Building Machine Learning Microservices & MLOps using UnionML

Gaurav Pandey, Shivay Lamba

yednapg@gmail.com, shivaylamba@gmail.com

About Union ML

What is UnionML?

UnionML is an open source Python framework built on top of Flyte, unifying the complex ecosystem of ML tools into a single interface.

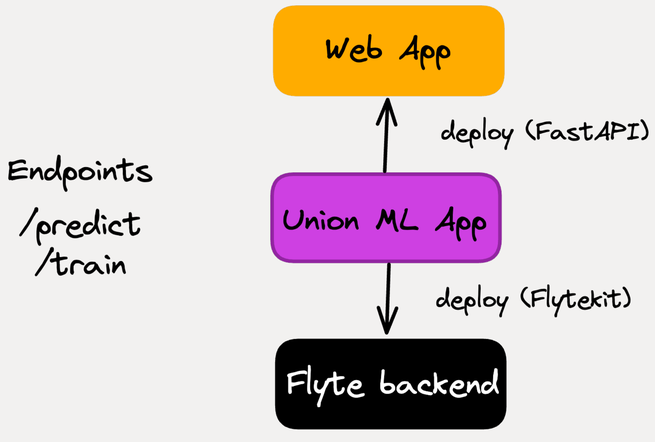
Problem UnionML addresses

- Complexity in deploying machine learning models to production
- Unifying the constantly evolving ecosystem of machine learning and data tools into a single interface for expressing microservices as Python functions

Features of UnionML

- A Unified Interface For Your ML Team
- Prototype Locally
- Scale Up According to Your Needs
- Serve Anywhere Seamlessly

Flow Chart to illustrate use of Union ML



Code Structure for UnionML with MNIST

Setup and importing libraries

```
from typing import List, Union

import pandas as pd
from sklearn.datasets import fetch_openml
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score
```

Caching Data

```
from unionml import Dataset, Model

dataset = Dataset(name="mnist_dataset", test_size=0.2, shuffle=True, targets=["class"])
model = Model(name="mnist_classifier", dataset=dataset)
```

Define Core UnionML Functions

```
from pathlib import Path
from joblib import Memory

memory = Memory(Path.home() / "tmp")
fetch_openml_cached = memory.cache(fetch_openml)

@dataset.reader(cache=True, cache_version="1")
def reader() -> pd.DataFrame:
    dataset = fetch_openml_cached("mnist_784",version=1,cache=True,as_frame=True,)
    return dataset.frame.sample(1000, random_state=42)

@model.init
def init(hyperparameters: dict) -> Pipeline:
    estimator = Pipeline(
        [("scaler", StandardScaler()),("classifier", LogisticRegression()),]
    )
    return estimator.set_params(**hyperparameters)

@model.trainer(cache=True, cache_version="1")
def trainer(
    estimator: Pipeline,
    features: pd.DataFrame,
    target: pd.DataFrame,
) -> Pipeline:
    return estimator.fit(features, target.squeeze())

@model.predictor
def predictor(
    estimator: Pipeline,
    features: pd.DataFrame,
) -> List[float]:
    return [float(x) for x in estimator.predict(features)]

@model.evaluator
def evaluator(
    estimator: Pipeline,
    features: pd.DataFrame,
    target: pd.DataFrame,
) -> float:
    return float(accuracy_score(target.squeeze(), estimator.predict(features)))
```

Training a Model Locally

```
estimator, metrics = model.train(
    hyperparameters={"classifier__penalty": "l2",
        "classifier__C": 0.1,
        "classifier__max_iter": 1000,
    }
)
print(estimator, metrics, sep="\n")
```

UnionML Model Functions

`init()`

The init argument is used to take a class that is initialized to produce a model object

`trainer()`

The trainer function should contain all the logic for training a model from scratch or a previously saved model checkpoint.

`predictor()`

The predictor function uses an estimator object and features dataframe to produce a list of values that represent the predicted result

`evaluator()`

The evaluator function evaluate the estimator object based on given features and a target value.

Ecosystem of Integrations