

The background features a complex network of thin grey lines connecting various points, forming a web-like structure. Scattered throughout are numerous triangles of different sizes and orientations, some with solid black dots at their vertices. The overall aesthetic is technical and modern.

# **WATERFLOW**

**Using Machine Learning for Water Potability Classification**

---

Brice & Enam  
29/05/2024

# Sommaire

## Using Machine Learning for Water Potability Classification

1. **Introduction.**
2. **Data Exploration & Analyse.**
3. **Entraînement du Modèle.**
4. **Prédiction et Évaluation.**
5. **Configuration de MLflow.**
6. **Initialisation de l'Expérience MLflow.**
7. **Enregistrement du Modèle.**
8. **Exploration de MLflow UI.**
9. **Transition du Modèle.**
10. **Déploiement du Modèle FLASK .**

# Introduction & Environnement

**MLOps:** A set of practices to deploy and maintain machine learning models in production reliably and efficiently.

## Importance of MLOps

- **Streamlining Workflows:** Automates and manages the end-to-end machine learning lifecycle, improving productivity and model performance.

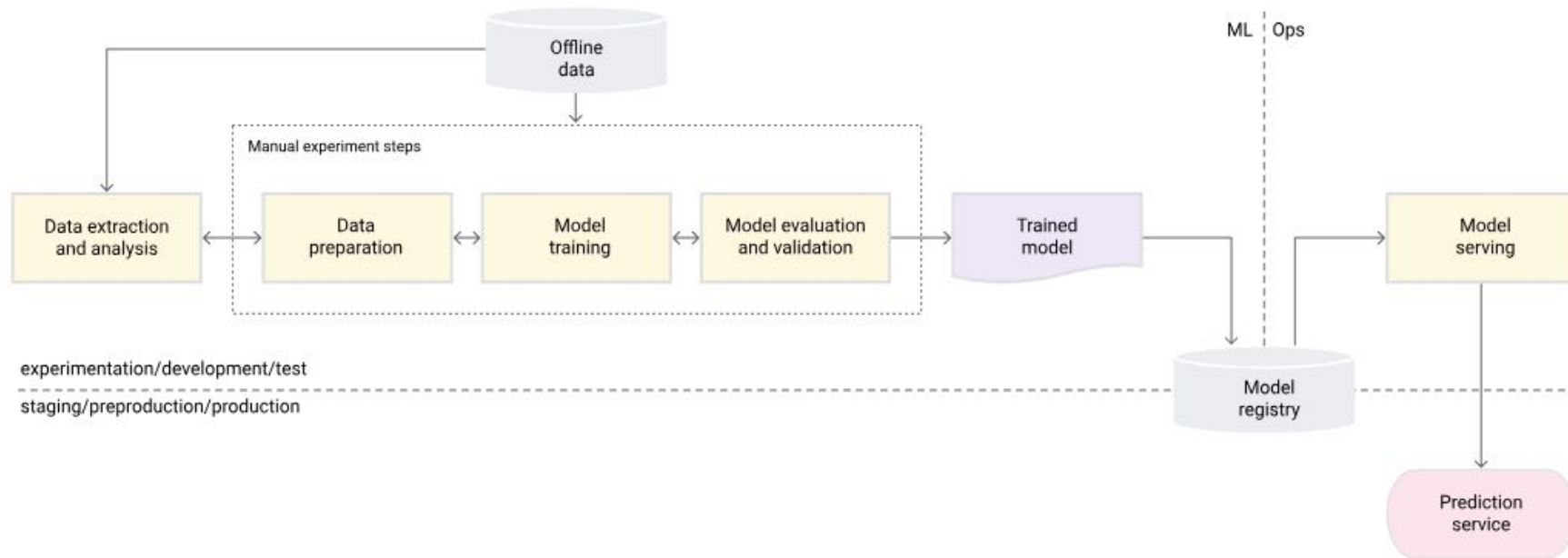
## Key Components

- **Version Control:** Tracks changes to data, code and models.
- **Automation:** Automates repetitive tasks such as model training and deployment.
- **Monitoring:** Continuously monitors model performance and data quality.
- **Testing:** Ensures model reliability through rigorous testing.

## Environnement



# ML OPS Niveau 0 : rappel



Managed the machine learning lifecycle using MLflow  
Components: Tracking, Projects, Models, Registry  
Benefits: Experiment tracking, reproducibility, model management

# Planning & Trello



**Trello** Espaces de travail Récent Favoris Modèles Créer

Parcourir

Espace de travail Trello Gratuit

WaterFlow Visible par l'espace de travail Tableau

Power-ups Automatisation Filtres BC EA Partager

Tableaux Membres Paramètres d'espace de travail

Tableur Calendrier

Back Log

Planning

Tâches

+ Ajouter une carte

To Do + Ajouter une carte

Doing + Ajouter une carte

To Validate + Ajouter une carte

Done

deploy on flask 29 mai EA

Statistiques 29 mai

create git 27 mai EA BC

perform eda 29 mai EA BC

try algorithms: 29 mai EA BC

perform test using pytest 29 mai EA BC

integrate with mlflow 29 mai EA

create trello 27 mai EA BC

+ Ajouter une carte

Maintenez l'ordre dans vos projets

Grâce à la structure et aux fonctionnalités personnalisables de Jira, gérez tous les projets et processus de vos équipes en toute simplicité.

Essayez-le gratuitement

Passez à Trello Premium

Votre écran est partagé par le biais de l'application meet.google.com. Arrêter le partage Masquer

- Used Trello for project management
- Organized tasks and milestones
- Monitored progress and collaborated effectively



# Data Processing & Analysis

---

# VALEURS MANQUANTES

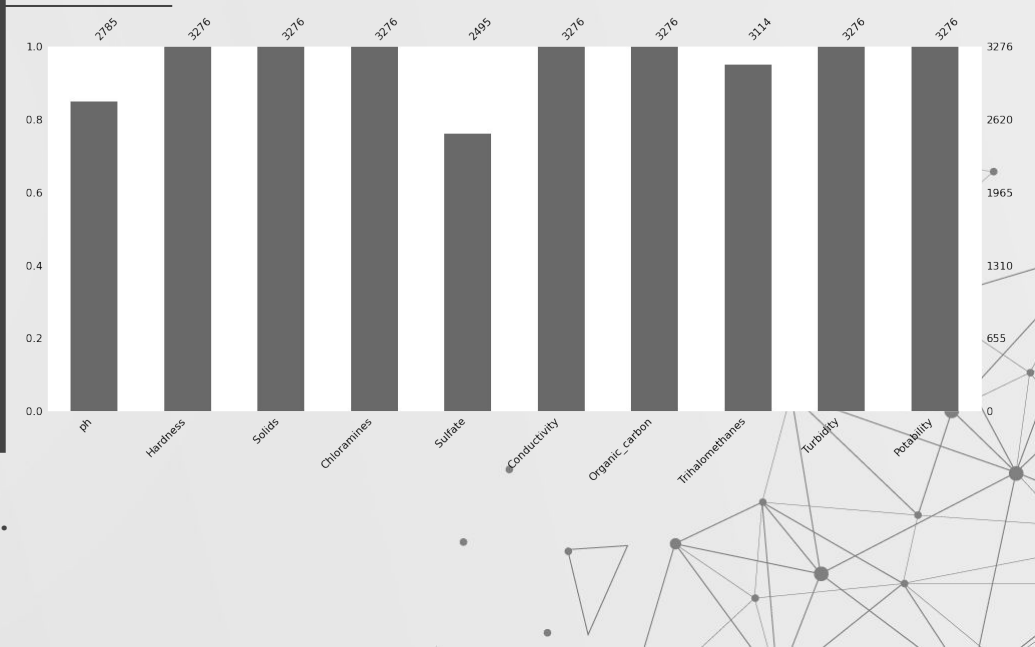
PH 14.99 Sulfate 23.84 TRIHALOMETHANES 4.95

DÉTECTION DES VALEURS MANQUANTES  
AIDE A LA PRISE DE DÉCISION SUR LES COLONNE.

|                 | Total | Manquants | %     |
|-----------------|-------|-----------|-------|
| ph              | 3276  | 491       | 14.99 |
| Hardness        | 3276  | 0         | 0.00  |
| Solids          | 3276  | 0         | 0.00  |
| Chloramines     | 3276  | 0         | 0.00  |
| Sulfate         | 3276  | 781       | 23.84 |
| Conductivity    | 3276  | 0         | 0.00  |
| Organic_carbon  | 3276  | 0         | 0.00  |
| Trihalomethanes | 3276  | 162       | 4.95  |
| Turbidity       | 3276  | 0         | 0.00  |
| Potability      | 3276  | 0         | 0.00  |

Stratégie choisie pour les valeurs manquantes : les moyennes.

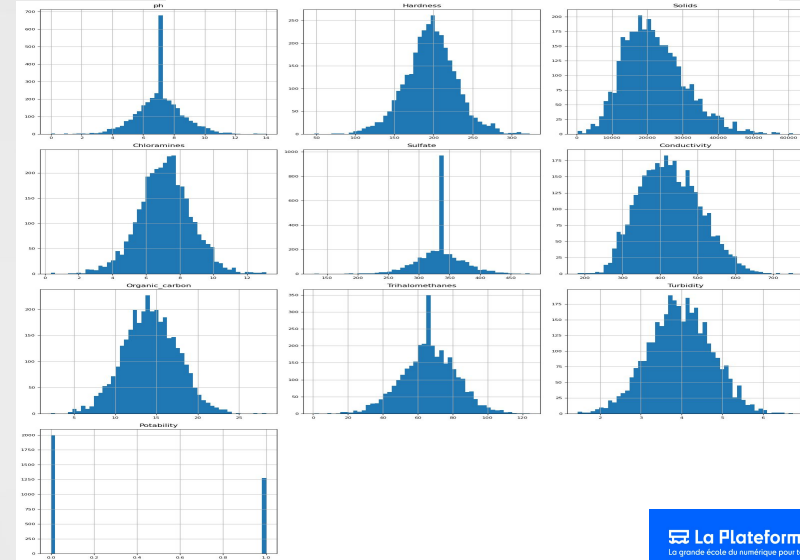
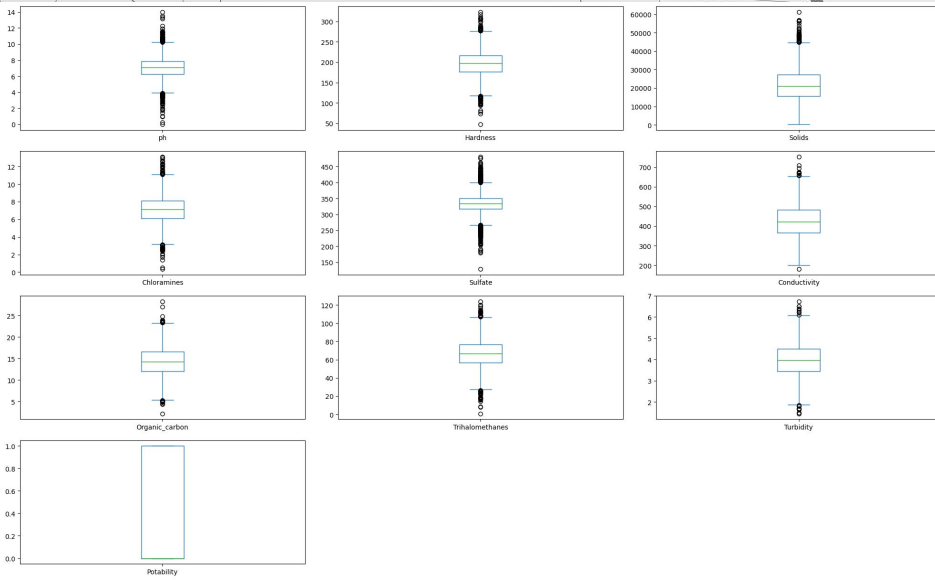
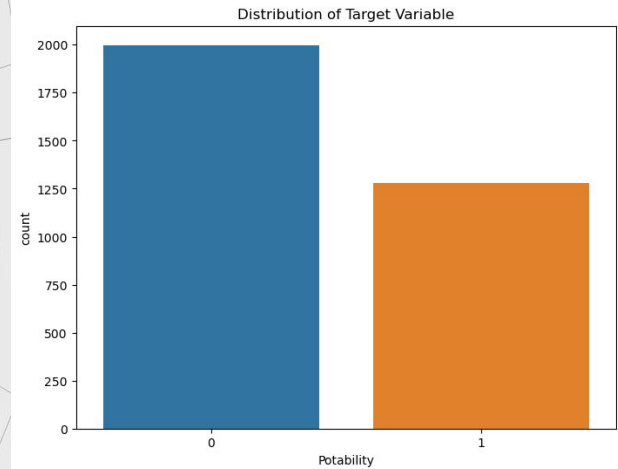
## HISTOGRAMME



**Data Preparation: Cleaned and preprocessed data**

**Data Exploration: Analyzed data for insights and patterns**

**Tools used: Pandas, NumPy, Scikit-learn**





# Data Version Control (DVC)



## What is DVC?

- **DVC:** A tool for versioning data and machine learning models, enabling reproducibility and collaboration in data science projects.

## Importance of DVC

- **Data Versioning:** Keeps track of changes to datasets and models.
- **Reproducibility:** Ensures experiments can be reliably reproduced.
- **Collaboration:** Facilitates collaboration among data scientists by managing data dependencies.

## Key Features

- **Data Management:** Efficiently manages large datasets.
- **Pipeline Management:** Automates and tracks machine learning pipelines.
- **Integration:** Integrates seamlessly with Git for version control

```
> data
> templates
.dvcignore
.gitignore
app.py
eda.ipynb
experiment.py
experiment_mlp.py
experiment_rf.py
experiment_xgb.py
features.csv
readme.md
targets.csv
test_experiment_mlp.py
test_experiment_xgb.py
test_mlops_rf.py
```

# Model Training

Split data into training and validation and test sets

Trained a binary classification model using MLP, Random Forest XBoost

Hyperparameter tuning with RandomizedSearchCV with more focus on MLP

experiment\_water\_quality\_mlp >

entertaining-cod-47

Model registered

Overview Model metrics System metrics Artifacts

No description

## Details

|                   |  |
|-------------------|--|
| Created at        | 2024-05-21 13:40:18  |
| Created by        | eakli  |
| Status            | 🟢 Finished   |
| Run ID            | 1cad89fafb6242a596f3ae49f5ef96c5   |
| Duration          | 12.2s  |
| Datasets used     | —  |
| Tags              | Add  |
| Source            | <code>./experiment_mlp.py</code> <code>f42a1a940c69d0f51e321869588f7886fbc1007c</code> |
| Logged models     | 🔗 tensorflow   |
| Registered models | 🔗 WaterQualityMLP v2   |

## Parameters (7)

| 🔍 Search parameters |                           |
|---------------------|---------------------------|
| Parameter           | Value                     |
| best_dropout_rate   | 0.3                       |
| best_learning_rate  | 0.001                     |
| data_url            | data/water_potability.csv |
| data_version        | v1                        |
| epochs              | 50                        |
| input_cols          | 9                         |
| input_rows          | 3276                      |

Best parameter after model tuning  
using RandomizedSearchCV



## Metrics (3)

| 🔍 Search metrics |                    |
|------------------|--------------------|
| Metric           | Value              |
| accuracy_test    | 0.6753048780487805 |
| precision_test   | 0.6165413533834586 |
| recall_test      | 0.3360655737704918 |

# Prediction & Evaluation

Made predictions on the validation set  
Evaluation metrics: Accuracy, Precision, Recall  
Confusion Matrix and Classification Report

experiment\_water\_quality\_mip >  
entertaining-cod-47

Overview Model metrics System metrics **Artifacts**

model

- data
- metadata
- MLmodel
- conda.yaml
- python\_env.yaml
- requirements.txt
- classification\_report.json
- confusion\_matrix.json
- features.csv
- scaler.pkl
- targets.csv

classification\_report.json 652B

Path: mlflow-artifacts/964585265176892967/1cad89afb6242a596f3ae49f5ef96c5/artifacts/classification\_report.json

```
{
  "0.0": {
    "precision": 0.69044595955831,
    "recall": 0.876213592238068,
    "f1-score": 0.772192513368884,
    "support": 412
  },
  "1.0": {
    "precision": 0.6165413533834586,
    "recall": 0.336865573784918,
    "f1-score": 0.4358132625994895,
    "support": 244
  },
  "accuracy": 0.6753848788487885,
  "macro avg": {
    "precision": 0.6534940596745208,
    "recall": 0.606109538017597,
    "f1-score": 0.603682878942258,
    "support": 656
  },
  "weighted avg": {
    "precision": 0.6628330783588173,
    "recall": 0.6753848788487885,
    "f1-score": 0.646782798518548,
    "support": 656
  }
}
```

experiment\_water\_quality\_mip >  
entertaining-cod-47

Overview Model metrics System metrics **Artifacts**

model

- data
- metadata
- MLmodel
- conda.yaml
- python\_env.yaml
- requirements.txt
- classification\_report.json
- confusion\_matrix.json
- features.csv
- scaler.pkl
- targets.csv

confusion\_matrix.json 107B

Path: mlflow-artifacts/964585265176892967/1cad89afb6242a596f3ae49f5ef96c5/artifacts/confusion\_matrix.json

```
{
  "confusion_matrix": [
    [
      361,
      51
    ],
    [
      162,
      82
    ]
  ]
}
```

# MLflow Configuration

Configured MLflow server on port 5000

Initialized MLflow experiment:

"experiment\_water\_quality\_mlp"

Logged model and metadata during training

```
def main():
    mlflow.set_tracking_uri("http://localhost:5000")
    experiment_name = "experiment_water_quality_mlp"
    experiment_id = mlflow.create_experiment(experiment_name)
    client = mlflow.tracking.MlflowClient()
    experiment = client.get_experiment(experiment_id)

    data_path = 'data/water_potability.csv'
    repo = r'C:\Users\ekali\Downloads\task\ecole\mlops-mlflow'
    version = 'v1'

    data = load_data(data_path, repo, version)
    X_train, X_val, X_test, y_train, y_val, y_test, feature_names, scaler = preprocess_data(data)
    best_params = tune_hyperparameters(X_train, y_train)
    final_model = train_final_model(X_train, y_train, X_val, y_val, best_params)
    evaluate_and_log_model(final_model, X_test, y_test, feature_names, experiment_id, data_path, version, y_train, y_val, best_params, scaler)

if __name__ == "__main__":
    main()
```

# Model Transition

Explored different model versions in MLflow

Transitioned the best model which is MLP to the Model Registry

Ensured model meets quality criteria before transition

mlflow 2.12.2 Experiments Models

Experiments

Search Experiments

- ☐ Default
- ☒ experiment\_water\_quality\_mlp
- ☐ experiment\_water\_quality\_xgb
- ☐ experiment\_water\_quality\_rf
- ☐ experiment\_water\_quality

experiment\_water\_quality\_mlp Provide Feedback Add Description

metrics.rmse < 1 and params.model = "tree" Time created State: Active Datasets Sort: Created Columns Group by

Table Chart Evaluation Experimental

|                          | Run Name            | Created    | Dataset | Duration | Source       | Models          |
|--------------------------|---------------------|------------|---------|----------|--------------|-----------------|
| <input type="checkbox"/> | entertaining-cod-47 | 7 days ago | -       | 12.2s    | .\experim... | WaterQuali.../2 |

# Model Deployment

Deployed the registered model using Flask  
Created an API for real-time predictions  
Integrated with a web interface for user input



Flask

```
app = Flask(__name__)

# Set the MLflow tracking URI
mlflow.set_tracking_uri("http://localhost:5000")

# Load model from the MLflow Model Registry
model_name = "WaterQualityMLP"
model_version = 2
model = mlflow.keras.load_model(f"models://{model_name}/{model_version}")

# Load scaler used during training
scaler = joblib.load("artifacts/scaler.pkl")

@app.route('/')
def home():
    return render_template('index.html')

@app.route('/predict', methods=['POST'])
def predict():
    # Get form data
    data = request.form.to_dict()

    # Convert form data to DataFrame
    input_df = pd.DataFrame([data])

    # Ensure the input data contains all necessary features
    required_features = [
        "ph", "Hardness", "Solids", "Chloramines", "Sulfate",
        "Conductivity", "Organic_carbon", "Trihalomethanes", "Turbidity"
    ]
```





# Flask Application Demo

## Water Quality Prediction

pH

Hardness

Solids

Chloramines

Sulfate

Conductivity

Organic Carbon

Trihalomethanes

Turbidity

Prediction: 1



- Live demonstration of the web application
- How to use the web interface to input data
- Real-time prediction results displayed on the UI

# Using Pytest for Unit Testing

- Ensured code reliability and correctness with **pytest**
- Created unit tests for data processing and model functions
- Integrated tests into the CI/CD pipeline
- Benefits: Early bug detection, improved code quality

```
cache: .pytest_cache
rootdir: C:\Users\ekali\Downloads\task\ecole\mlops-mlflow
plugins: hydra-core-1.3.2, mock-3.14.0
collected 2 items

test_experiment_mlp.py::test_load_data PASSED [ 50%]
test_experiment_mlp.py::test_main PASSED [100%]

===== warnings summary =====
..\..\..\..\anaconda3\envs\env_mlops\Lib\site-packages\mlflow\utils\requirements_utils.py:20
C:\Users\ekali\anaconda3\envs\env_mlops\Lib\site-packages\mlfl
```

```
import pytest
from experiment_mlp import load_data, preprocess_data, create_model, train_final_model

def test_load_data():
    data = load_data('data/water_potability.csv', 'repo', 'v1')
    assert not data.empty

def test_preprocess_data():
    data = load_data('data/water_potability.csv', 'repo', 'v1')
    X_train, X_val, X_test, y_train, y_val, y_test, feature_names, scaler = preprocess_data(data)
    assert X_train.shape[0] > 0
    assert y_train.shape[0] > 0

def test_create_model():
    model = create_model(input_shape=9, learning_rate=0.001, dropout_rate=0.5)
    assert model is not None

def test_train_final_model():
    data = load_data('data/water_potability.csv', 'repo', 'v1')
    X_train, X_val, X_test, y_train, y_val, y_test, feature_names, scaler = preprocess_data(data)
    best_params = {'learning_rate': 0.001, 'dropout_rate': 0.5, 'epochs': 50}
    model = train_final_model(X_train, y_train, X_val, y_val, best_params)
    assert model is not None
```



# Challenges and Solutions

- Handling missing data - mean or median
- Hyperparameter tuning -
- Model deployment using MLflow
- Integrating model with a web application
- Some tests failed

# Conclusion

- **Random Forest est le efficace.**

# References

[MLflow: A Tool for Managing the Machine Learning Lifecycle — MLflow 2.13.0 documentation](#)

[Home | Data Version Control · DVC](#)

[Home - MLOps Community](#)



# THANK YOU