INNOMATICS®
RESEARCH LABS

INNOVATION. AUTOMATION. ANALYTICS

PROJECT ON

Using MLflow for Experiment Tracking and Model Management and Prefect - Sentiment Analysis of Flipkart Reviews

Prepared by Eleshala Pravalika

Objective of the Report:

- The objective is to introduce MLflow for experiment tracking, model management, and reproducibility in **Sentiment Analysis of Flipkart reviews**.

MLFlow:

MLFlow: Unified Platform for Experiment Tracking and Model Registry

MLflow is an open-source platform for managing the end-to-end machine learning lifecycle. It provides a suite of tools and components designed to streamline the development, experimentation, productionisation, and collaboration aspects of machine learning projects. MLflow is widely used by data scientists, machine learning engineers, and researchers to track experiments, package and share code, and deploy models at scale.

Key Features:

1. Experiment Tracking
2. Model Registry

Integration of MLflow into projects:

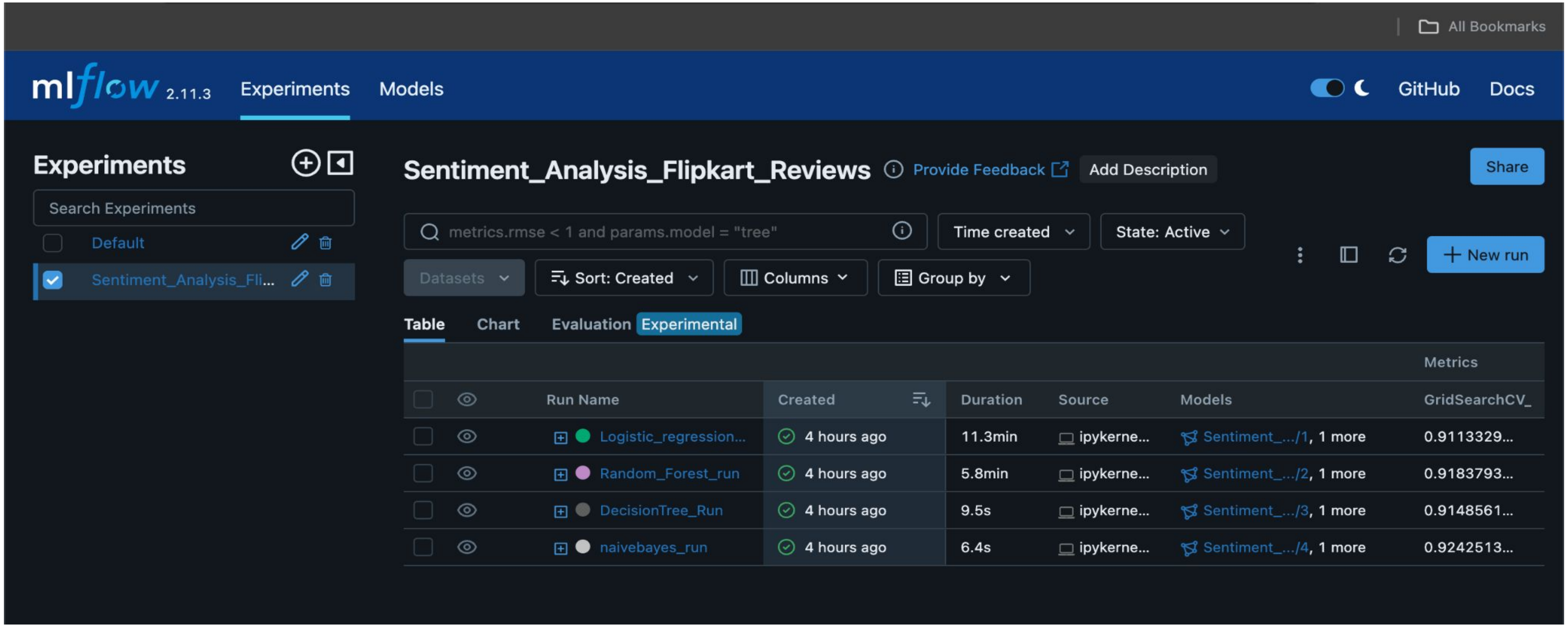
```
pip install mlflow
mlflow ui

import mlflow
mlflow.set_experiment("Sentiment_Analysis_Flipkart_Reviews")

mlflow.sklearn.autolog(max_tuning_runs=None)

with mlflow.start_run() as run:
    %time grid_search.fit(X_train, y_train)
```

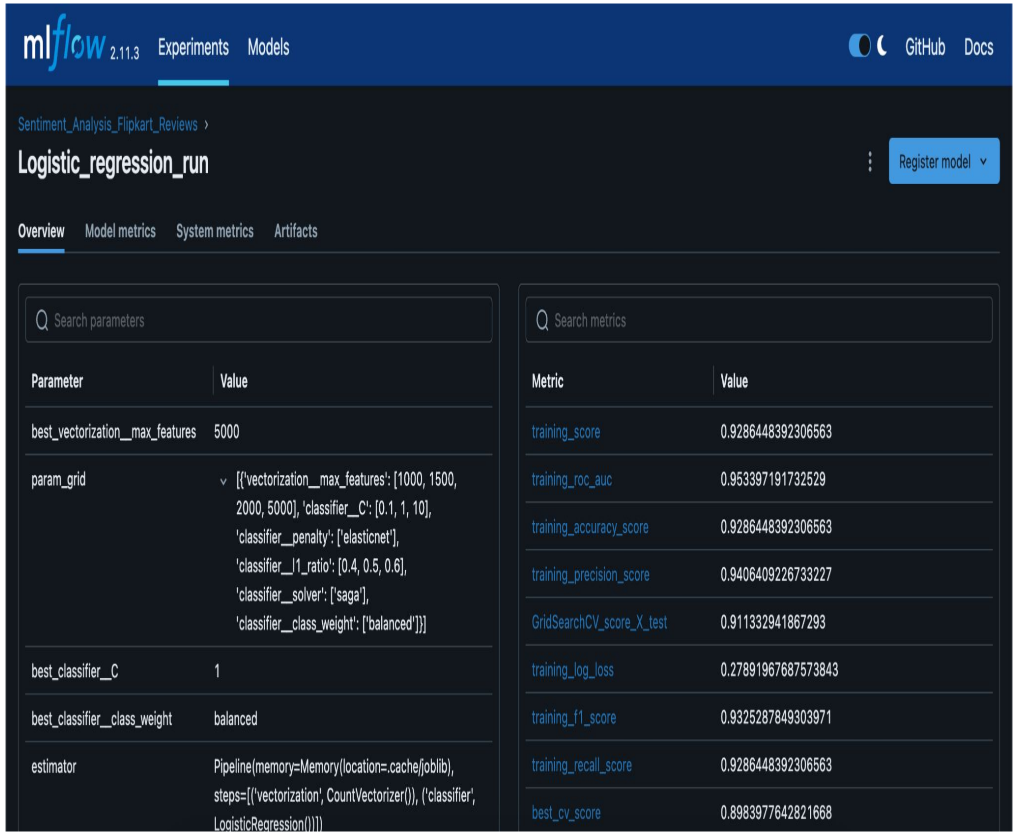
MLflow dashboard:



Sentiment_Analysis_Flipkart_Reviews

Run Name	Created	Duration	Source	Models	Metrics: GridSearchCV_
Logistic_regression...	4 hours ago	11.3min	ipykerne...	Sentiment_.../1, 1 more	0.9113329...
Random_Forest_run	4 hours ago	5.8min	ipykerne...	Sentiment_.../2, 1 more	0.9183793...
DecisionTree_Run	4 hours ago	9.5s	ipykerne...	Sentiment_.../3, 1 more	0.9148561...
naivebayes_run	4 hours ago	6.4s	ipykerne...	Sentiment_.../4, 1 more	0.9242513...

Demonstration of logging parameters, metrics, and artifacts using MLflow tracking APIs:

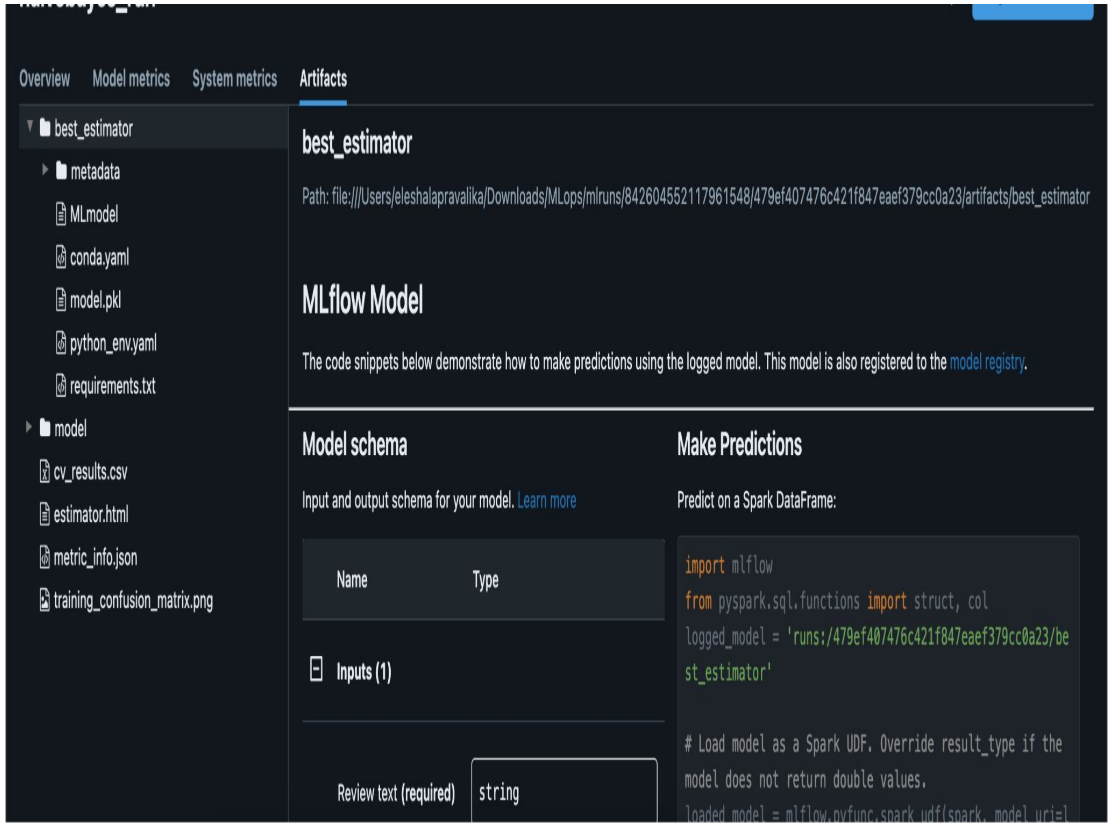


Sentiment_Analysis_Flipkart_Reviews ›

Logistic_regression_run

Parameter	Value
best_vectorization__max_features	5000
param_grid	[{'vectorization__max_features': [1000, 1500, 2000, 5000], 'classifier__C': [0.1, 1, 10], 'classifier__penalty': ['elasticnet'], 'classifier__l1_ratio': [0.4, 0.5, 0.6], 'classifier__solver': ['saga'], 'classifier__class_weight': ['balanced']}]
best_classifier__C	1
best_classifier__class_weight	balanced
estimator	Pipeline(memory=Memory(location=cache/joblib), steps=[('vectorization', CountVectorizer()), ('classifier', LogisticRegression())])

Metric	Value
training_score	0.9286448392306563
training_roc_auc	0.953397191732529
training_accuracy_score	0.9286448392306563
training_precision_score	0.9406409226733227
GridSearchCV_score_X_test	0.911332941867293
training_log_loss	0.27891967687573843
training_f1_score	0.9325287849303971
training_recall_score	0.9286448392306563
best_cv_score	0.8983977642821668



best_estimator

Path: file:///Users/eleshalapravalika/Downloads/MLops/mlruns/842604552117961548/479ef407476c421f847eaef379cc0a23/artifacts/best_estimator

MLflow Model

The code snippets below demonstrate how to make predictions using the logged model. This model is also registered to the model registry.

Model schema

Input and output schema for your model. Learn more

Name	Type
Inputs (1)	
Review text (required)	string

Make Predictions

Predict on a Spark DataFrame:

```
import mlflow
from pyspark.sql.functions import struct, col
logged_model = 'runs:/479ef407476c421f847eaef379cc0a23/best_estimator'

# Load model as a Spark UDF. Override result_type if the model does not return double values.
loaded_model = mlflow.pyfunc.spark_udf(spark, model_uri=l
```

INNOMATICS RESEARCH LABS

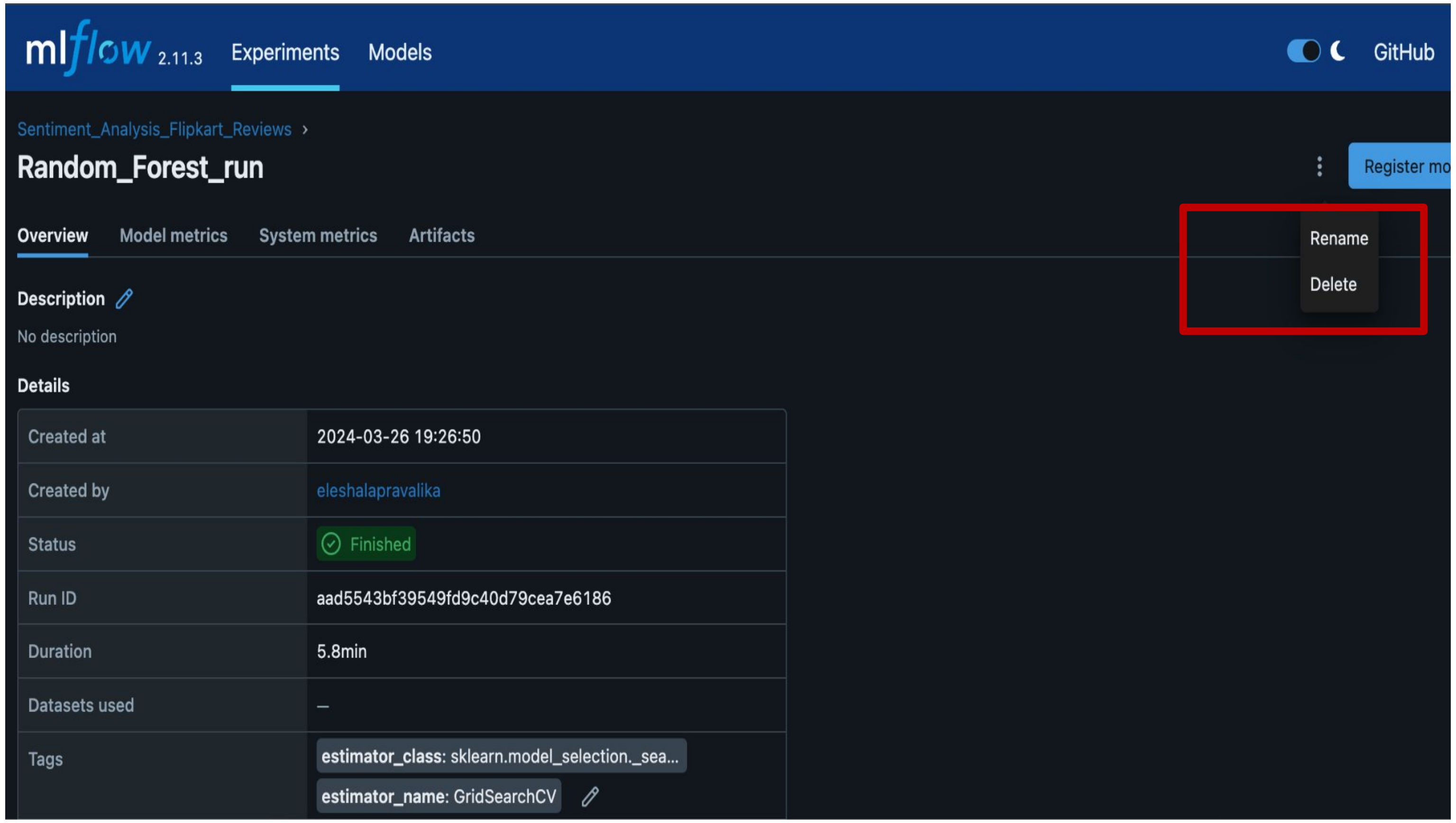
Demonstration of logging parameters, metrics, and artifacts using MLflow tracking APIs:

Click on Experiment_name → Run name→ (scroll down for) → Parameters , Metrics, Artifacts.

Demonstration of logging parameters, metrics, and artifacts using MLflow tracking APIs:

Click on the attachment for video

Customizing Mlflow UI with run names:



Sentiment_Analysis_Flipkart_Reviews ›

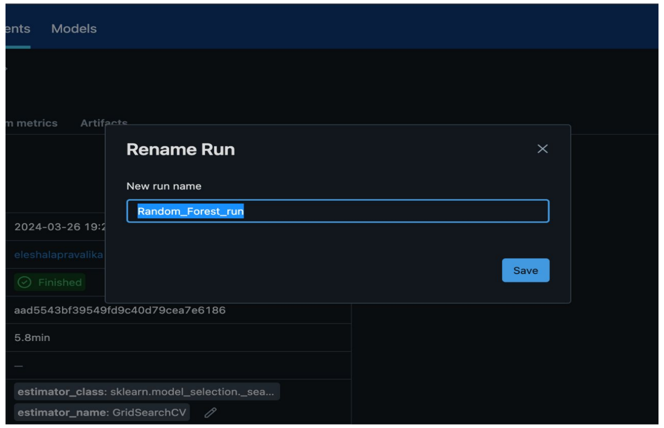
Random_Forest_run

Description

No description

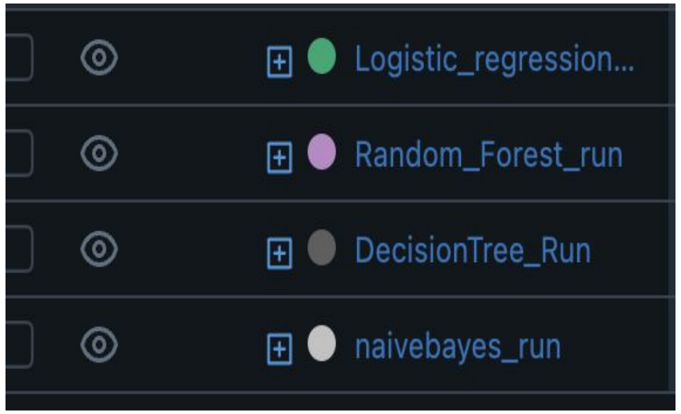
Details

Created at	2024-03-26 19:26:50
Created by	eleshalapravalika
Status	Finished
Run ID	aad5543bf39549fd9c40d79cea7e6186
Duration	5.8min
Datasets used	—
Tags	estimator_class: sklearn.model_selection._sea... estimator_name: GridSearchCV



Rename Run

New run name: Random_Forest_run



Logistic_regression...

Random_Forest_run

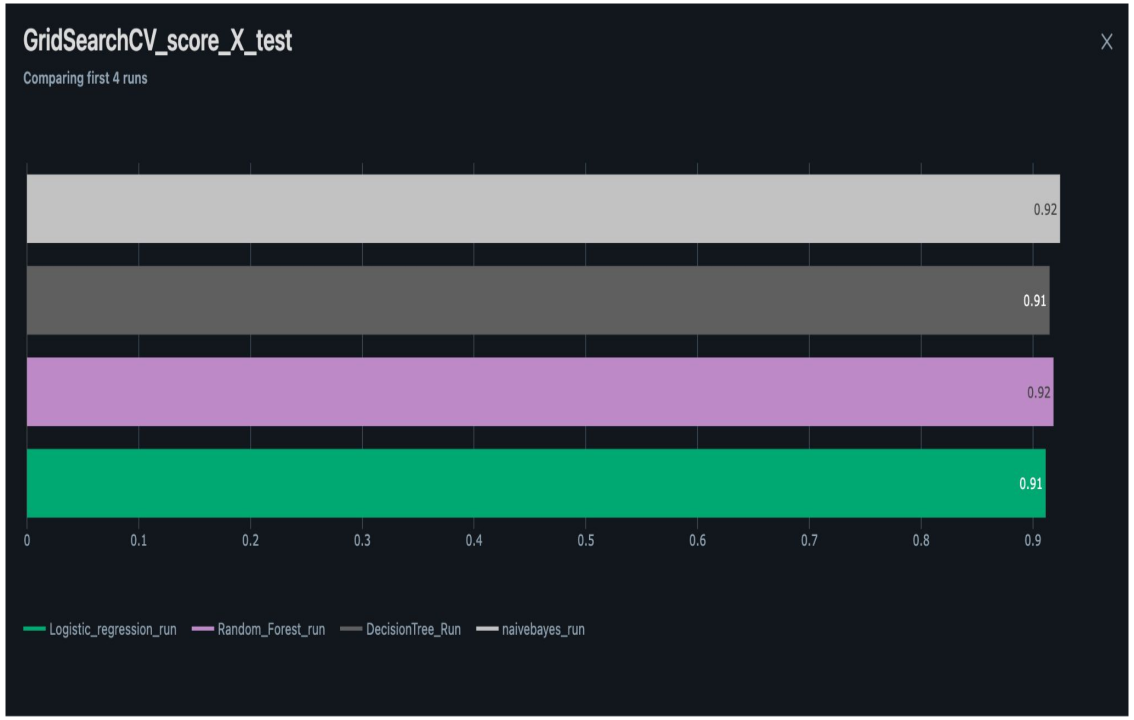
DecisionTree_Run

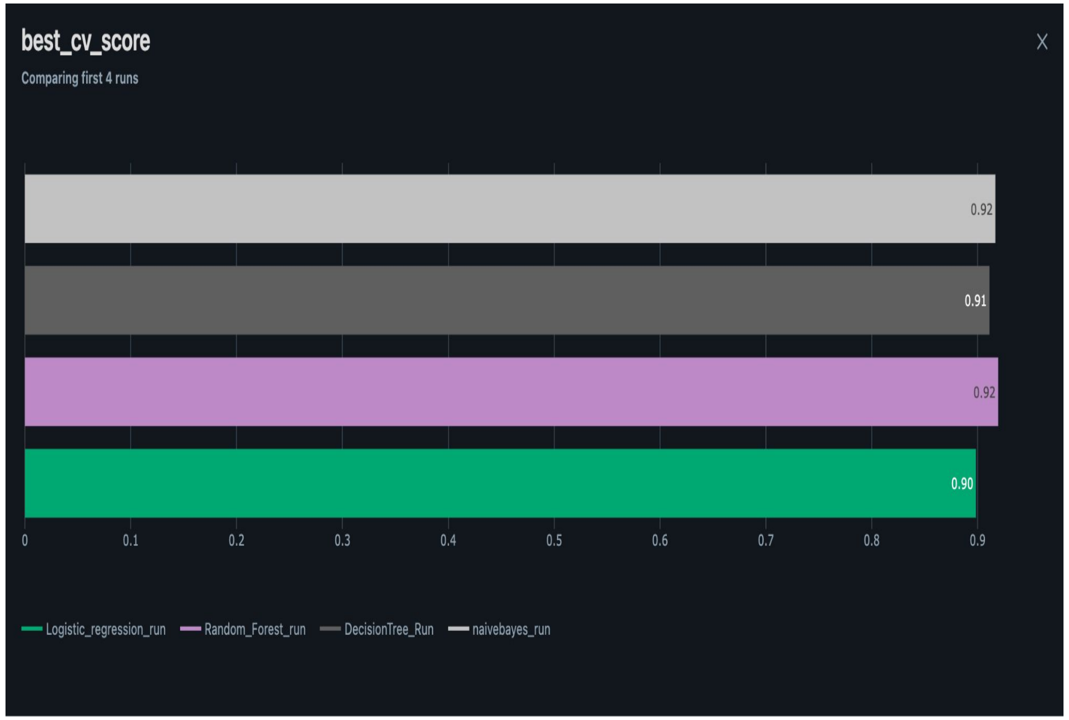
naivebayes_run

Customizing Mlflow UI with run names:

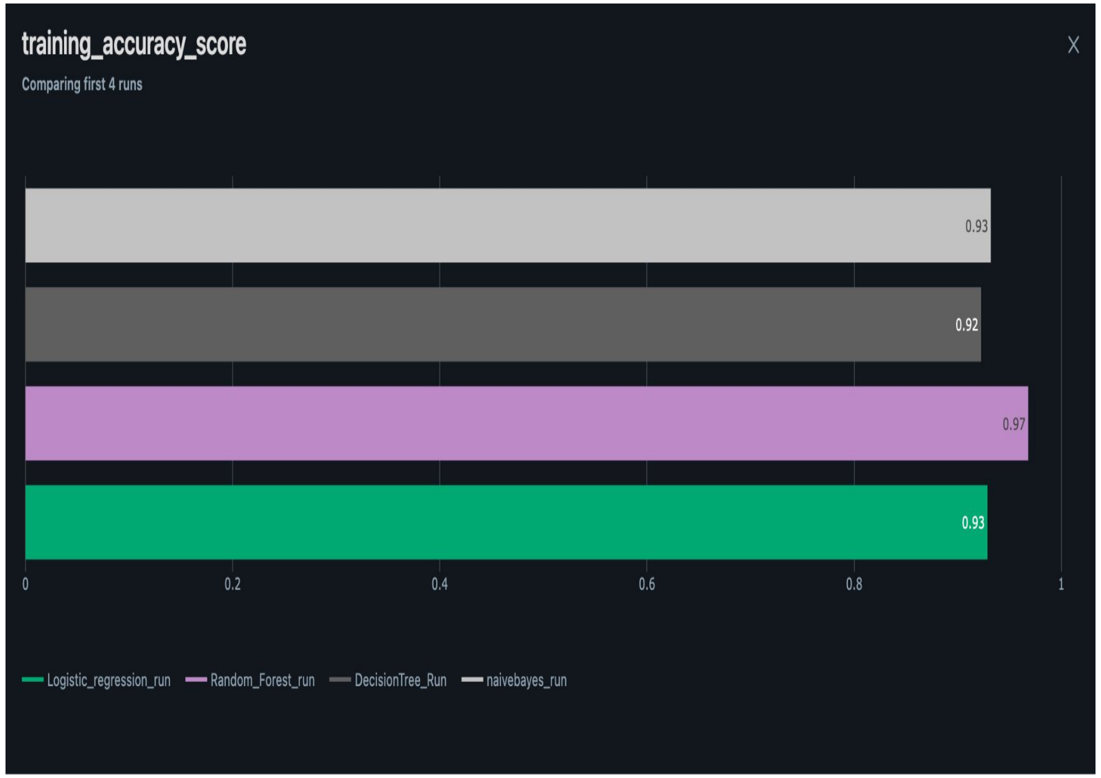
Click on Run name⟶Right corner : 3 dots⟶ Rename ⟶ Give the name for run⟶ Save.

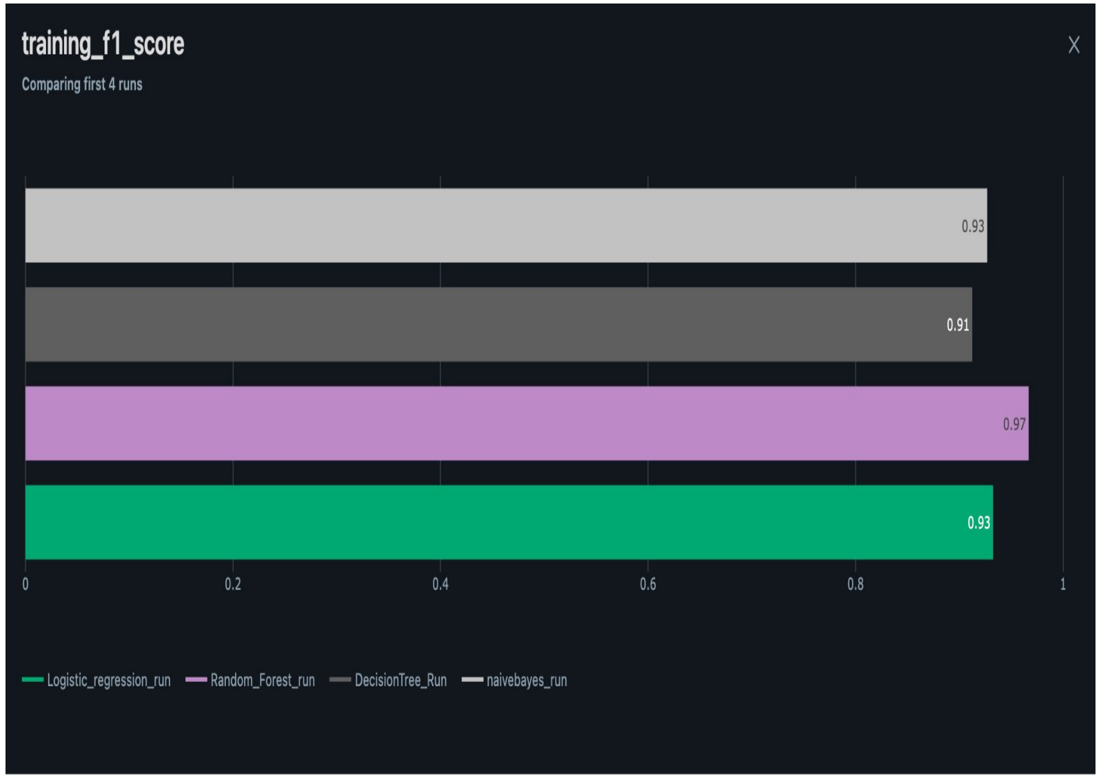
Metric Plots :





Metric Plots :



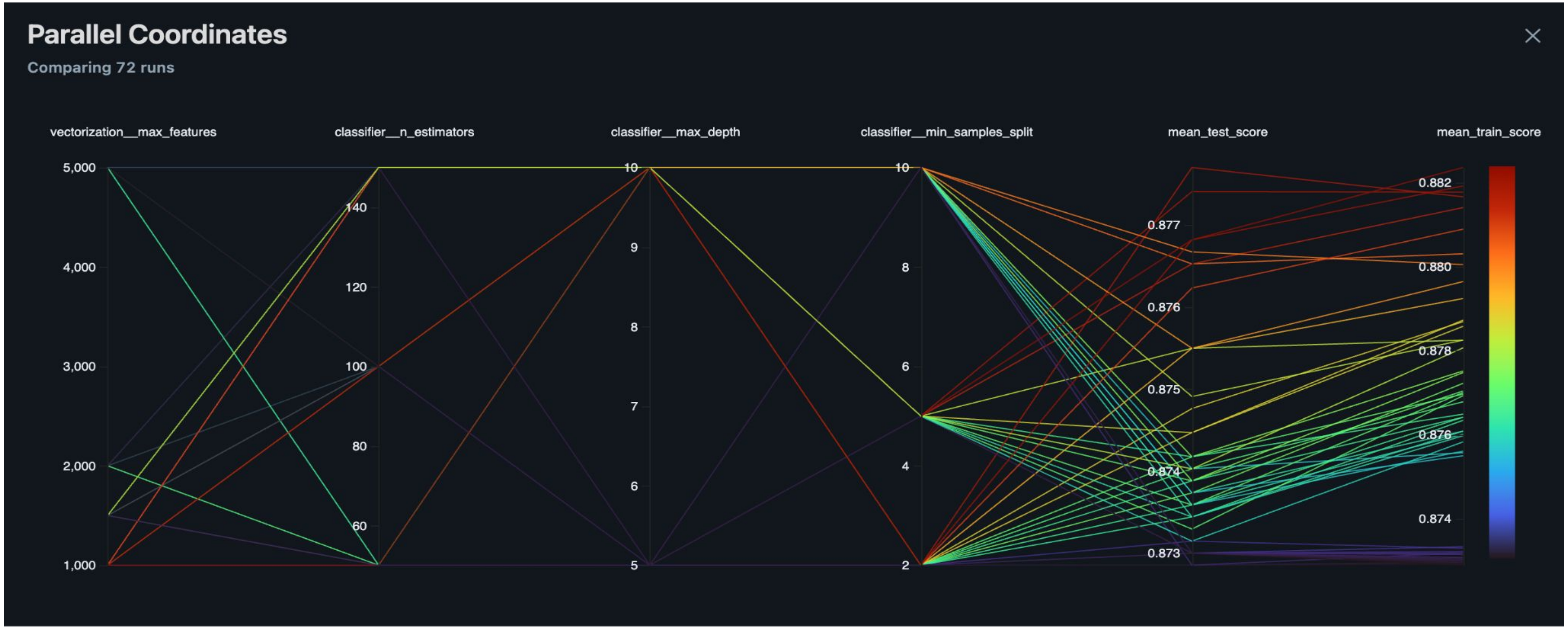


INNOMATICS RESEARCH LABS

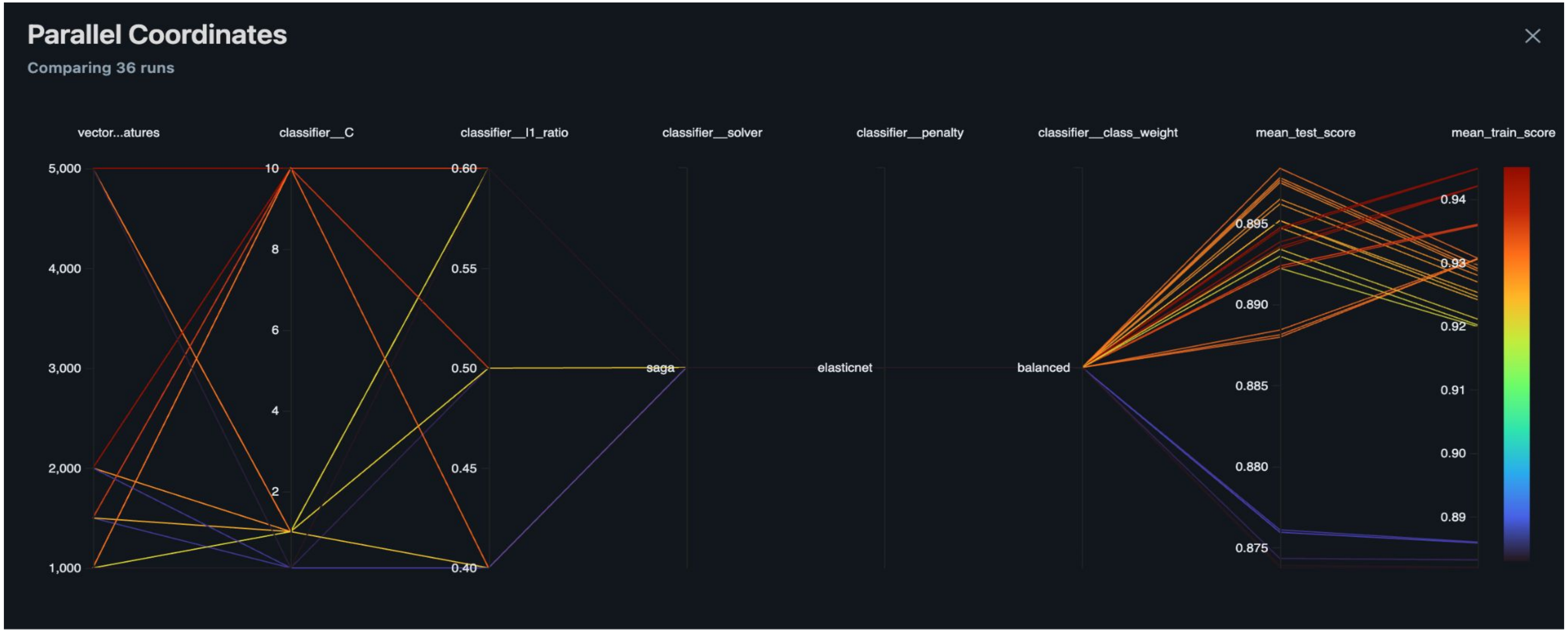
HyperParameter Plots Creation:

Click on charts⟶ Add Section ⟶ Name hyperparameter with model ⟶ Add Parallel coordinates chart ⟶ give parameters and metrics of particular run ⟶ click on the runname’s “+” : This will pop the interactive chart

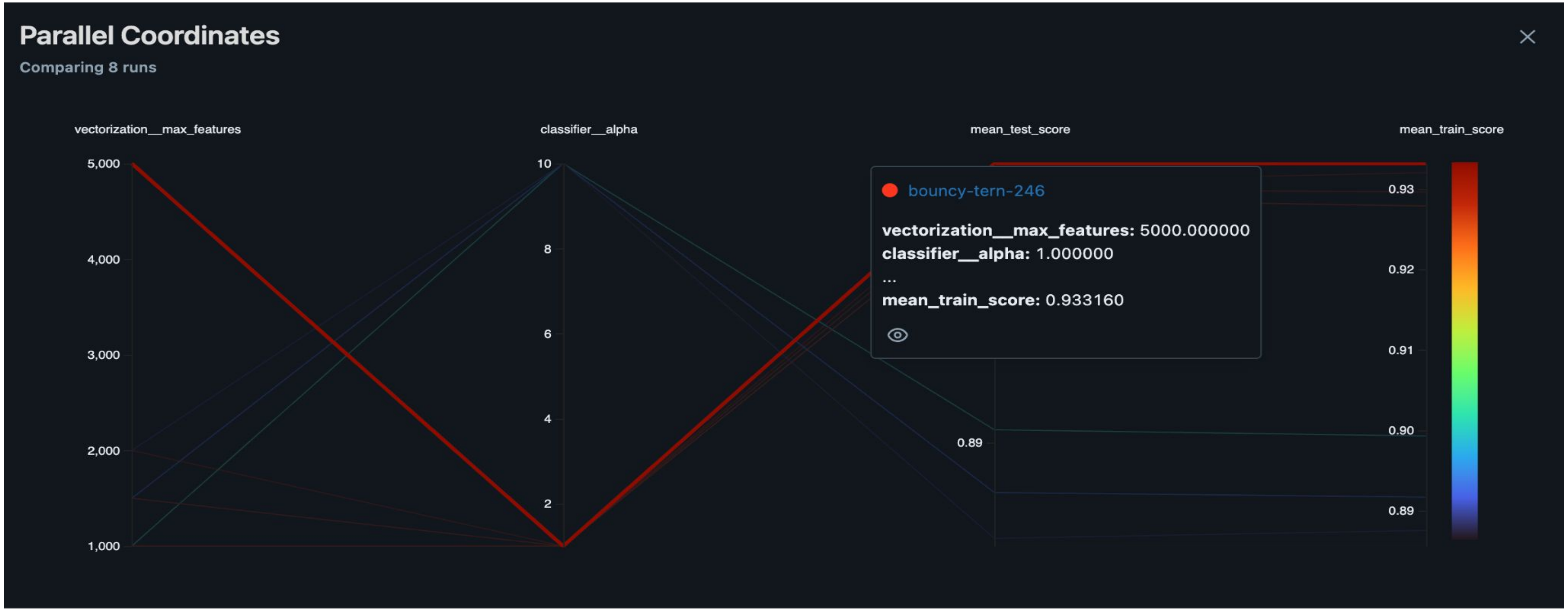
HyperParameter Plots :



HyperParameter Plots :



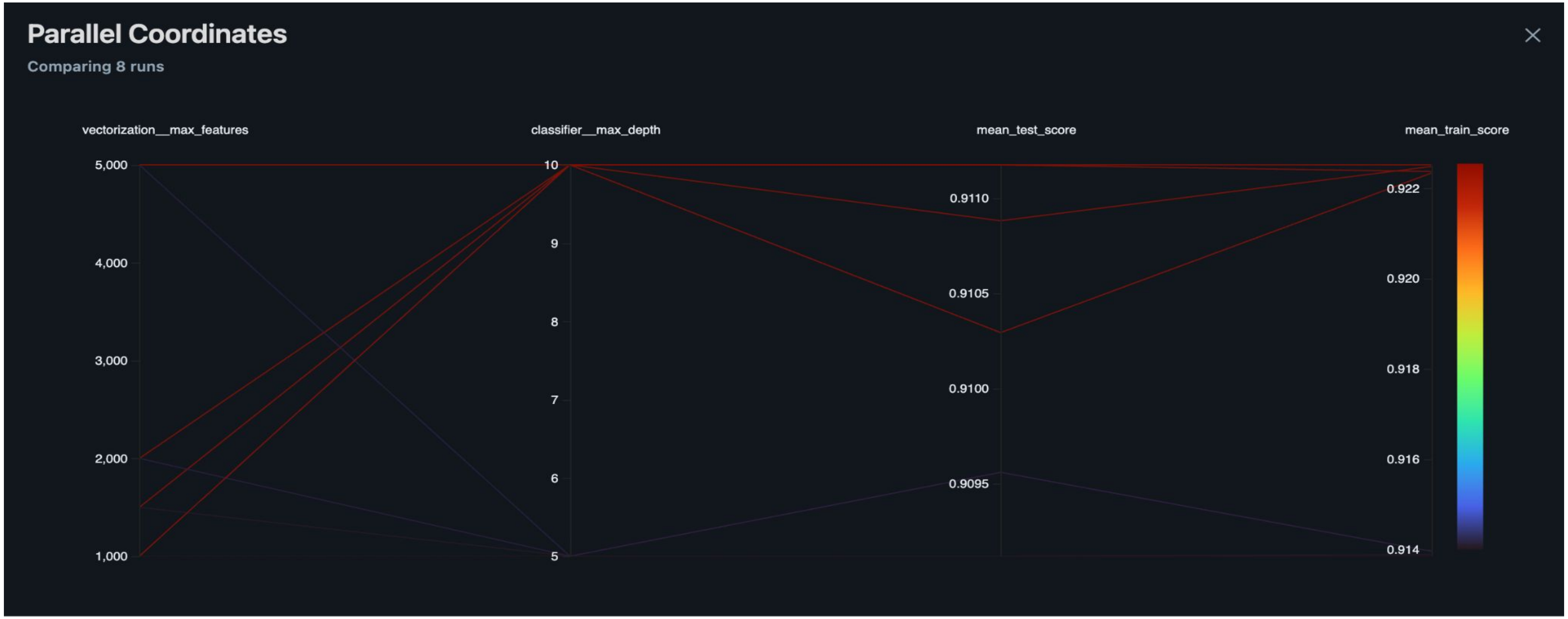
HyperParameter Plots :



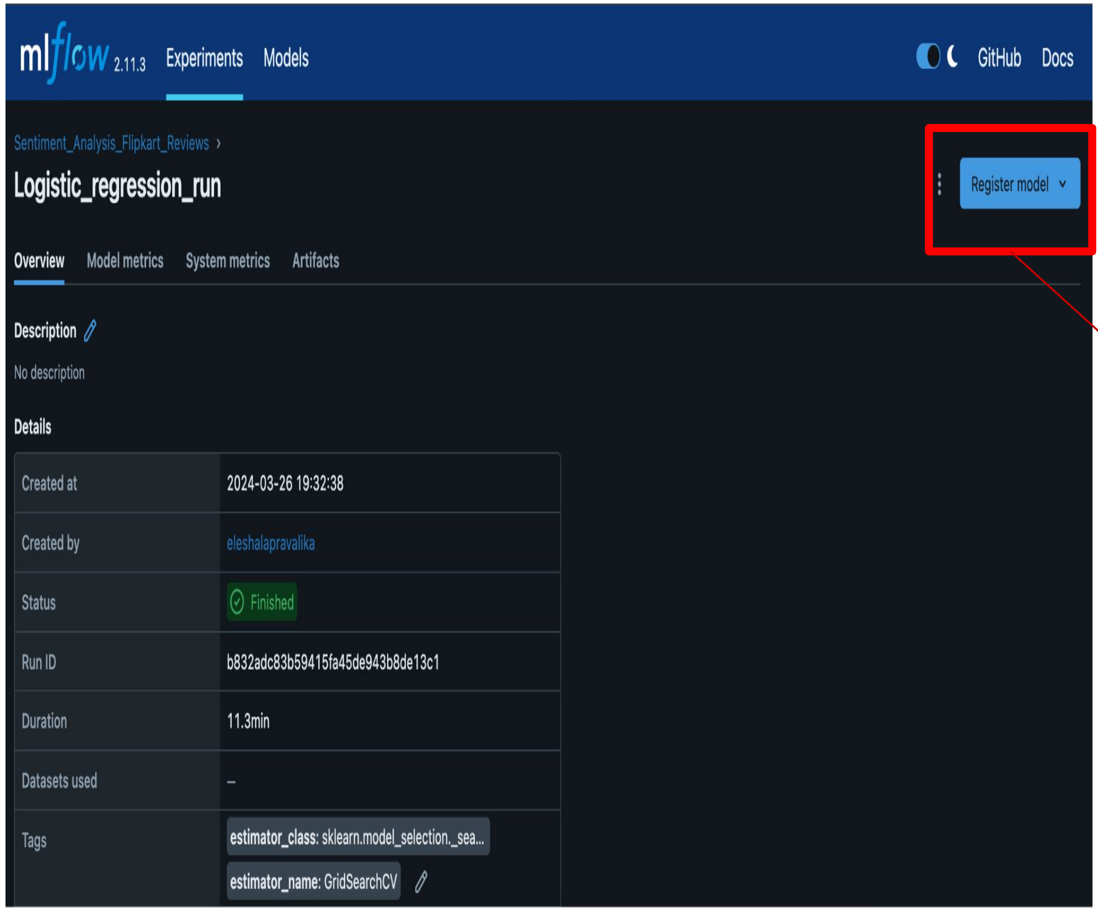
● bouncy-tern-246

vectorization__max_features: 5000.000000
classifier__alpha: 1.000000
...
mean_train_score: 0.933160

HyperParameter Plots :



Registering models and Managing with tags:



Sentiment_Analysis_Flipkart_Reviews ›

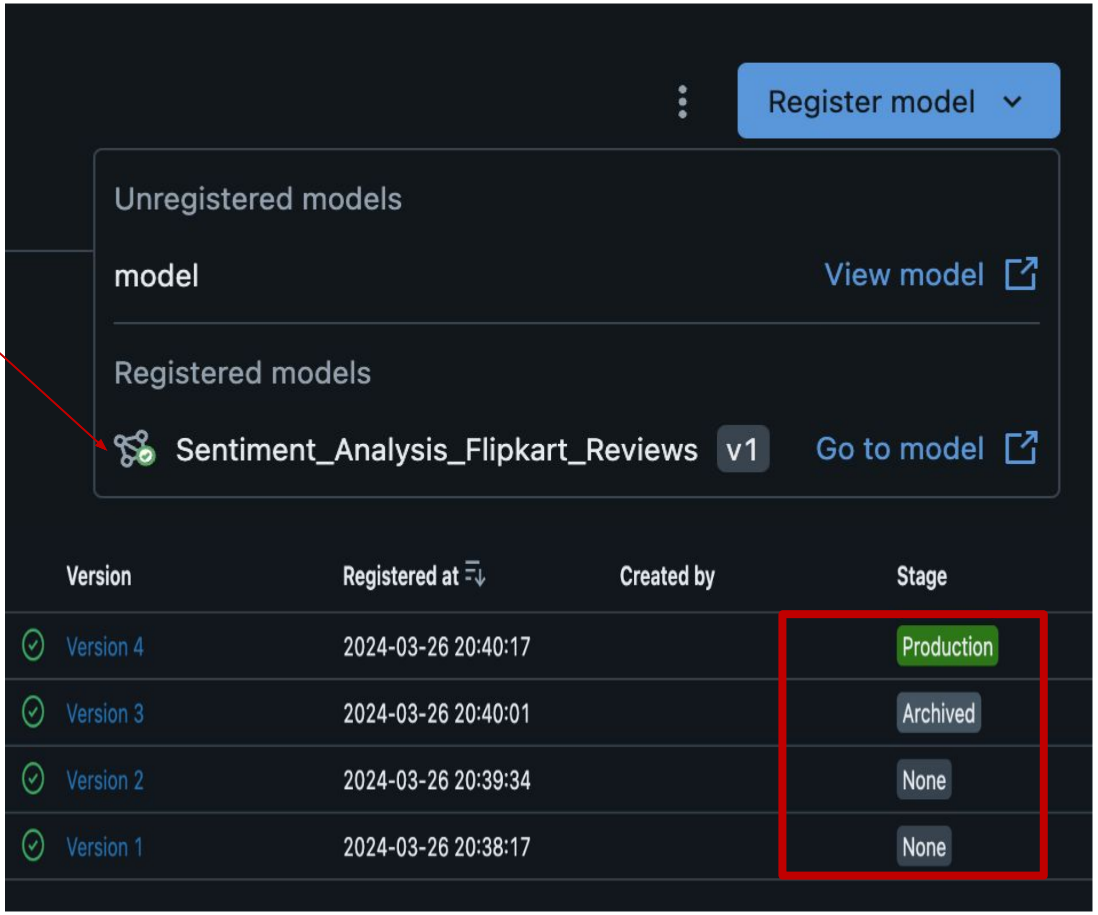
Logistic_regression_run

Description

No description

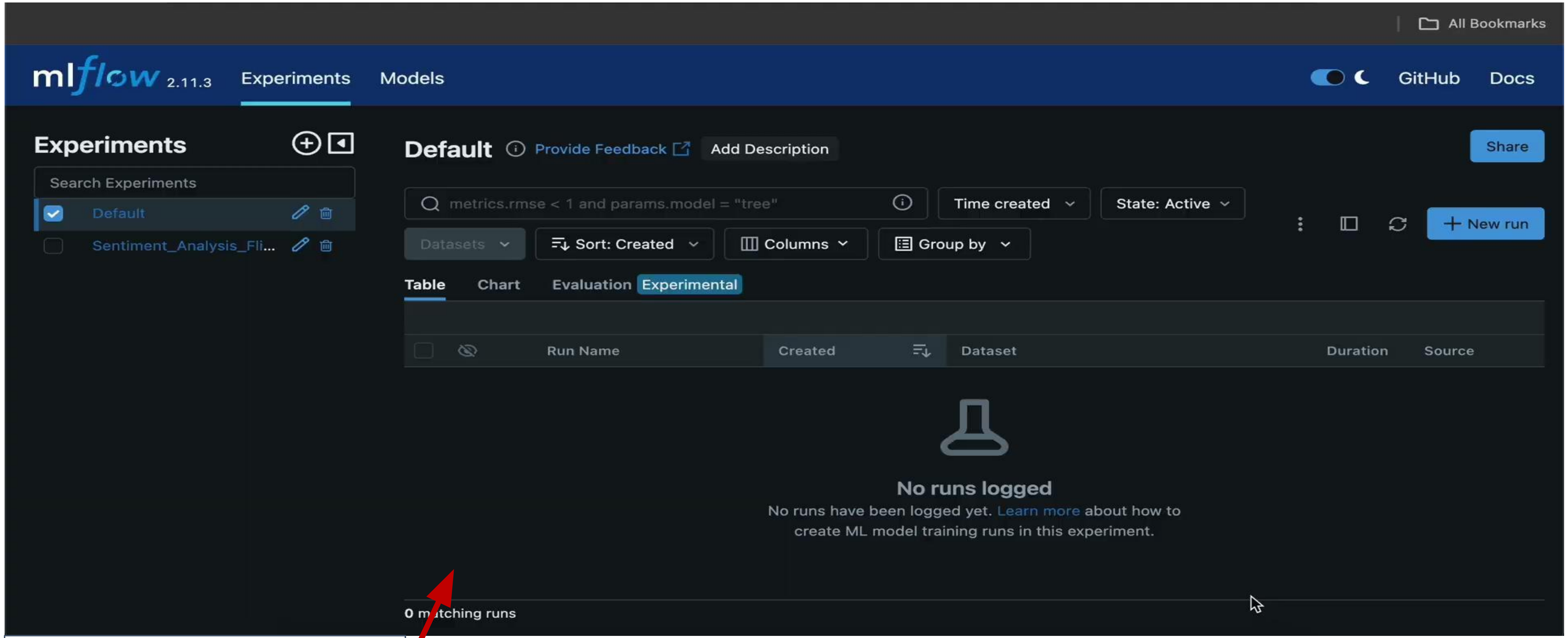
Details

Created at	2024-03-26 19:32:38
Created by	eleshalapravalika
Status	Finished
Run ID	b832adc83b59415fa45de943b8de13c1
Duration	11.3min
Datasets used	–
Tags	estimator_class: sklearn.model_selection._sea... estimator_name: GridSearchCV



Version	Registered at	Created by	Stage
Version 4	2024-03-26 20:40:17		Production
Version 3	2024-03-26 20:40:01		Archived
Version 2	2024-03-26 20:39:34		None
Version 1	2024-03-26 20:38:17		None

MLflow Experiment Tracking and Model Management



No runs logged

No runs have been logged yet. Learn more about how to create ML model training runs in this experiment.

0 matching runs

Click on the attachment for video

Prefect:

- Prefect is an open-source orchestration and observability platform that empowers developers to build and scale resilient code quickly, turning their Python scripts into resilient, recurring workflows.
- Prefect streamlines the orchestration of machine learning workflows by providing a flexible, scalable, and reliable framework for building, deploying, and managing complex data pipelines with ease.
- It empowers data scientists and engineers to focus on building machine learning models and solving business problems while abstracting away the complexities of workflow management and execution.

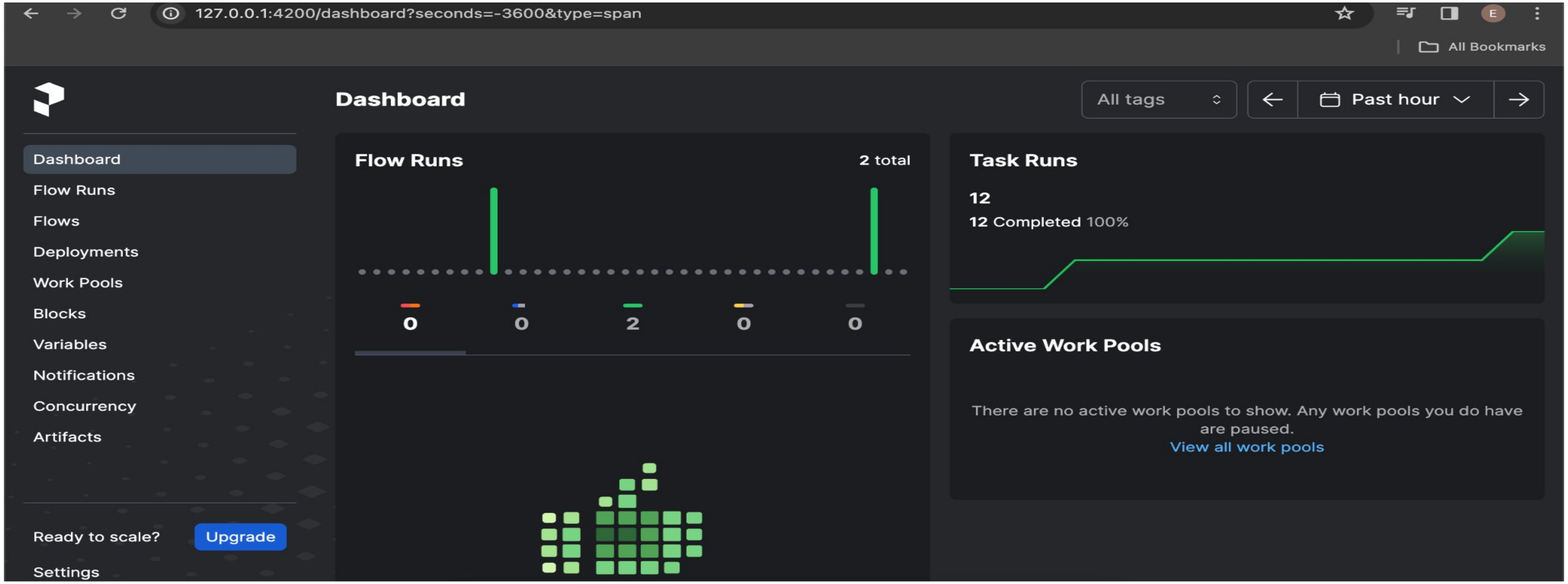
Prefect- Installation:

```
pip install prefect
```

```
prefect server start
```

```
(myenv) (base) eleshalapravalika@Eleshalas-MBP Prefect % prefect server start
 ___ ___ ___ ___ ___ ___ _____
| _ \ _ \ __| __| __/ __|_   _|
|  _/   / _|| _|| _| (__  | |
|_| |_|_\___|_| |___\___| |_|
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
```

Prefect- DashBoard:



127.0.0.1:4200/dashboard?seconds=-3600&type=span

All Bookmarks

Dashboard

All tags

Past hour

Dashboard
Flow Runs
Flows
Deployments
Work Pools
Blocks
Variables
Notifications
Concurrency
Artifacts

Ready to scale? Upgrade

Settings

Flow Runs 2 total

0 0 2 0 0

Task Runs

12

12 Completed 100%

Active Work Pools

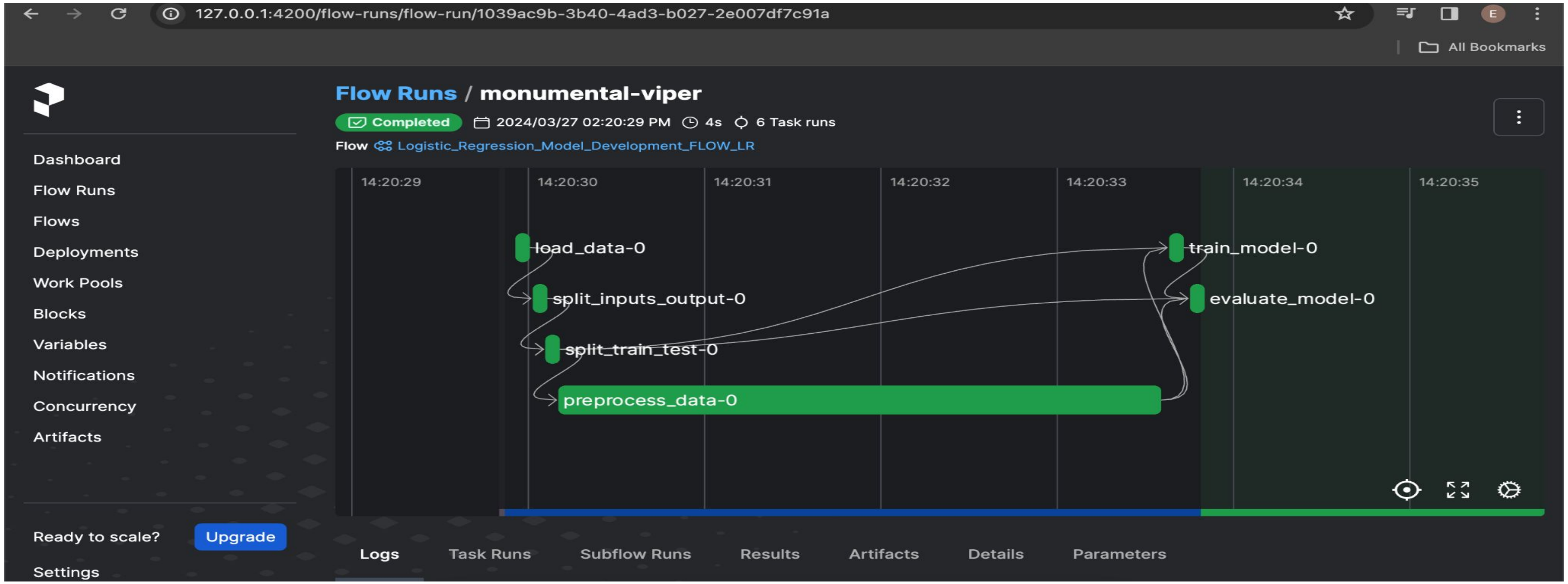
There are no active work pools to show. Any work pools you do have are paused.

View all work pools

Building a Prefect Workflow:

Import Prefect modules → Define Prefect Tasks → Define Prefect Flow → Run Prefect Flow

Prefect- Workflow:



INNOMATICS RESEARCH LABS

THANK YOU

INNOMATICS RESEARCH LABS