

MLflow

Introduction to Experiment Tracking and Model Management

MLflow is an open-source platform to manage the ML lifecycle, including experimentation, reproducibility, deployment, and a central model registry. MLflow currently offers four components:

- **MLflow Tracking**-Record and query experiments: code, data, config, and results
- **MLflow projects**-Package data science code in a format to reproduce runs on any platform
- **MLflow models**- Deploy Machine learning models in diverse serving environments
- **Model Registry**- Store, annotate, discover, and manage models in a central repository

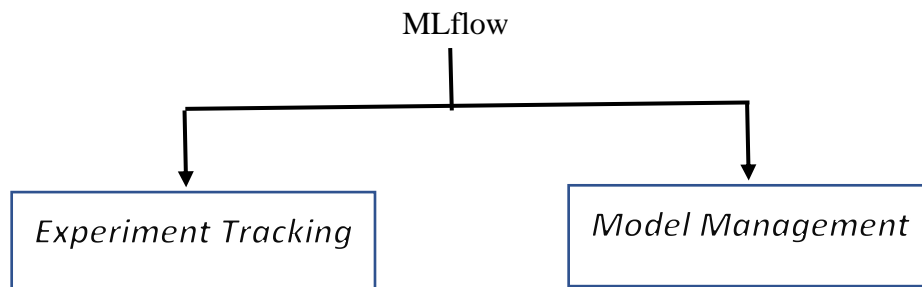
MLflow: Tracking of Experiments, logging or recording all the Experiments.

Login is important for Organization of Production pipeline properly.

MLflow interface is helping to track for-

- What kind of Algorithm
- What kind of Hyper parameters
- What kind of scores we get for the model.

For the production we choose best one based on the above data.



Terminologies:

1. Experiment -Each trial of some Experiment
2. Run
3. Metadata -All Information related to an Experiment Run (i.e. Tags, Parameters, Data, Train-Test size, Algorithms, Metrics)
4. Artifacts -Output files associated with experiment runs (i.e. Pickle files)

Why Track?

Organization is much easier

Optimization is much more meaningful

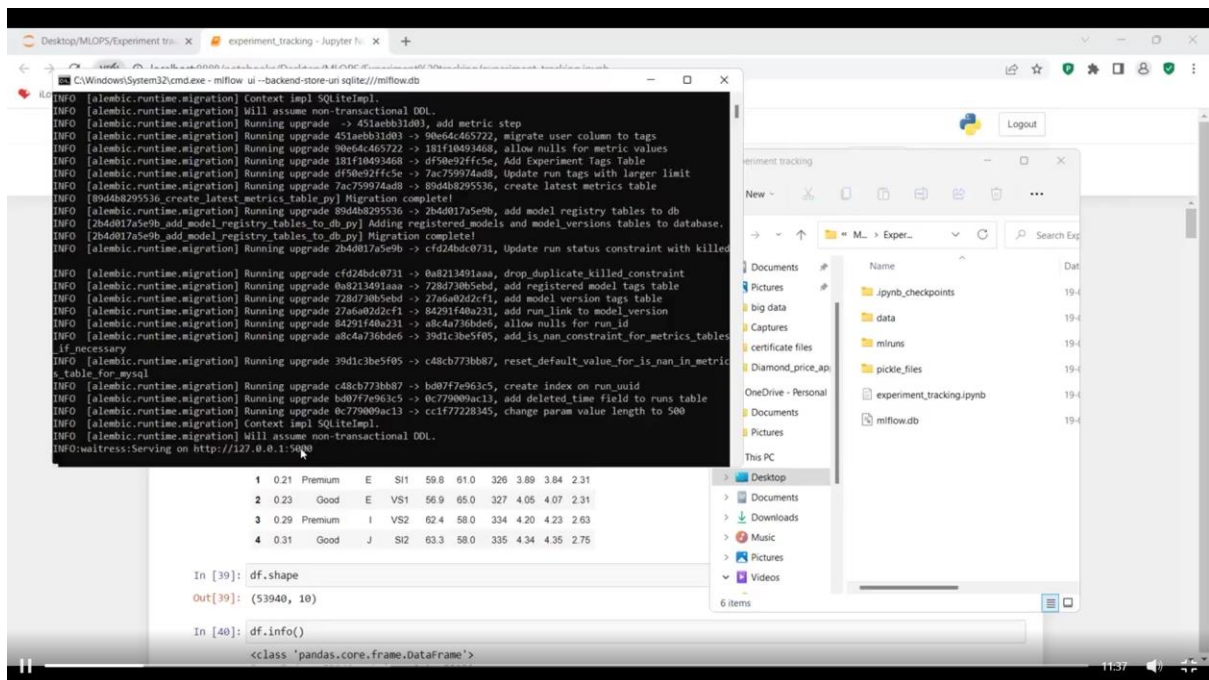
Reproducibility of Experiment run

Run below mentioned commands to install mlflow on your system:

```
pip install mlflow
```

```
mlflow ui --backend-store-uri sqlite:///mlflow.db
```

After executing above command in cmd we get user interface <http://127.0.0.1:5000>



Step 1 - Import MLFlow

```
import mlflow
```

Step 2 - Set the tracker and experiment

```
mlflow.set_tracking_uri(DATABASE_URI)
```

```
mlflow.set_experiment("EXPERIMENT_NAME")
```

Step 3 - Start a experiment run

```
with mlflow.start_run():
```

Step 4 - Logging the metadata

```
mlflow.set_tag(KEY, VALUE)
```

```
mlflow.log_param(KEY, VALUE) mlflow.log_metric(KEY, VALUE)
```

Step 5 - Logging the model and other files (2 ways)

Way 1 - `mlflow.<FRAMEWORK>.log_model(MODEL_OBJECT, artifact_path="PATH")`

Way 2 - `mlflow.log_artifact(LOCAL_PATH, artifact_path="PATH")`

Running the Experiment

```
import mlflow

mlflow.set_tracking_uri("sqlite:///mlflow.db")
mlflow.set_experiment("Diamond Price Prediction")

2022/09/19 13:28:55 INFO mlflow.tracking.fluent: Experiment with name 'Diamond Price Prediction' does not exist
<Experiment: artifact_location='./mlruns/1', experiment_id='1', lifecycle_stage='active', name='Diamond Price Prediction'>

from sklearn.neighbors import KNeighborsRegressor
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.svm import SVR
from sklearn.ensemble import GradientBoostingRegressor

from sklearn import metrics

from pickle import dump
dump(enc, open('pickle_files/Ordinal_Encoder.pkl', 'wb'))
dump(scaler, open('pickle_files/Standard_Scaler.pkl', 'wb'))
```

Experiment 1 - Training KNN Regressor

```
with mlflow.start_run():
    mlflow.set_tag("dev", "Veena Grace")
    mlflow.set_tag("algo", "KNN")
    # Log the data for each run using log_param, log_metric, log_model
    mlflow.log_param("data-path", "data/diamonds.csv")
    k = 3
    mlflow.log_param("n_neighbors", k)
    knn_regressor = KNeighborsRegressor(n_neighbors=k)
    knn_regressor.fit(X_train_rescaled, y_train)
    y_test_pred = knn_regressor.predict(X_test_rescaled)
    MAE = metrics.mean_absolute_error(y_test, y_test_pred)
    mlflow.log_metric("Mean Absolute error", MAE)
    mlflow.sklearn.log_model(knn_regressor, artifact_path="models")
    mlflow.log_artifact("pickle_files/Standard_Scaler.pkl")
    mlflow.log_artifact("pickle_files/Ordinal_Encoder.pkl")
```

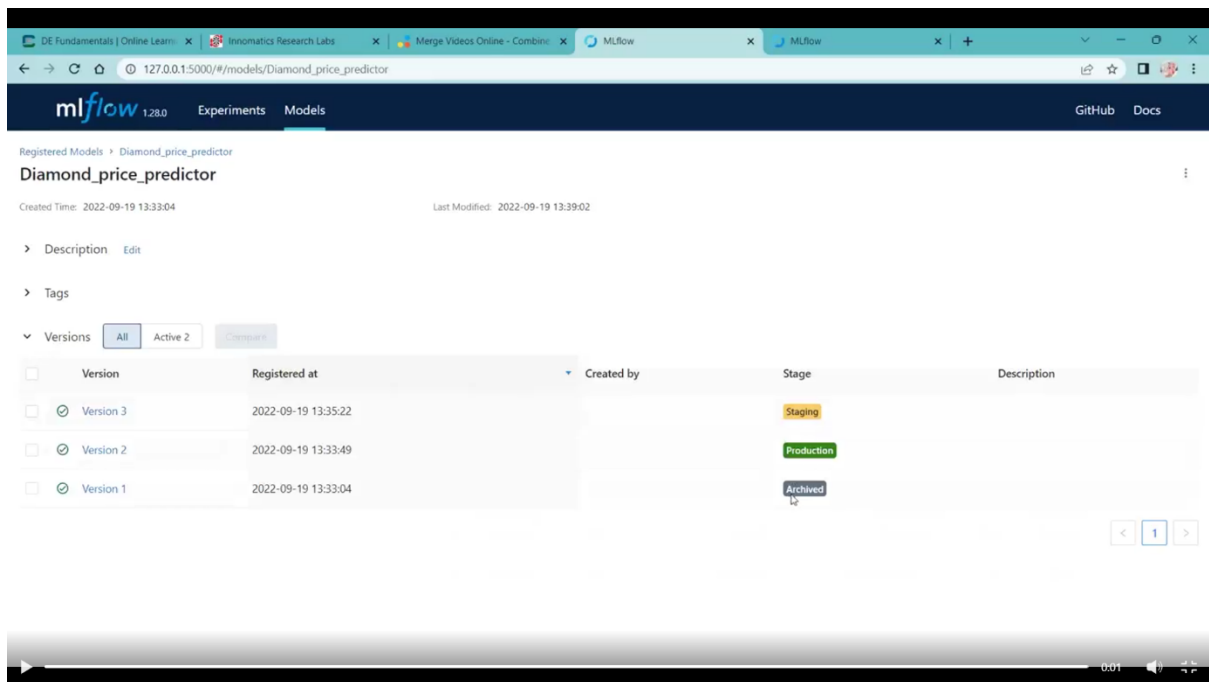
After Executing the above code and open the URL to get the-

Mlflow Interface For Tracking Experiments

The screenshot shows the MLflow web interface for tracking experiments. The experiment name is "Diamond Price Prediction". The interface displays a table of 6 runs with columns for Start Time, Duration, Run Name, User, Source, Version, Models, Metrics, and Parameters. The selected run (Diamond_pr_1) shows a Mean Absolute Error of 287.7.

	Start Time	Duration	Run Name	User	Source	Version	Models	Metrics	Parameters
<input type="checkbox"/>	7 minutes ago	-	-	Jessy	C:\Users\...	-	-	-	data-path
<input checked="" type="checkbox"/>	8 minutes ago	33.3s	-	Jessy	C:\Users\...	-	Diamond_pr_1	287.7	-
<input type="checkbox"/>	9 minutes ago	29.3s	-	Jessy	C:\Users\...	-	Diamond_pr_3	551	-
<input type="checkbox"/>	9 minutes ago	14.2s	-	Jessy	C:\Users\...	-	Diamond_pr_2	585.4	-
<input type="checkbox"/>	9 minutes ago	10.9s	-	Jessy	C:\Users\...	-	sklearn	859.5	0.1
<input type="checkbox"/>	10 minutes ago	16.3s	-	Jessy	C:\Users\...	-	sklearn	401.4	-

MLFlow Interface for Model Management



PREFECT

ORCHESTRATE ML PIPELINES

Managing Machine Learning Workflows using the tool Prefect 2.0

Why Prefect?

- Python-based open source tool
- Manage ML Pipelines
- Schedule and Monitor the flow
- Gives observability into failures
- Native dask integration for scaling (Dask is used for parallel computing)

Creating And Activating A Virtual Environment

In order to install prefect, create a virtual environment:

```
$ python -m venv mlops
```

Enter the Virtual Environment using below mentioned command:

```
$ .\mlops\Scripts\activate
```

Installing Prefect 2.0

\$ pip install prefect

OR if you have Prefect 1, upgrade to Prefect 2 using this command:

\$ pip install -U prefect

OR to install a specific version:

\$ pip install prefect==2.4

Check Prefect Version

\$ prefect version

Running Prefect Dashboard

\$ prefect orion start

```
C:\Windows\System32\cmd.exe - prefect orion start
Microsoft Windows [Version 10.0.22000.978]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Jessa\Desktop\MLOPS\Orchestrate ML pipeline>.mlops_env\Scripts\activate
(mlops_env) C:\Users\Jessa\Desktop\MLOPS\Orchestrate ML pipeline>prefect orion start

PREFECT ORION

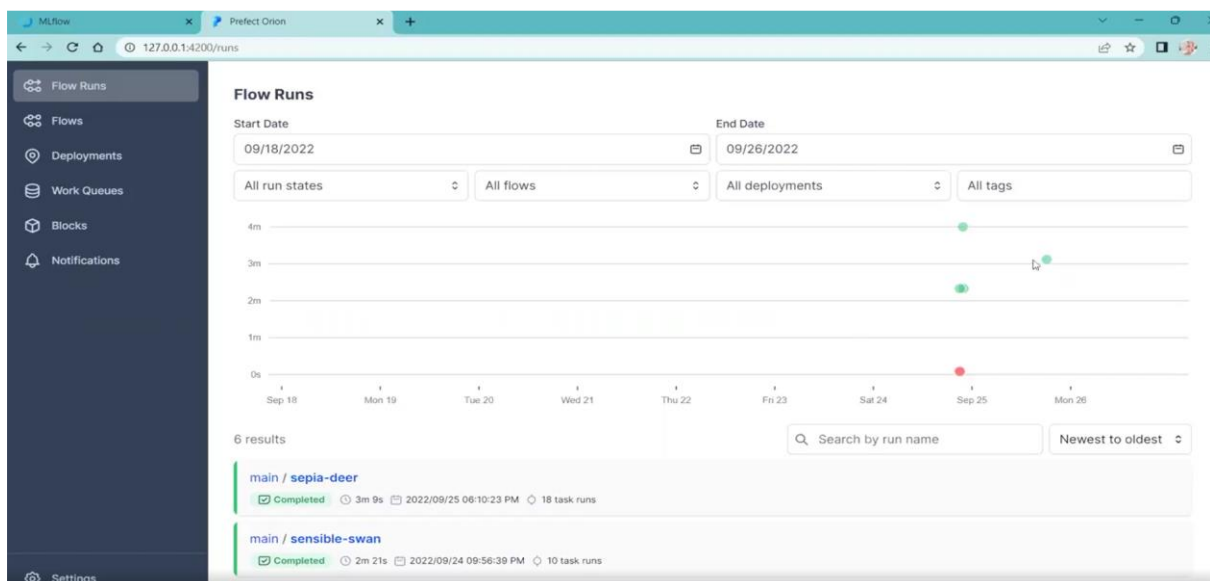
Configure Prefect to communicate with the server with:

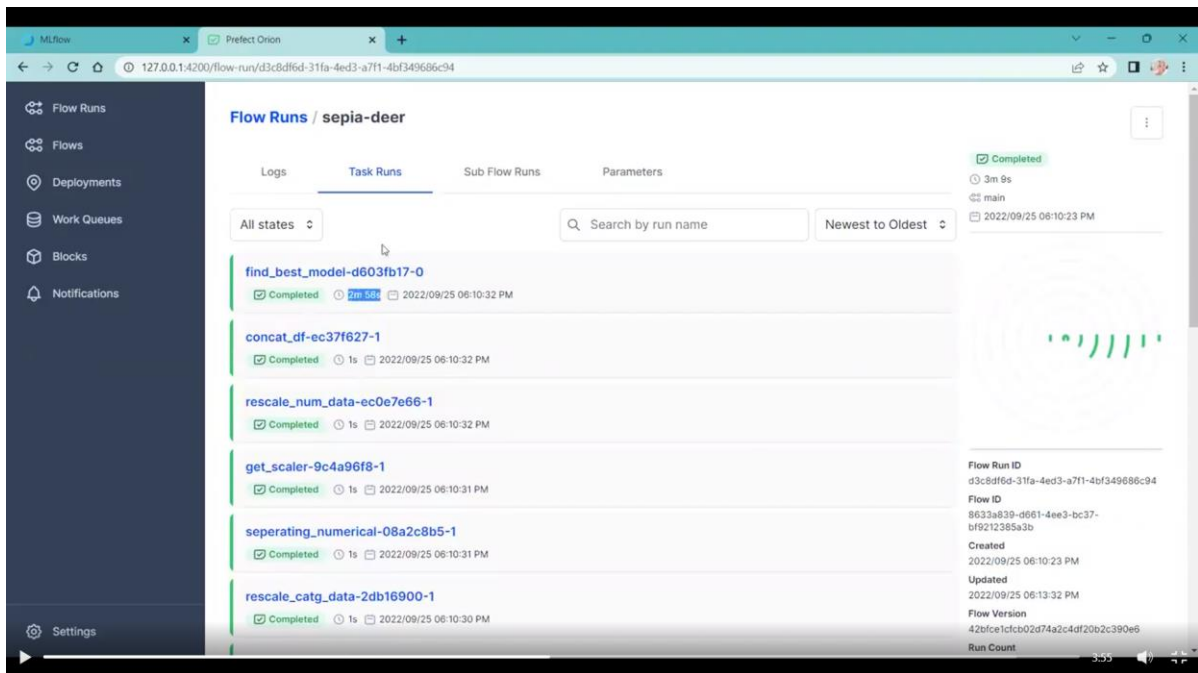
  prefect config set PREFECT_API_URL=http://127.0.0.1:4200/api

View the API reference documentation at http://127.0.0.1:4200/docs

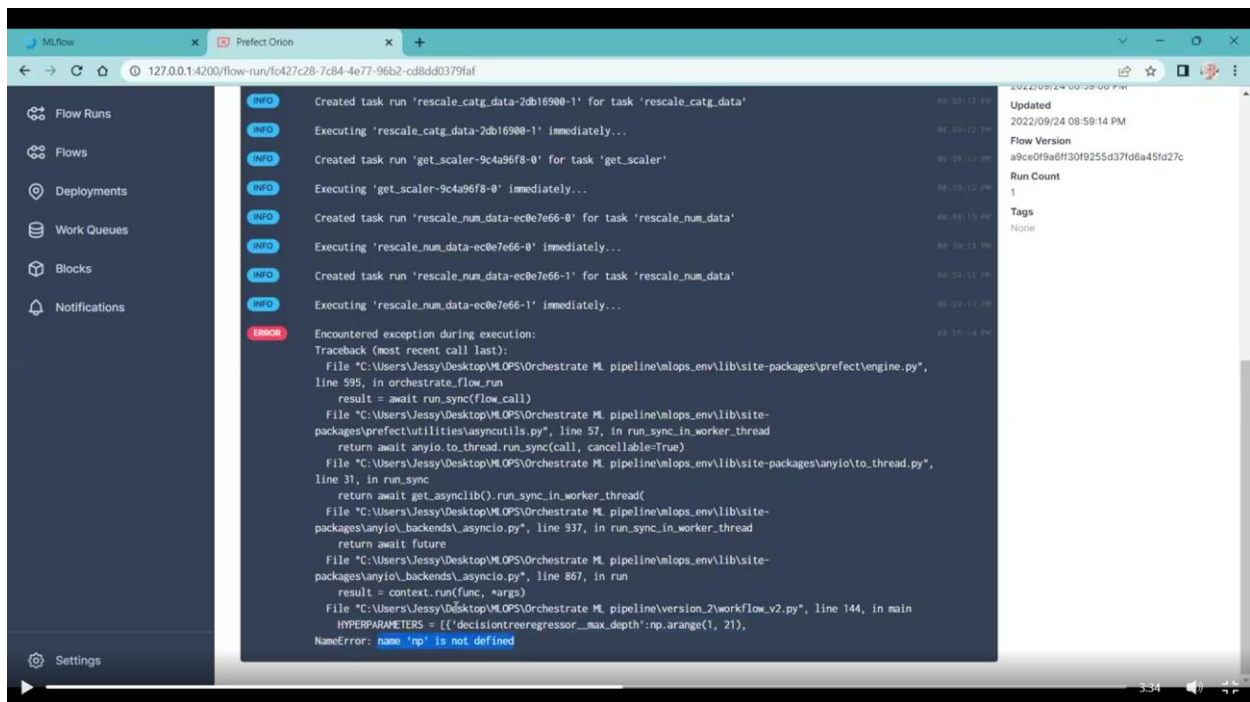
Check out the dashboard at http://127.0.0.1:4200
```

After starting the dashboard at <http://127.0.0.1:4200> the below screen is opened.

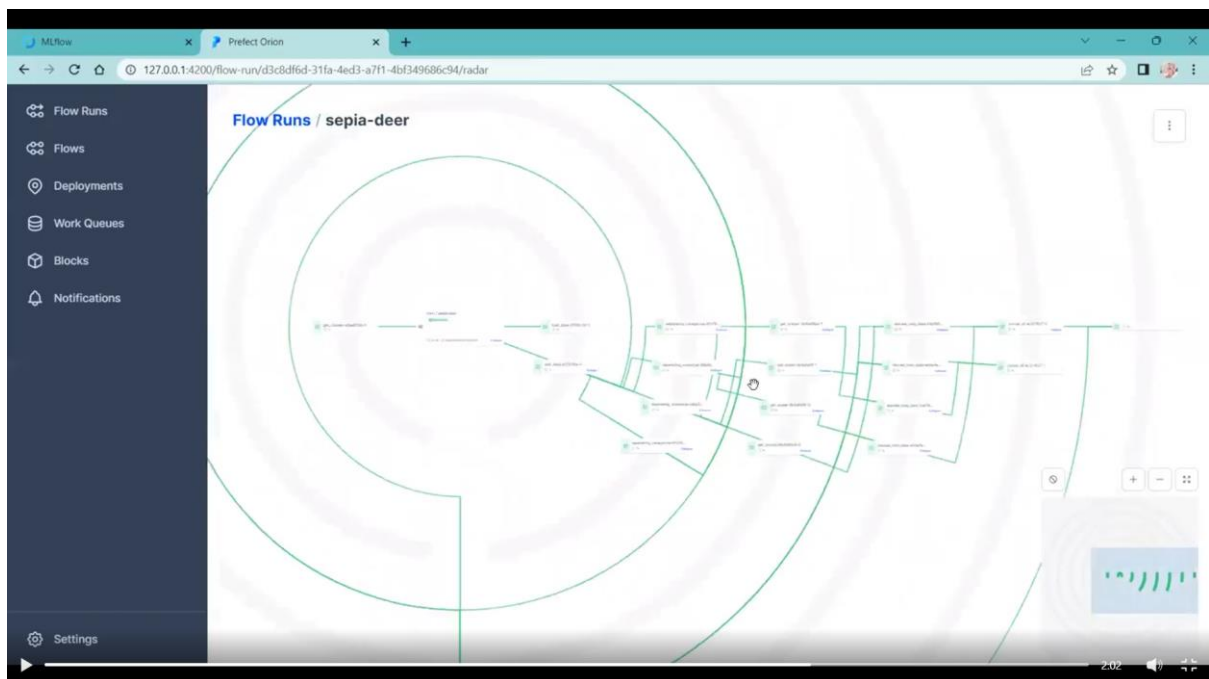




By using this prefect we can also detect the errors in a detailed manner.



Flow runs displayed as



Deployment of Prefect Flow

- `work_queue_name` is used to submit the deployment to the a specific work queue.
- You don't need to create a work queue before using the work queue. A work queue will be created if it doesn't exist.

The screenshot shows the Prefect UI interface for configuring a deployment. The sidebar on the left is the same as in the previous image. The main area is titled 'Flows / main' and has a 'Deployments' tab selected. Below the tab, it says '1 Deployment' and there is a search bar labeled 'Search deployments'. A table lists the deployment with the following columns: Name, Schedule, and Tags. The table contains one entry: 'main / model_training' with a schedule of 'Every 7 days' and a toggle switch that is currently turned on. To the right of the table, there is a 'Description' field with the value 'None'. Below the description, there is a 'Flow ID' field with the value '8633a839-d661-4ee3-bc37-bf9212385a3b'. At the bottom, there are 'Created' and 'Updated' fields, both showing the timestamp '2022/09/24 07:37:27 PM'.

Name	Schedule	Tags
main / model_training	Every 7 days	

Search deployments

Description: None

Flow ID: 8633a839-d661-4ee3-bc37-bf9212385a3b

Created: 2022/09/24 07:37:27 PM

Updated: 2022/09/24 07:37:27 PM

Running an Agent

\$ `prefect agent start --work-queue "ml"`

The screenshot shows the Prefect Orion web interface in a browser. The address bar shows the URL `127.0.0.1:4200/work-queue/c5867743-d400-48d5-848e-af0027ac183d`. The left sidebar contains navigation links: Flow Runs, Flows, Deployments, Work Queues, Blocks, and Notifications. The main content area is titled "Work Queues / ml" and features a blue banner that says "Work queue is ready to go!" with a subtext "Work queues define the work to be done and agents poll a specific work queue for new work." Below the banner is a terminal window showing the command `prefect agent start --work-queue "ml"`. Under the "Upcoming Runs" section, there is a table of scheduled runs:

Run Name	Branch	Deployment	Status	Scheduled For
main / mysterious-porcupine	main	auto-scheduled	Scheduled	2022/10/02 06:17:50 PM
main / provocative-polecat	main	auto-scheduled	Scheduled	2022/10/09 06:17:50 PM
main / sticky-mastodon	main	auto-scheduled	Scheduled	2022/10/16 06:17:50 PM
main / just-gerbil	main	auto-scheduled	Scheduled	2022/10/23 06:17:50 PM

On the right side, a "Description" panel shows the following details:

- Work Queue ID: `c5867743-d400-48d5-848e-af0027ac183d`
- Flow Run Concurrency: `None`
- Created: 2022/09/24 09:41:12 PM