

Ход выполнения лабораторной работы

В рамках данной лабораторной работы предлагается построить два пайплайна:

- Пайплайн, который обучает любой классификатор из `sklearn` по заданному набору параметров.
- Пайплайн, который выбирает лучшую модель из обученных и производит её хостинг.

Для построения такого пайплайна воспользуемся следующими инструментами:

- Apache Airflow
- MLflow

Первый pipeline

Построенный пайплайн будет выполнять следующие действия поочередно:

1. Производить мониторинг целевой папки на предмет появления новых конфигурационных файлов классификаторов (в форматах `.json` или `.yaml`).
2. Обучать классификатор в соответствии с полученными параметрами.
 - Производить логгирование параметров модели в **MLflow**.
 - Производить логгирование процесса обучения **MLflow**.
 - Производить тестирование модели и сохранять его результаты в **MLflow**.
3. Сохранять обученный классификатор в model registry **MLflow**.

Реализация первого пайплайна представлена на рисунке 1. В качестве `fs_conn_id` использовалось соединение созданное на Airflow по ссылке

<http://localhost:8080/connection/list/>. Для запуска питон файлов использовался BashOperator. В prepare_data.py происходит загрузка и создание данных для обучения, валидации и тестирования, а также их сохранение для дальнейшего использования. FileSensor сканирует определённую папку на появление конфигурационного файла. В train_model.py загружаются ранее полученные тренировочный и валидационный наборы данных. После для каждый полученный классификатор обучается на тренировочном датасете в соответствии с полученными параметрами. После обучения происходит все требуемое логгирование и модель сохраняется.

```
from datetime import datetime
from airflow import DAG
from airflow.sensors.filesystem import FileSensor
from airflow.operators.bash_operator import BashOperator
import os

os.environ["AWS_ACCESS_KEY_ID"] = "minio"
os.environ["AWS_SECRET_ACCESS_KEY"] = "minio123"
os.environ["MLFLOW_S3_ENDPOINT_URL"] = "http://minio:9000"

default_args = {
    'owner': 'airflow',
    'start_date': datetime(2023, 1, 1),
    'retries': 1,
}

dag = DAG(
    'airflow_lab3_first_pipeline',
    default_args=default_args,
    description='DAG for training sk-learn classifier according to configuration files',
    schedule_interval=None,
)

prepare_data = BashOperator(
    task_id="prepare_data",
    bash_command="python /opt/airflow/data/lab3/prepare_data.py",
    dag=dag
)

wait_for_conf_file = FileSensor(
    task_id='wait_for_conf_file',
    poke_interval=10,
    filepath='/opt/airflow/data/lab3/conf.json',
    fs_conn_id='FileSensor',
    dag=dag,
)

train_model = BashOperator(
    task_id="train_model",
    bash_command="python /opt/airflow/data/lab3/train_model.py",
    dag=dag
)

prepare_data >> wait_for_conf_file >> train_model
```

Рисунок 1 – DAG файл для первого pipeline

После запуска пайплайна получаем следующий результат:



Рисунок 2 – результат 2х запусков пайплайна

Код DAG представлен в файле – `airflow-lab3-first-pipeline.py`.

ВТОРОЙ PIPELINE

Построенный пайплайн будет выполнять следующие действия поочередно:

1. В соответствии с таймером производит валидацию новых моделей из model registry.
2. Модель с лучшим показателем метрики переводится на stage: Production

В итоге получился следующий DAG:

```

from datetime import datetime
from airflow import DAG
from airflow.operators.bash_operator import BashOperator
import os

os.environ["AWS_ACCESS_KEY_ID"] = "minio"
os.environ["AWS_SECRET_ACCESS_KEY"] = "minio123"
os.environ["MLFLOW_S3_ENDPOINT_URL"] = "http://minio:9000"

default_args = {
    'owner': 'airflow',
    'start_date': datetime(2023, 11, 17),
    'retries': 1,
}

dag = DAG(
    'airflow_lab3_second_pipeline',
    default_args=default_args,
    description='DAG for validate and host sk-learn model',
    schedule_interval='@weekly',
)

validate_model = BashOperator(
    task_id="validate_model",
    bash_command="python /opt/airflow/data/lab3/validate_model.py",
    dag=dag
)

validate_model

```

Рисунок 3 – DAG для второго пайплайна

После запуска этого пайплайна появляется новый запуск. И модель с наилучшим результатом переходит в стадию Production.

Default [Provide Feedback](#)

Experiment ID: 0 Artifact Location: s3://mlflow/0

> Description [Edit](#)

Q metrics.rmse < 1 and params.model = "tree"

Table	Chart	Evaluation	Experimental
🔍	Run Name		
🔍	●	validate model	
🔍	●	MLPClassifier	
🔍	●	KNeighborsClassifier	
🔍	●	SVC	
🔍	●	RandomForestClassifier	
🔍	●	SGDClassifier	

Рисунок 4 – новый запуск

Registered Models

Filter registered models by name or tags


Name 	Latest version	Staging	Production	Created by	Last modified	Tags
KNeighborsClassifier	Version 3	—	Version 3		2023-11-18 18:16:02	—
MLPClassifier	Version 3	—	—		2023-11-18 18:15:16	—
RandomForestClassifier	Version 3	—	—		2023-11-18 18:14:57	—
SGDClassifier	Version 3	—	—		2023-11-18 18:14:50	—
SVC	Version 3	—	—		2023-11-18 18:15:03	—

Рисунок 5 – Результат работы 2го пайплайна