WATERFLOW

Using Machine Learning for Water Potability Classification

Brice & Enam
29/05/2024

La Plateforme
La grande école du numérique pour tous

Sommaire

Using Machine Learning for Water Potability Classification

La Plateforme
La grande école du numérique pour tous

Introduction & Environnement

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

Importance of MLOps

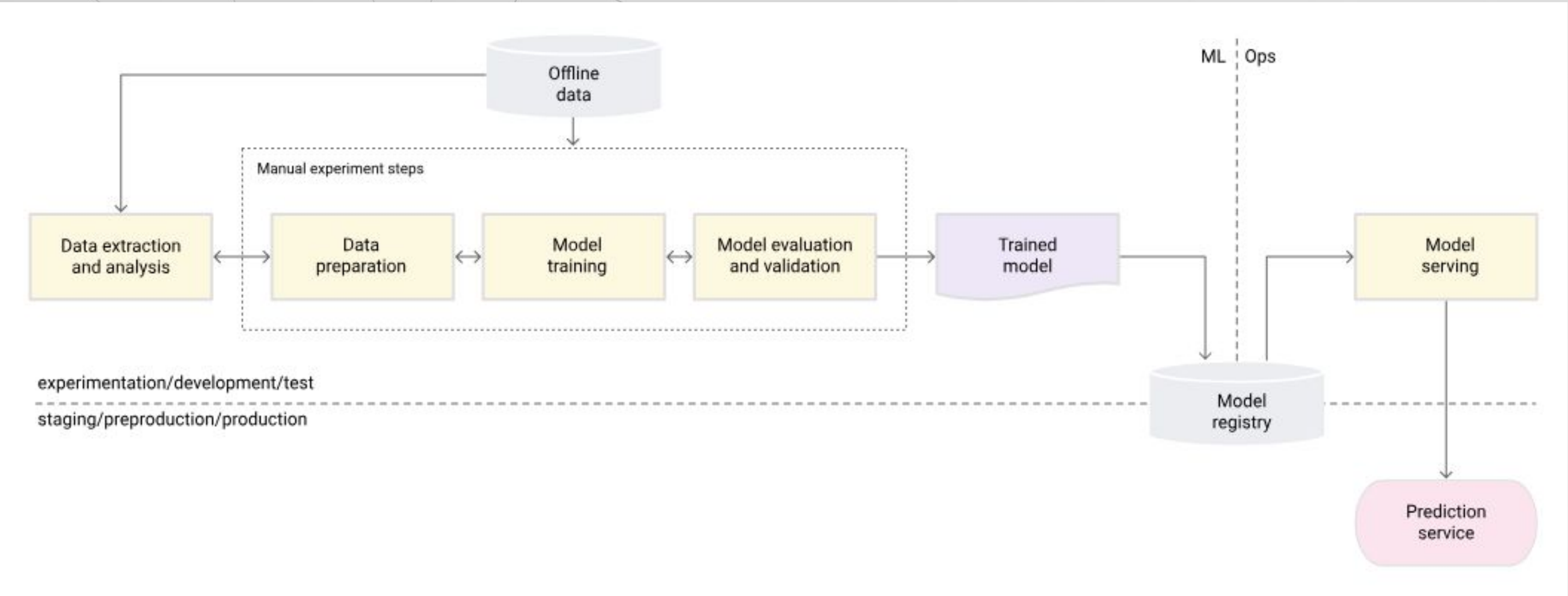
- *Streamlining Workflows*: Automates and manages the end-to-end machine learning lifecycle, improving productivity and model performance.

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

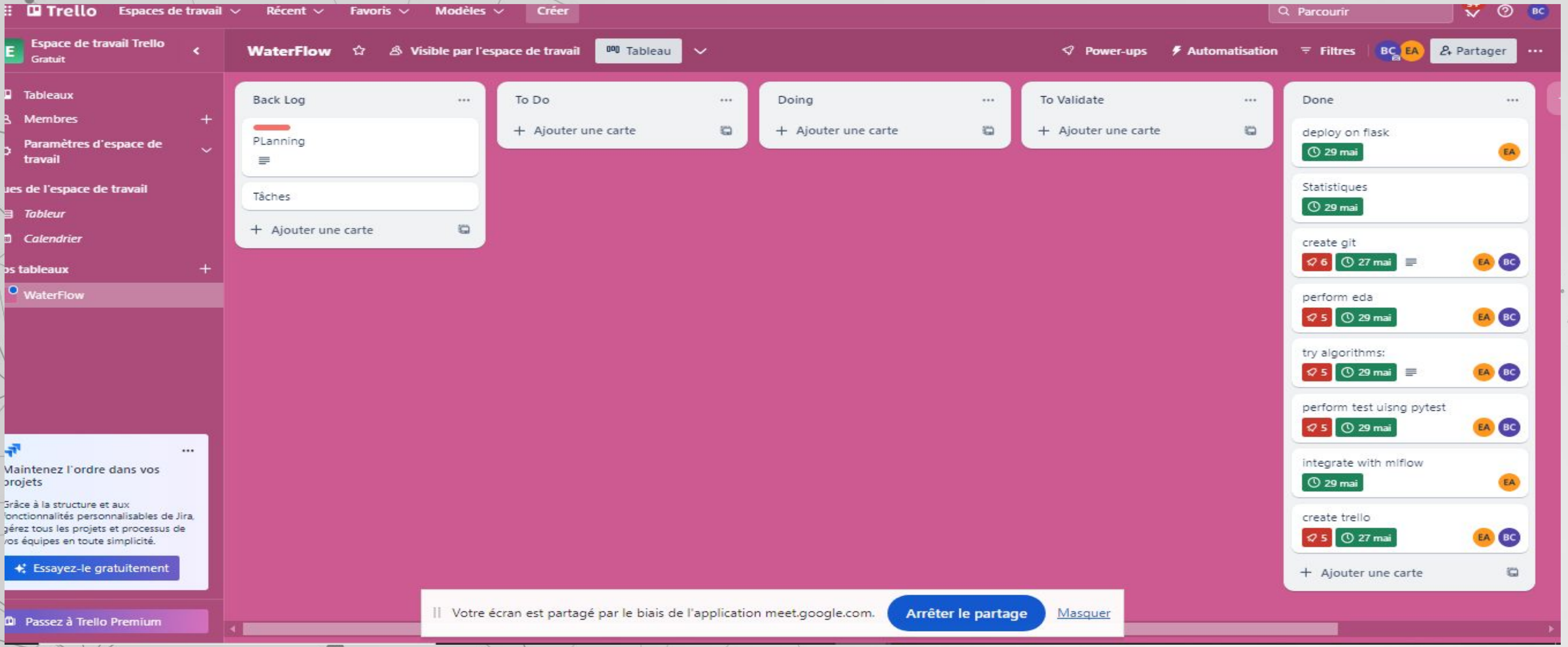
ML OPS Niveau 0 : rappel



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

Planning & Trello

Trello

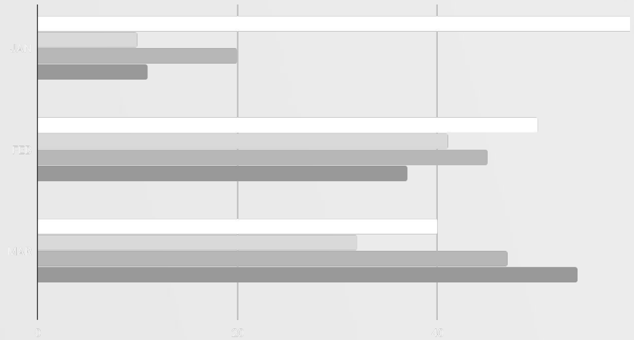


- Used Trello for project management
- Organized tasks and milestones
- Monitored progress and collaborated effectively

Data Processing & Analysis

La Plateforme
La grande école du numérique pour tous

VALEURS MANQUANTES

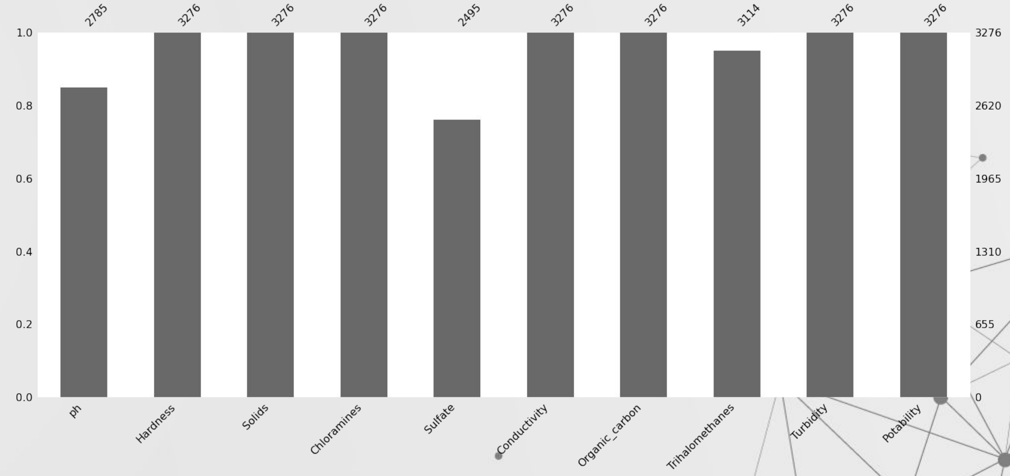


PH 14.99 SULFATE 23.84 TRIHALOMETHANES 4.95

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

HISTOGRAMME

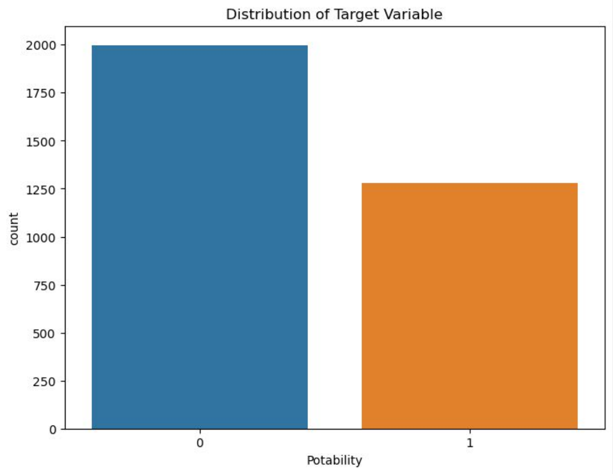


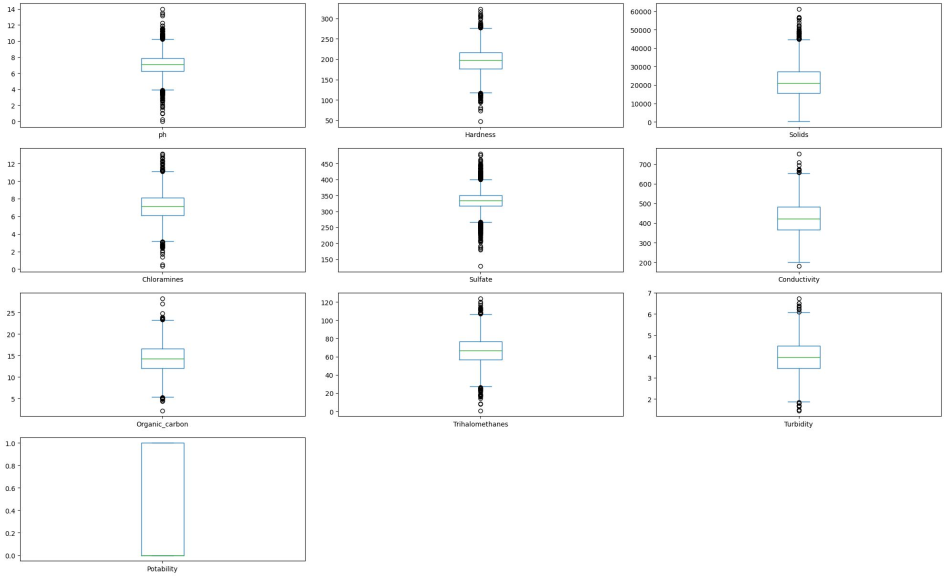
Stratégie choisie pour les valeurs manquantes : les moyennes.

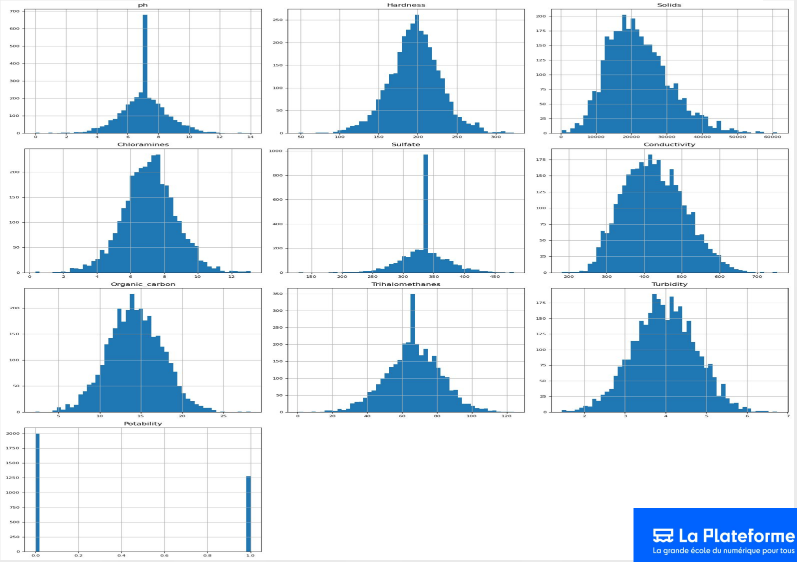
Data Preparation: Cleaned and preprocessed data

Data Exploration: Analyzed data for insights and patterns

Tools used: Pandas, NumPy, Scikit-learn







La Plateforme
La grande école du numérique pour tous

Data Version Control (DVC)

What is DVC?

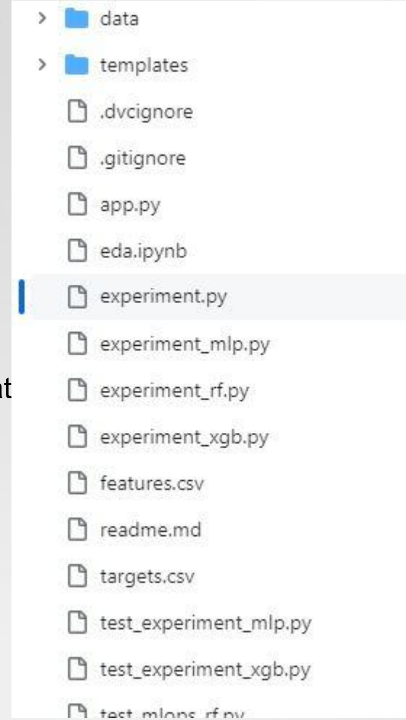
- **DVC**: A tool for versioning data and machine learning models, enabling reproducibility and collaborat data science projects.

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



La Plateforme
La grande école du numérique pour tous

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 registere

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	.\experiment_mlp.py f42a1a940c69d0f51e321869588f7886fbc1007c
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

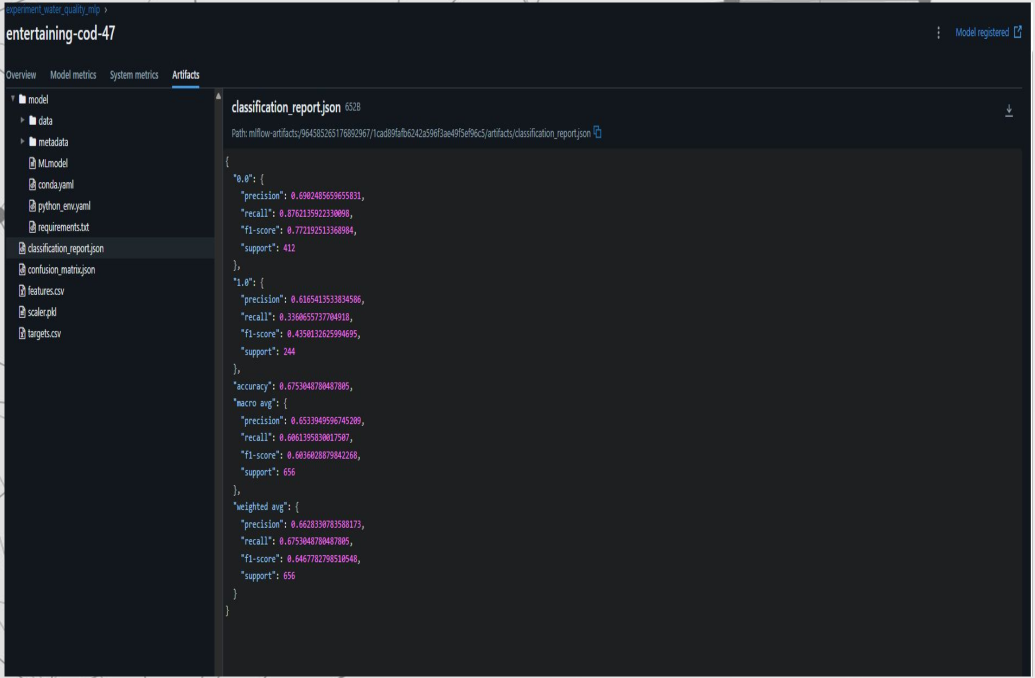
Metrics (3)

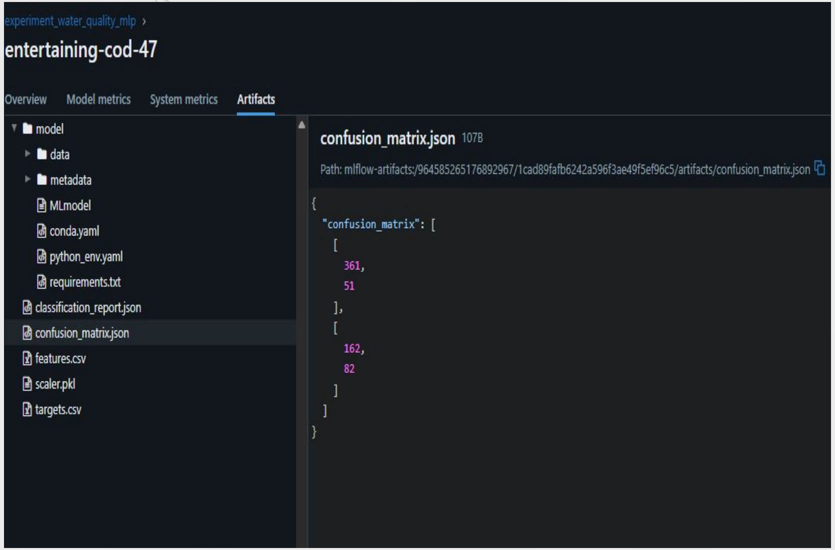
Search metrics

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





MLflow Configuration

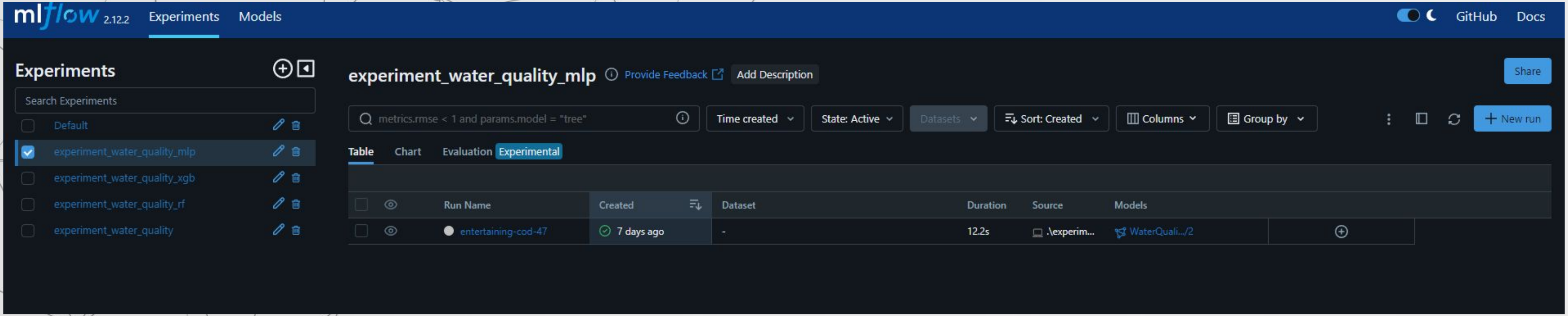
Configured MLflow server on port 5000
Initialized MLflow experiment: "experiment_water_quality_mlp"
Logged model and metadata during training

```python
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\eakli\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()
```

La Plateforme
La grande école du numérique pour tous

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



La Plateforme
La grande école du numérique pour tous

Model Deployment

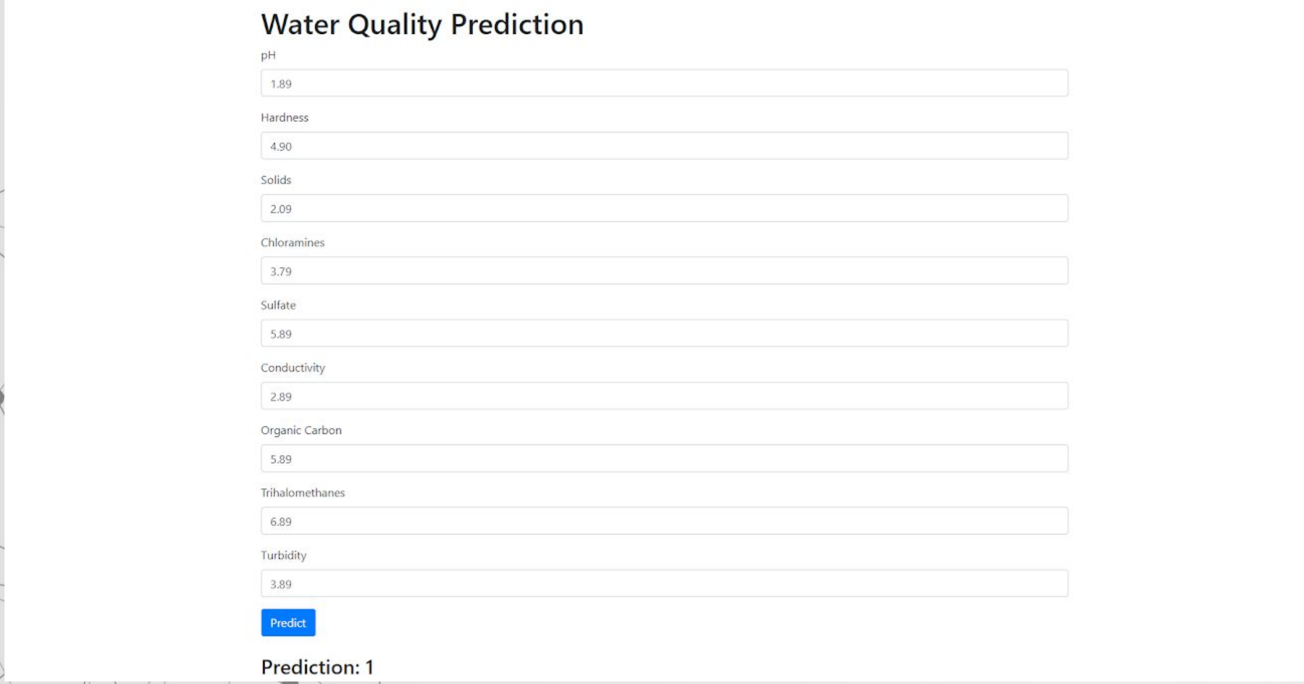
Deployed the registered model using Flask
Created an API for real-time predictions
Integrated with a web interface for user input

Flask

```python
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"
    ]
```

La Plateforme
La grande école du numérique pour tous

Flask Application Demo



- 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

```
rootdir: C:\Users\eakli\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\ut
ils\requirements_utils.py:20
  C:\Users\eakli\anaconda3\envs\env_mlops\Lib\site-packages\mlfl
```

```python
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_dat
    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_dat
    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

La Plateforme
La grande école du numérique pour tous

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

La Plateforme
La grande école du numérique pour tous

THANK YOU

La Plateforme
La grande école du numérique pour tous