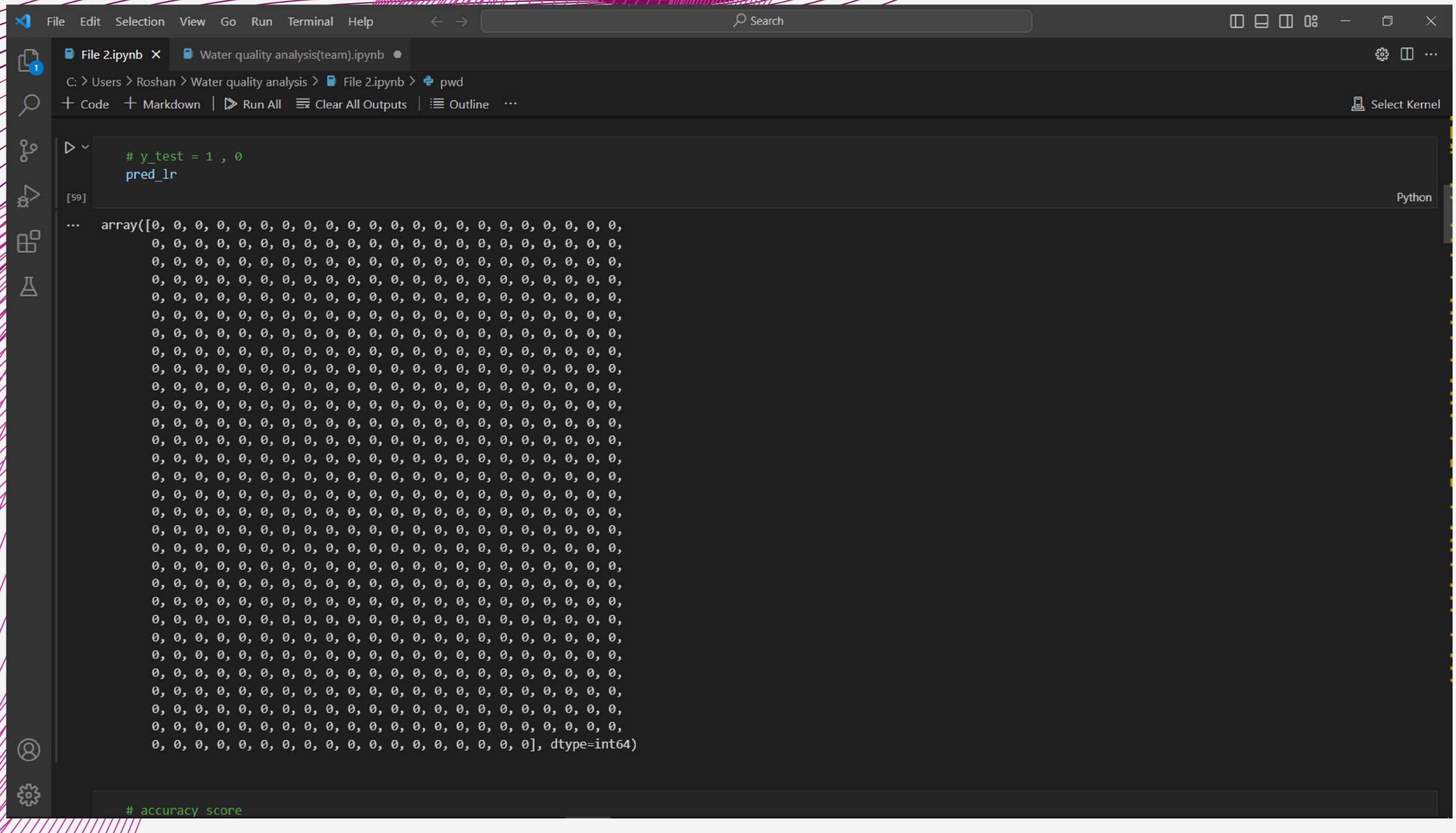
POTABILITY OF WATER

Machine Learning project built to practice and improve coding and deployment skills using python, scikit-learn, jupyter-notebooks, and some visualization packages.



.ipynb PROGRAMS





The screenshot shows a Jupyter Notebook interface with a dark theme. The top bar includes File, Edit, Selection, View, Go, Run, Terminal, Help, a search bar, and window control buttons. The left sidebar has icons for file operations, search, and settings.

The notebook displays two files: "File 2.ipynb" and "Water quality analysis(team).ipynb". The current cell is "File 2.ipynb". The path in the header is "C: > Users > Roshan > Water quality analysis > File 2.ipynb".

The code in the notebook is:

```
# accuracy score
accuracy_score_lg = accuracy_score(y_test,pred_lr)
accuracy_score

[60] ... <function sklearn.metrics._classification.accuracy_score(y_true, y_pred, *, normalize=True, sample_weight=None)>

# accuracy score
accuracy_score_lg = accuracy_score(y_test,pred_lr)
(function) def accuracy_score(
    y_true: MatrixLike,
    y_pred: MatrixLike,
    *,
    normalize: bool = True,
    sample_weight: Any | None = None
) -> Float

[61] ... 0.6097560975609756

# accuracy score
accuracy_score_lr = accuracy_score(y_test,pred_lr)
accuracy_score_lr

[63] ... 0.6097560975609756
```

The code at [61] shows a tooltip for the `accuracy_score` function from `sklearn.metrics._classification`.

Decision Tree Classifier

```
from sklearn.tree import DecisionTreeClassifier

# creating the model object
model_dt = DecisionTreeClassifier(max_depth = 4)
```

[64] Python

The screenshot shows a Jupyter Notebook interface with a dark theme. The top bar includes File, Edit, Selection, View, Go, Run, Terminal, Help, a search bar, and window control buttons. The left sidebar has icons for file operations, search, and settings.

The notebook displays three code cells:

- Cell 1:** [60] Python
accuracy score
accuracy_score_lg = accuracy_score(y_test,pred_lr)
accuracy_score
- Cell 2:** [61] Python
... <function sklearn.metrics._classification.accuracy_score>

accuracy score
accuracy_score_lg = accuracy_score(y_test,pred_lr)
accuracy_score_lg
- Cell 3:** [63] Python
... 0.6097560975609756

accuracy score
accuracy_score_lr = accuracy_score(y_test,pred_lr)
accuracy_score_lr

Below the cells are buttons for '+ Code' and '+ Markdown'. The bottom section features a large title 'Decision Tree Classifier' and a code cell [64] Python:

```
from sklearn.tree import DecisionTreeClassifier  
  
# creating the model object  
model_dt = DecisionTreeClassifier(max_depth = 4)
```

File Edit Selection View Go Run Terminal Help ⏪ ⏩ Search

File 2.ipynb X Water quality analysis(team).ipynb ●

C: > Users > Roshan > Water quality analysis > File 2.ipynb > pwd

+ Code + Markdown | ▶ Run All | Clear All Outputs | Outline ... Select Kernel

Decision Tree Classifier

```
from sklearn.tree import DecisionTreeClassifier  
  
# creating the model object  
model_dt = DecisionTreeClassifier(max_depth = 4)
```

[64] Python

```
# Training of Decision Tree  
model_dt.fit(x_train,y_train)
```

[65] Python

```
... ▾ DecisionTreeClassifier  
DecisionTreeClassifier(max_depth=4)
```

```
# Making prediction using Decision Tree  
pred_dt = model_dt.predict(x_test)
```

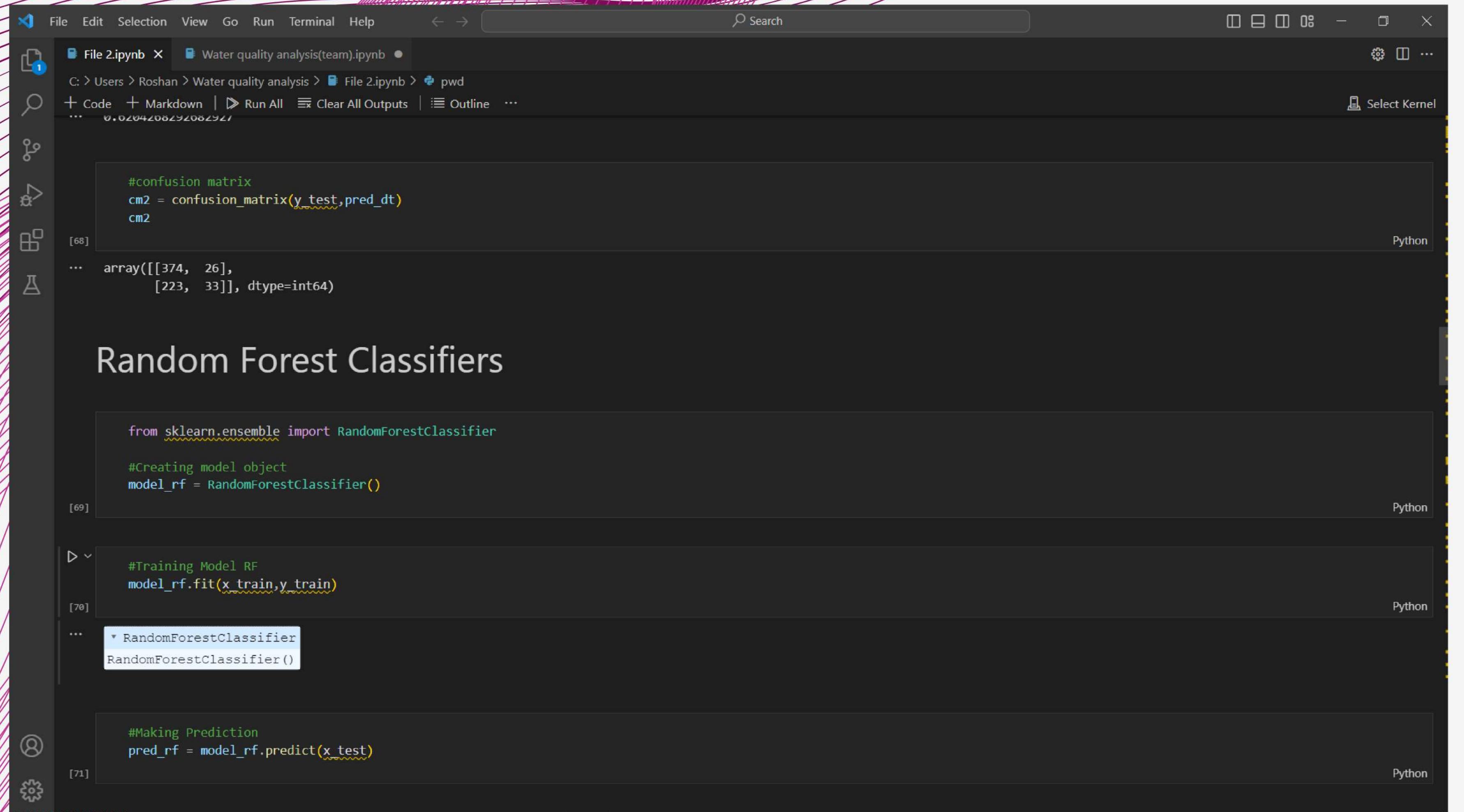
[66] Python

```
accuracy_score_dt = accuracy_score(y_test,pred_dt)  
accuracy_score_dt
```

[67] Python

```
... 0.6204268292682927
```

```
#confusion matrix  
cm2 = confusion_matrix(y_test,pred_dt)  
cm2
```



The screenshot shows a Jupyter Notebook interface with a dark theme. The top menu bar includes File, Edit, Selection, View, Go, Run, Terminal, Help, and a search bar. The left sidebar features icons for file operations like Open, Save, and New, along with a magnifying glass for search, a gear for settings, and a gear with a plus sign for kernel selection. The main workspace displays two code cells:

Cell 1 (Python):

```
accuracy_score_rf = accuracy_score(y_test,pred_rf)
accuracy_score_rf*100
```

[73] ... 65.39634146341463

Cell 2 (Python):

```
cm3 = confusion_matrix(y_test,pred_rf)
cm3
```

[74] ... array([[348, 52],
 [175, 81]], dtype=int64)

Below the cells are buttons for '+ Code' and '+ Markdown'. The bottom right corner of the interface has a small yellow progress bar.

KNN -- K-Neighbours

Cell 3 (Python):

```
from sklearn.neighbors import KNeighborsClassifier
#Creating Model object
model_knn = KNeighborsClassifier()
```

[76]

Cell 4 (Python):

```
for i in range(4,11):
    model_knn = KNeighborsClassifier(n_neighbors=i)
    model_knn.fit(x_train,y_train)
    pred_knn = model_knn.predict(x_test)
    accuracy_score_knn = accuracy_score(y_test,pred_knn)
    print(i,accuracy_score_knn)
```

[78] ... 4 0.6524390243902439

The screenshot shows a Jupyter Notebook interface with a dark theme. The top bar includes File, Edit, Selection, View, Go, Run, Terminal, Help, a search bar, and window control buttons. The left sidebar has icons for file operations like Open, Save, Find, and Kernel selection.

The notebook contains two code cells:

Cell 76 (Python):

```
for i in range(4,11):
    model_knn = KNeighborsClassifier(n_neighbors=i)
    model_knn.fit(x_train,y_train)
    pred_knn = model_knn.predict(x_test)
    accuracy_score_knn = accuracy_score(y_test,pred_knn)
    print(i,accuracy_score_knn)
```

Output 76:

```
... 4 0.6524390243902439
5 0.6341463414634146
6 0.6371951219512195
7 0.6189024390243902
8 0.6463414634146342
9 0.6341463414634146
10 0.6432926829268293
```

Cell 78 (Python):

```
model_knn = KNeighborsClassifier(n_neighbors=i)
model_knn.fit(x_train,y_train)
pred_knn = model_knn.predict(x_test)
accuracy_score_knn = accuracy_score(y_test,pred_knn)
print(accuracy_score_knn*100)
```

Output 78:

```
... 64.32926829268293
```

SVM

Cell 79 (Python):

```
from sklearn.svm import SVC
#Creating object of Model
```

File Edit Selection View Go Run Terminal Help ⌘ ⌘ ⌘ ⌘ ⌘ ⌘ ⌘ ⌘ ⌘

Search

File 2.ipynb X Water quality analysis(team).ipynb ●

C: > Users > Roshan > Water quality analysis > File 2.ipynb > pwd

+ Code + Markdown | ▶ Run All ⌘ Clear All Outputs | ⌘ Outline ⌘

Select Kernel

SVM

```
from sklearn.svm import SVC  
  
#Creating object of Model  
model_svm = SVC(kernel="rbf")
```

[81] Python

```
#Model training  
model_svm.fit(x_train,y_train)
```

[82] Python

```
...  
▼ SVC  
SVC ()
```

```
#Make prediction  
pred_svm = model_svm.predict(x_test)
```

[83] Python

```
accuracy_score_svm = accuracy_score(y_test,pred_svm)  
accuracy_score_svm*100
```

[84] Python

```
... 67.53048780487805
```

AdaBoost Classifiers

File Edit Selection View Go Run Terminal Help ⌘ ⌘ ⌘ ⌘ ⌘ ⌘ ⌘ ⌘ ⌘

File 2.ipynb X Water quality analysis(team).ipynb ●

C: > Users > Roshan > Water quality analysis > File 2.ipynb > pwd

+ Code + Markdown | ▶ Run All ⌘ Clear All Outputs | ⌘ Outline ⌘

Select Kernel

AdaBoost Classifiers

```
from sklearn.ensemble import AdaBoostClassifier  
  
#Making object of model  
model_ada = AdaBoostClassifier(n_estimators=200,learning_rate=0.03)
```

[118] Python

```
#Training the model  
model_ada.fit(x_train,y_train)
```

[119] Python

```
...  
AdaBoostClassifier  
AdaBoostClassifier(learning_rate=0.03, n_estimators=200)
```

```
#Making prediction  
pred_ada = model_ada.predict(x_test)
```

[120] Python

```
#accuracy check  
accuracy_score_ada = accuracy_score(y_test,pred_ada)  
accuracy_score_ada*100
```

[121] Python

```
... 61.28048780487805
```

Python

File Edit Selection View Go Run Terminal Help ⏪ ⏩ Search

File 2.ipynb X Water quality analysis(team).ipynb ●

C: > Users > Roshan > Water quality analysis > File 2.ipynb > pwd

+ Code + Markdown | ▶ Run All ⌛ Clear All Outputs | ⌂ Outline ... Select Kernel

XGBoost

```
from xgboost import XGBClassifier  
  
# Create model  
model_xgb = XGBClassifier(n_estimators=200,learning_rate=0.03)  
[131] Python
```

```
#Training Model  
model_xgb.fit(x_train,y_train)  
[132] Python
```

```
...  
XGBClassifier  
XGBClassifier(base_score=None, booster=None, callbacks=None,  
    colsample_bylevel=None, colsample_bynode=None,  
    colsample_bytree=None, device=None, early_stopping_rounds=None,  
    enable_categorical=False, eval_metric=None, feature_types=None,  
    gamma=None, grow_policy=None, importance_type=None,  
    interaction_constraints=None, learning_rate=0.03, max_bin=None,  
    max_cat_threshold=None, max_cat_to_onehot=None,  
    max_delta_step=None, max_depth=None, max_leaves=None,  
    min_child_weight=None, missing=nan, monotone_constraints=None,  
    multi_strategy=None, n_estimators=200, n_jobs=None,  
    num_parallel_tree=None, random_state=None, ...)
```

```
#Prediction  
pred_xgb = model_xgb.predict(x_test)  
[133] Python
```

```
#accuracy  
accuracy_score_xgb = accuracy_score(y_test,pred_xgb)  
accuracy_score_xgb*100
```

File Edit Selection View Go Run Terminal Help ⏪ ⏩ Search

File 2.ipynb X Water quality analysis(team).ipynb ●

C: > Users > Roshan > Water quality analysis > File 2.ipynb > pwd

+ Code + Markdown | ▶ Run All | Clear All Outputs | Outline ...

accuracy_score_xgb = accuracy_score(y_test,pred_xgb)
accuracy_score_xgb*100

[134] Python

... 64.48170731707317

models = pd.DataFrame({
 "Model": ["Logistic Regression",
 "Decision Tree",
 "Random Forest",
 "KNN",
 "SVM",
 "AdaBoost",
 "XGBoost"] ,

 "Accuracy Score" : [accuracy_score_lr,accuracy_score_dt,accuracy_score_rf,accuracy_score_knn,accuracy_score_svm,accuracy_score_ada,accuracy_score_xgb]
})

[142] Python

models

[143] Python

	Model	Accuracy Score
0	Logistic Regression	0.609756
1	Decision Tree	0.620427
2	Random Forest	0.653963
3	KNN	0.643293
4	SVM	0.675305
5	AdaBoost	0.612805
6	XGBoost	0.644817

File Edit Selection View Go Run Terminal Help ⌘ ⌘ ⌘ ⌘ ⌘ ⌘ ⌘ ⌘

File 2.ipynb X Water quality analysis(team).ipynb ●

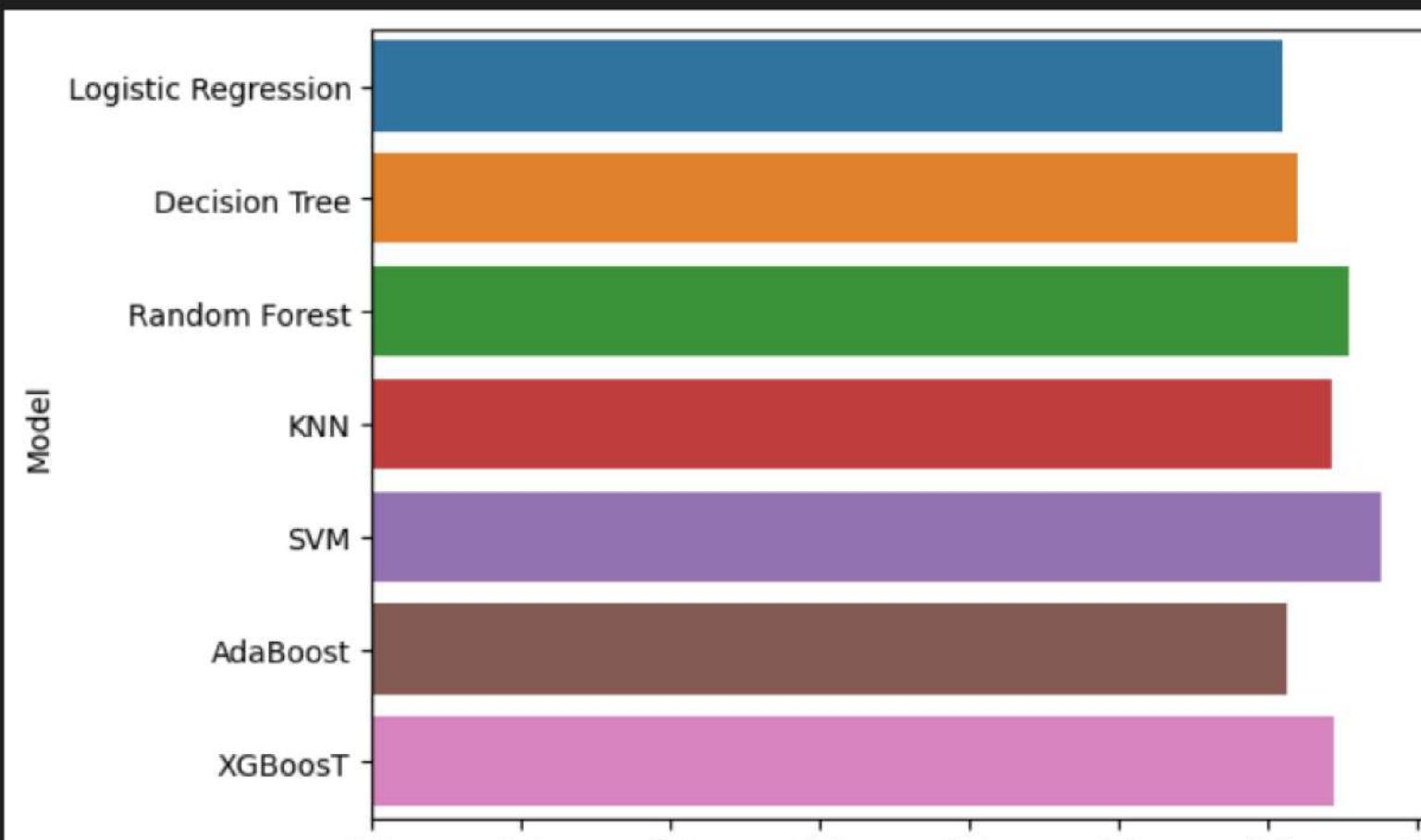
C: > Users > Roshan > Water quality analysis > File 2.ipynb > pwd

+ Code + Markdown | ▶ Run All | Clear All Outputs | Outline ... Select Kernel

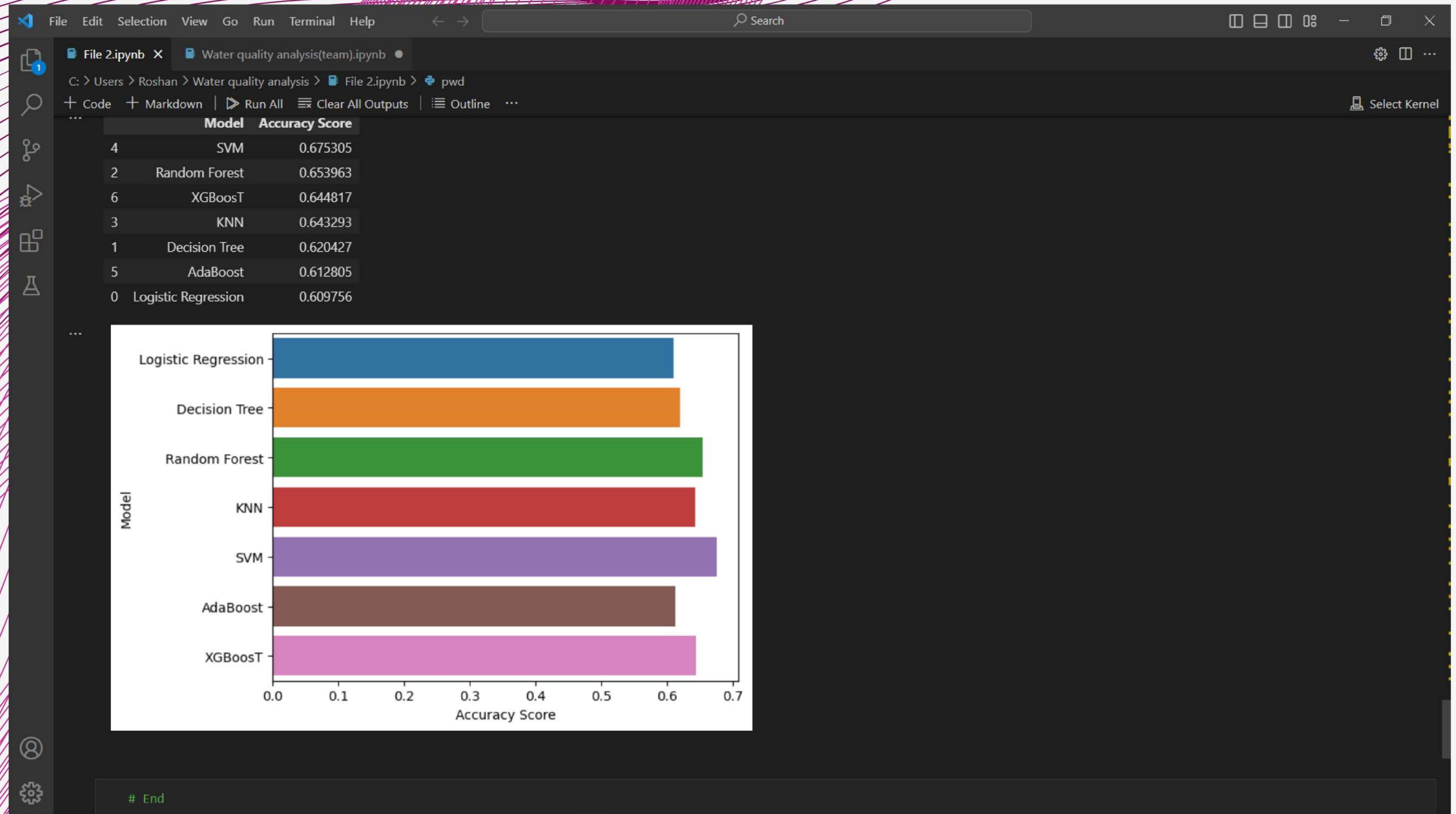
...
[145] Python

```
sns.barplot(x="Accuracy Score" ,y= "Model",data=models)
models.sort_values(by="Accuracy Score",ascending= False)
```

...
Model Accuracy Score
4 SVM 0.675305
2 Random Forest 0.653963
6 XGBoosT 0.644817
3 KNN 0.643293
1 Decision Tree 0.620427
5 AdaBoost 0.612805
0 Logistic Regression 0.609756

...


Model	Accuracy Score
Logistic Regression	0.61
Decision Tree	0.62
Random Forest	0.64
KNN	0.64
SVM	0.66
AdaBoost	0.61
XGBoosT	0.64





**All other documents
are submitted through
Github**

Team members:

Roshan.R

Ashwin.N

Rishwanth.S

Samual Rithick.R

Mariya Fernandez.F