

MLOps and comparison of different models

- *Irfan Toor (OpenClassrooms | Projet No. 7)*
- *keywords*: MLOps, MLFlow, traçage, organisation, modèles m/l, optimisation, hyper-paramètres, versioning, autolog, grid-search, reproduction, erreurs, package, partage, model simple, LSTM, BERT, transfer-learning, comparaison, embeddings, word2vec, Glove.6B, TweetEval, roBERTa, sentiment, sst-2, analysis

ABSTRACT

"I think we don't last if we lose our bearings.¹»

— David de Rothschild

Artificial Intelligence has the potential to change the way humans interact and behave with the existing or the new phenomenas of the world. The automatic collection of data, it's cleaning, processing or feature engineering and the ever increasing compute-power are improving the existing models at an exponential rate. More inclined we are to this ever changing environment and the it's associated dependencies, which are changing at a phenomenal rate, more we are prone to the risks associated with finding ourself in *the unknown lands and the territories of the machine control*, thereby falling prey to this self invented trap and to an ultimate oblivion.

AI is not a new technology, which is taking the world by storm but its the *logical evolution* of what we have already been creating through machine learning models. We, the humans, have understood after the experience of millenniums that *keeping track of whatever we are doing is fundamental* to our boundless potential, flawless evolution and behemoth success in every domain we touch. So, its quite natural that we transfer and apply the knowledge and the concepts learned in one domain to another, which could not have been possible without tracking every bit of our experiments -- be it *psychology or science*.

The concepts of DevOps have found their way into the M/L world, not only facilitating the versioning, logging and tracking of models but are extended into keeping track of the data, the parameters and the performance of these models as well. This concept of tracking and managing all of the life-cycle of M/L models, the associated data, their versioning, performance and parameter tracking and even staging from development to deployment is handled by **Machine/Learning Operations** i.e MLOps. *We will discuss about MLOps, its deployment, and its usage through the comparison of three models namely: **Simple, LSTM and BERT**.*

¹ « Je crois qu'on ne perdure pas si on perd les repères. »

1. INTRODUCTION

1.1 MLOps - Experiment tracking

What is MLOps?

MLOps is to Machine Learning models, as **DevOps** is to development. Since the requirements for Machine Learning are more than that of the Development, the MLOps have more to manage and handle as compared to DevOps. Its not about keeping track of the code, the dependencies or staging alone, but that of the data versioning, hyper-parameters, performance metrics also.

Furthermore the life cycles of the models, their different versions, development, staging and deployment etc. are all managed by a combination of tools, protocols, policies and practices known as MLOps.

How an MLOps paradigm helps us?

MLOps paradigm gives us the following advantages:

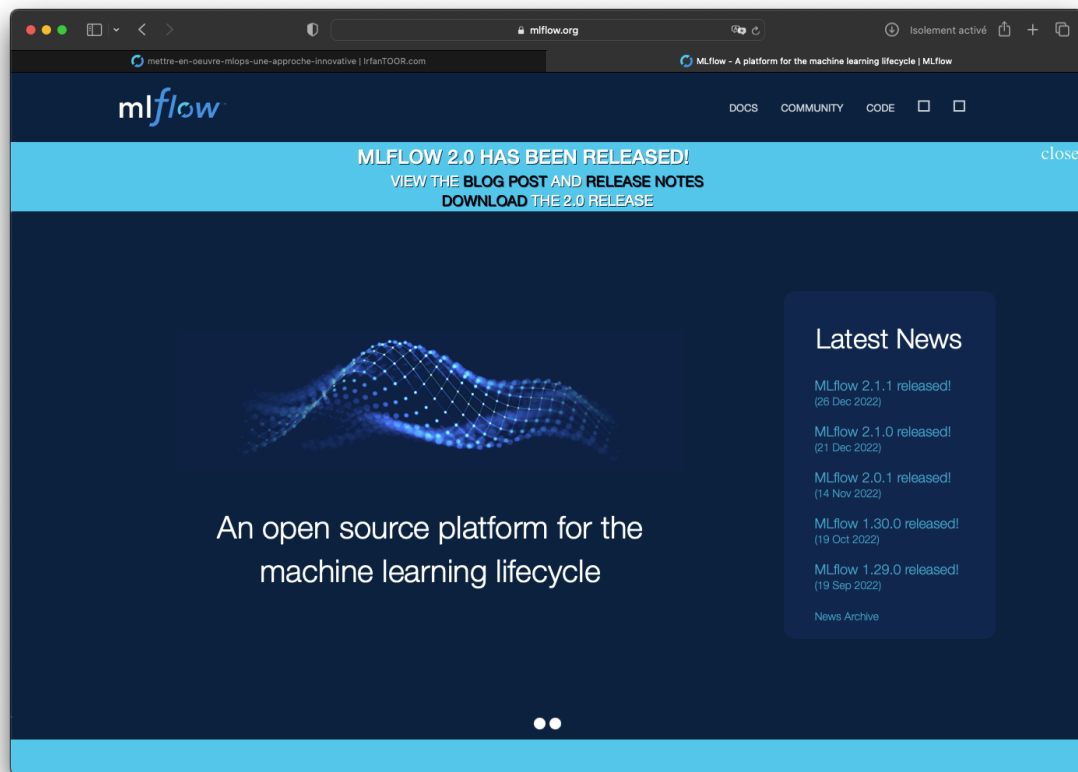
- For reproducibility of errors or the state of models
- Organisation of different models, data, hyper-parameters and the results etc.
- Optimisation of models/hyper-parameters
- Automatic logging
- Consistance between the standards of logging

Why the other classic techniques like spread-sheets are not so effective?

- Possibility of incorporating errors due to its manual logging nature.
- No standards have so far been established regarding the spread-sheet logging
- Different or inconsistant logging is prone to erroneous results.
- Problems of interpretability of different columns or values might arise over time.

1.2 MLFlow - Manage the complete life-cycle of your M/L Models

What is MLFlow?



MLFlow helps in managing the complete life cycle of an M/L Project.

In a nutshell MLFlow is the combination of tools or techniques to give the AI or M/L Engineers, the possibility to not only manage their models, and their data, but to easily fine tune these models and play with different versions and are capable to be deployed at different places with different policies or parameters.

It helps track the results by not only automatically logging the metrics, the parameters and even the models but it gives you the flexibility to compare different models on the basis of different parameters or performance, not only in the tabular form but in the graphical format as well.

It can be easily integrated into any existing models easily and provides with a web interface, where the different M/L models can be comprehensively analyzed and

compared to other models. Recently, certain M/L Model integration code has also been included in mlflow, facilitating the auto-logging of a lot of regression or logistic models.

The four components offered by MLFlow:

- **MLflow Tracking** -- Record and query experiments i.e the code, data, parameters and the results
- **MLflow Projects** -- Packaging ML code, reproducible to share, transfer to diff stages
- **MLflow Models** -- Deploying models from ML Libraries in diverse environments
- **Model Registry** -- Central model repository to manage versioning, annotation and discovery

Note : MLFlow is library-agnostic

What is MLFlow Tracking?

MLFlow tracking includes but is not limited to following components:

- Organisation of Experiments
 - Logging the result of different runs
 - Tracking API
- Track of different M/L models :
 - Source Code source versioning
 - Trained Model
 - Data and its versioning
 - Associated Artifacts / Metadata
- Automatic logging :
 - Scikit-learn
 - Keras
 - Pytorch
 - XGBoost
 - LightBost
- Hyper-Parameters
- Evaluation/Performance Metrics
- Storage
 - Backend store : Filesystem / Sqlite db etc.
 - Artifacts storage : Amazon S3, , Azure Blob Storage, Google Cloud, FTP etc.

For comparing different models, it helps with:

- Tabular comparaison
- Graphical comparaison
- Effect of parameters on performance

What is MLFlow Projects?

MLFlow Projects is a convention for projects and data storage, so that the other data scientists can interact or use your code managed by MLFlow :

- The root directory is named after the project's name
- An entry point is defined with a **.sh** or **.py** file. These entry points do not have any parameters by default.
- While using API, the parameters can be passed to the entry point.
- Helps in building multi step flows:
 - The data science code be modularized and version so that different teams can work on different branches at the same time
 - Hyper-parameter tuning can be achieved through multiple runs locally or even in cloud environments for taking benefits of parallel computing, thus helping in finding the best performing models
 - Cross-validation and grid search can be used directly by providing different ranges of hyper-parameters, to find the best parameters for a specific kind of a data.

What is an MLFlow Model?

MLFlow model is a standard format for packaging M/L Models, so that these models can be used with a lot of other available tools and can be made available to public to be used in their projects, not having to be dependent upon the associated dependencies in their usage.

MLFlow Model keeps stores the models and the related environment information in the following directory structure:

```
model_alpha_beta_zetta2/  
├── MLmodel  
├── model.pk  
├── conda.yaml  
├── python_env.yaml  
└── requirements.txt
```

This model can be used by :

```
(.venv) % mlflow models serve model_alpha_beta_zetta3
```

Some interesting fields concerning the management of projects consist of signatures, inputs and the runtimes etc.

² The structure of the model as maintained by MLFlow

³ This command can is launcher in the virtual environment to serve a specific model

What is MLFlow Registry?

MLFlow Registry is a central repository for the storage of models, a set of APIs and UI to help manage the life cycle of your ML Models. It provides model annotation, their lineage (i.e. a kind of traceability of experiments to different runs with different parameters to different models), version control and their staging from development to deployment.

MLFlow Registry provides you with :

- Web UI : Registering / Using the model
- API Workflow : Adding/Registering, Fetching, Serving, Staging, Listing and Searching etc.

2. DEPLOYMENT

How to install MLFlow?

MLFlow can be installed in your python workflow by using :

```
(.venv) % pip install mlflow
```

Note :

- *As of this writing the latest version of MLFlow available for mac is 2.1.1*
- *Certain libraries might need a downgrading to function properly with MLFlow*

How to run the MLFlow Server?

MLFlow server instance can be initiated with the following command locally:

```
(.venv) % mlflow server -p 8000
```

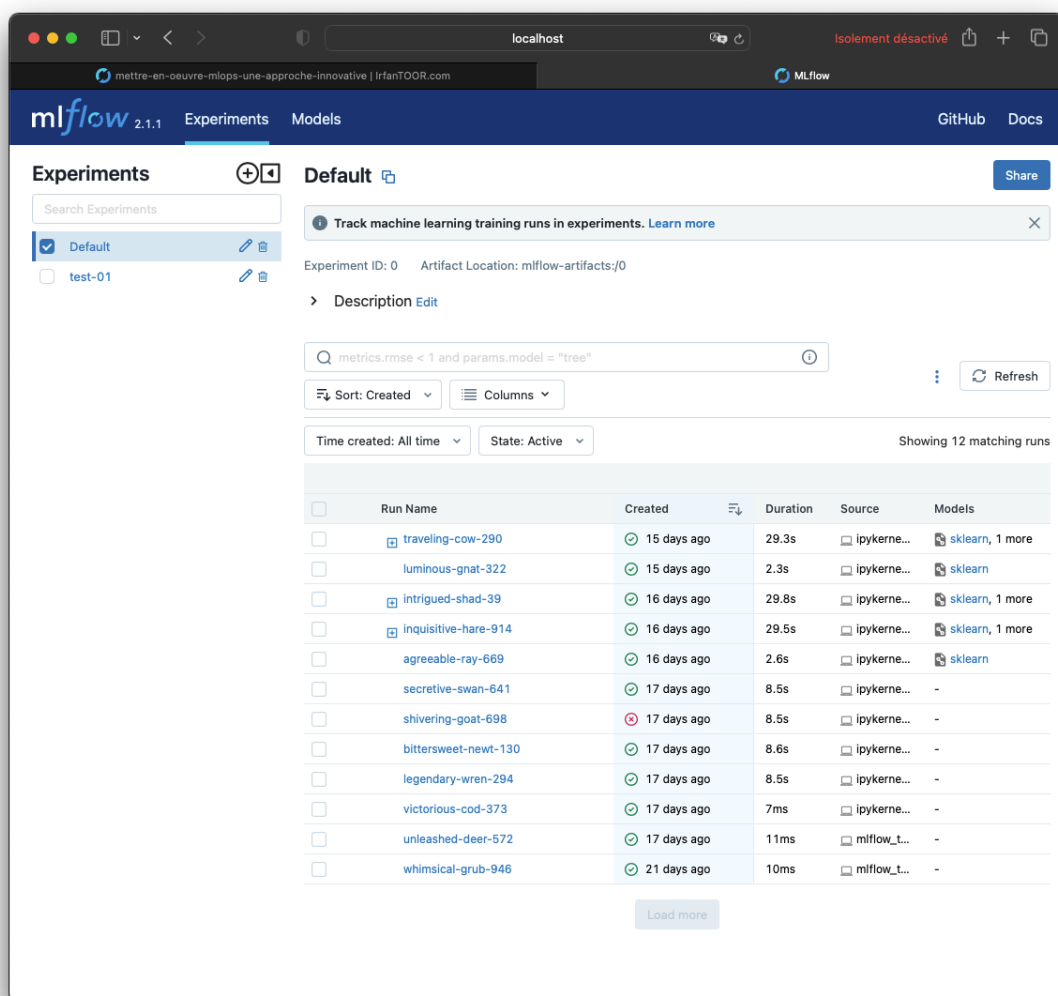
```
[2023-02-20 19:13:42 +0100] [58273] [INFO] Starting gunicorn 20.1.0
[2023-02-20 19:13:42 +0100] [58273] [INFO] Listening at: http://
127.0.0.1:8000 (58273)
[2023-02-20 19:13:42 +0100] [58273] [INFO] Using worker: sync
[2023-02-20 19:13:42 +0100] [58274] [INFO] Booting worker with pid:
58274
[2023-02-20 19:13:42 +0100] [58275] [INFO] Booting worker with pid:
58275
[2023-02-20 19:13:42 +0100] [58276] [INFO] Booting worker with pid:
58276
[2023-02-20 19:13:42 +0100] [58277] [INFO] Booting worker with pid:
58277
```

Note : -p helps defining the localhost port, since it uses 5000 by default, which might pose a problem of connectivity with a few versions of MacOS

Note that the server launches multiple listening threads, to load-balance any incoming requests. After launching this server you can access its Web UI, by accessing <http://127.0.0.1:8000>, using your navigator i.e. Safari or Firefox etc.

MLFlow Experiments :

MLFlow is capable of managing and grouping the results in different experiments, so that different projects or teams can be separated at experimentation levels easily.



MLFlow Model :

Each model can be tracked. It's description, parameters, metrics, tags and artifacts can be tracked. As an example the artifacts of the model can be seen in the next figures.

The screenshot shows the MLflow web interface for an experiment named 'delightful-colt-929'. The interface includes a sidebar with navigation links for 'Experiments' and 'Models'. The main content area displays the experiment's details, including the Run ID, Date, Source, Git Commit, User, Duration, Status, and Lifecycle Stage. A list of artifacts is shown on the left, including 'best_estimator', 'MLmodel', 'conda.yaml', 'model.pkl', 'python_env.yaml', 'requirements.txt', 'model', 'cv_results.csv', 'estimator.html', 'training_confusion_matrix.png', 'training_precision_recall_curve.png', and 'training_roc_curve.png'. The 'best_estimator' artifact is selected, showing its full path and a link to the 'delightful-colt-929, v1' model registry entry. Below this, the 'MLflow Model' section provides code snippets for making predictions using the logged model, including a 'Model schema' table and a 'Make Predictions' code block.

delightful-colt-929

Run ID: bde9be51f74f4ff5b3147807e6562e89 Date: 2023-02-04 23:30:35 Source: ipykernel_launcher.py

Git Commit: 70dacb35a14d370d306715fada9dba8c1631f771 User: it Duration: 16.9s

Status: FINISHED Lifecycle Stage: active

> Description Edit

> Parameters (11)

> Metrics (8)

> Tags (2)

▼ Artifacts

best_estimator

Full Path: mlflow-artifacts/682780464809541461/bde9be51f74f4ff5b3147807e6562e89/artifacts/best_estimator

delightful-colt-929, v1
Registered on 2023/02/09

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](#)

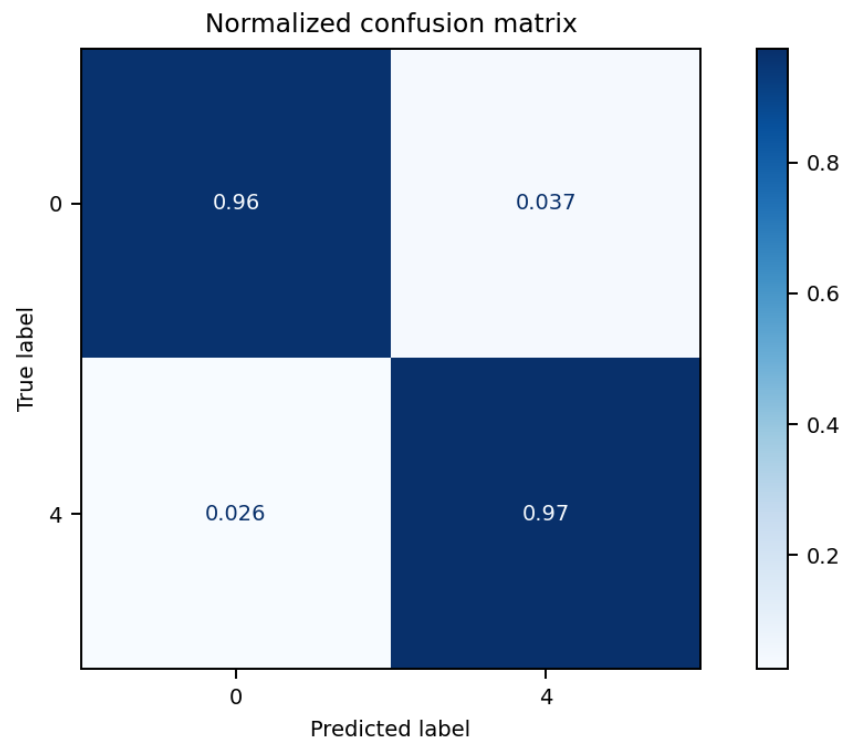
Name	Type
Inputs (2520)	
000	double
00pm	double

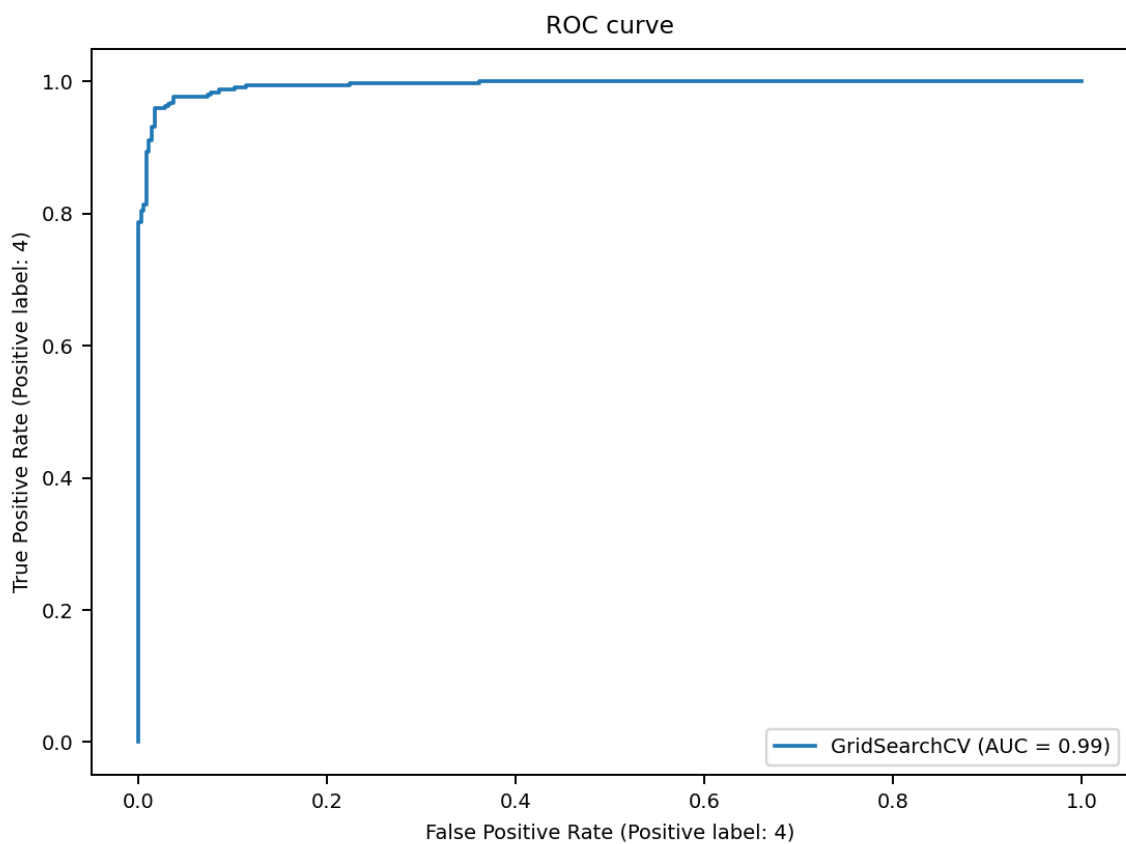
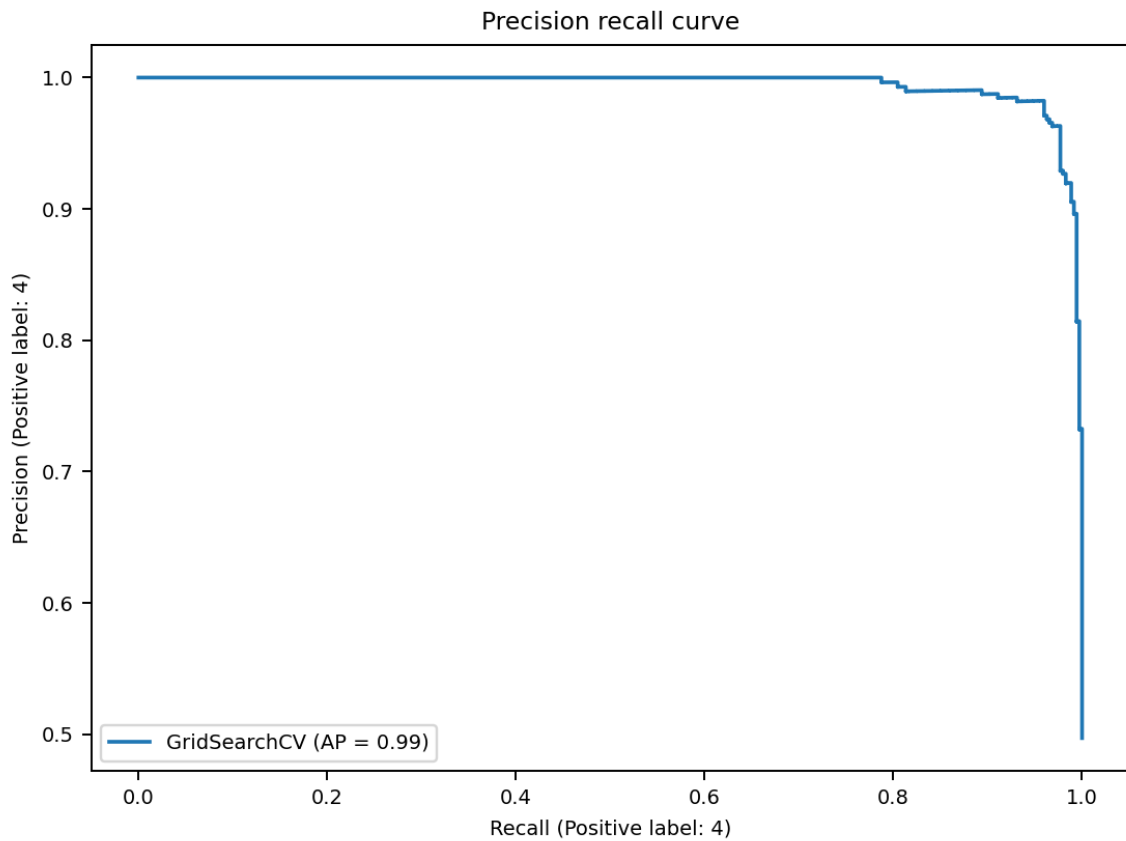
Make Predictions

Predict on a Spark DataFrame:

```
import mlflow
from pyspark.sql.functions import struct, col
logged_model = 'runs:/bde9be51f74f4ff5b3147807e6562e89/best_estimator'

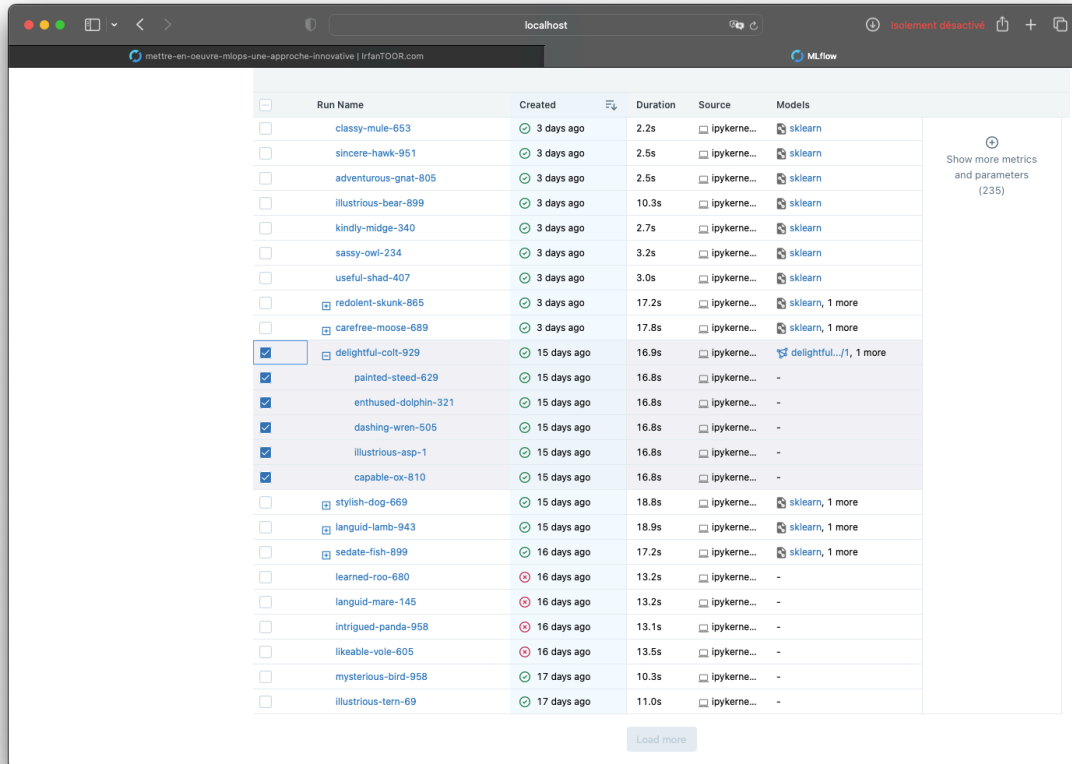
# Load model as a Spark UDF. Override result_type if the model does not return double values.
loaded_model = mlflow.pyfunc.spark_udf(spark, model_uri=logged_model, result_type='double')
```





MLFlow Grid-Search :

We can select all of the models created and grouped as sub models, while doing a grid search, and launch do a comparative analysis of their parameters as well as their results.

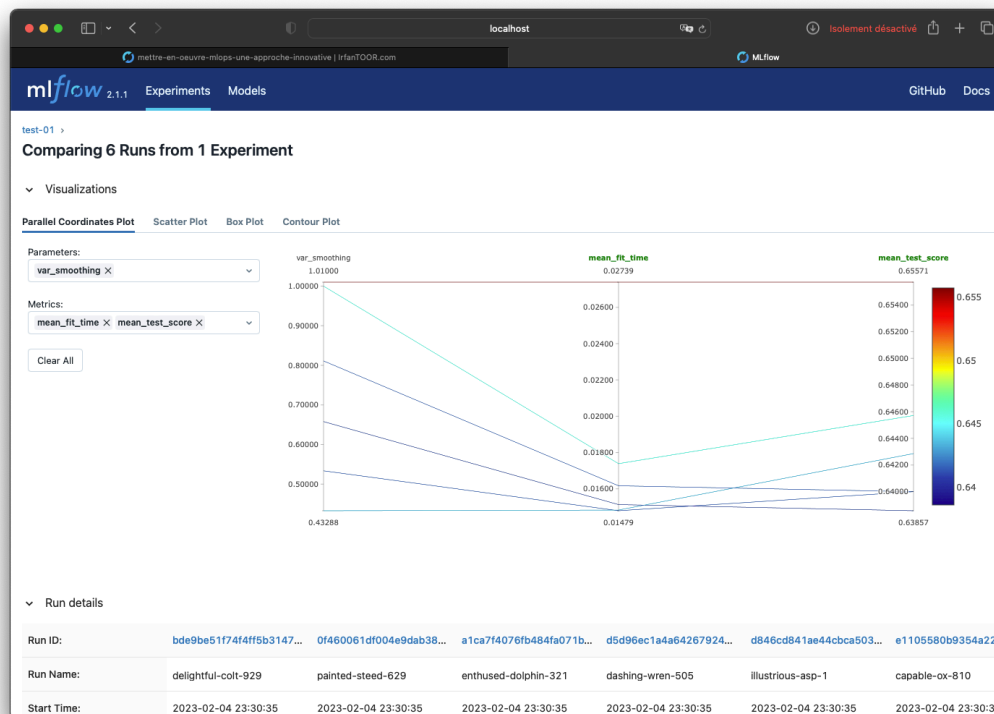
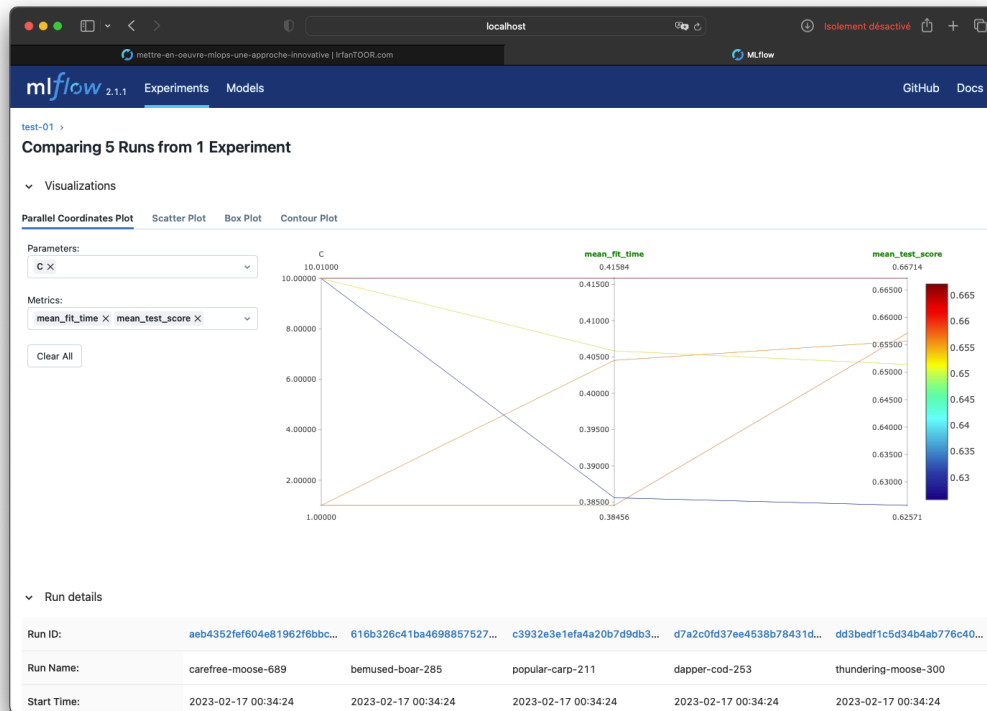


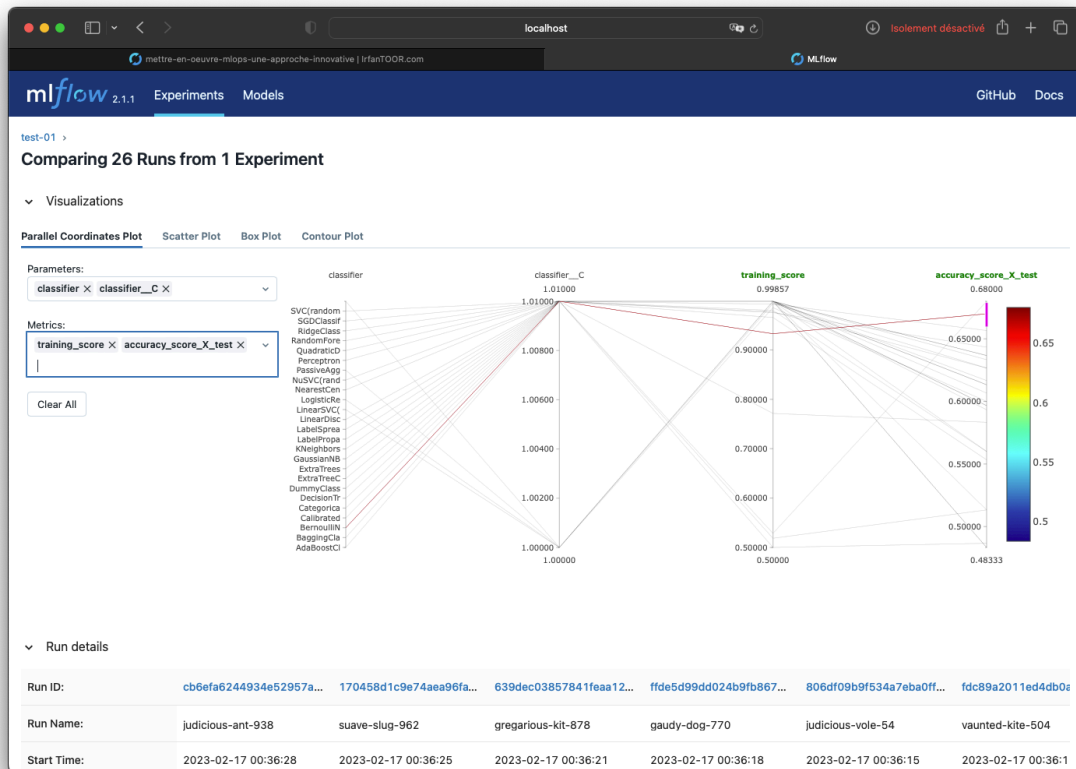
<input type="checkbox"/>	Run Name	Created	Duration	Source	Models
<input type="checkbox"/>	classy-mule-653	3 days ago	2.2s	ipykerne...	sklearn
<input type="checkbox"/>	sincere-hawk-951	3 days ago	2.5s	ipykerne...	sklearn
<input type="checkbox"/>	adventurous-gnat-805	3 days ago	2.5s	ipykerne...	sklearn
<input type="checkbox"/>	illustrious-bear-899	3 days ago	10.3s	ipykerne...	sklearn
<input type="checkbox"/>	kindly-midge-340	3 days ago	2.7s	ipykerne...	sklearn
<input type="checkbox"/>	sassy-owl-234	3 days ago	3.2s	ipykerne...	sklearn
<input type="checkbox"/>	useful-shad-407	3 days ago	3.0s	ipykerne...	sklearn
<input type="checkbox"/>	redolent-skunk-865	3 days ago	17.2s	ipykerne...	sklearn, 1 more
<input type="checkbox"/>	carefree-moose-689	3 days ago	17.8s	ipykerne...	sklearn, 1 more
<input checked="" type="checkbox"/>	delightful-coit-929	15 days ago	16.9s	ipykerne...	delightful.../1, 1 more
<input checked="" type="checkbox"/>	painted-steed-629	15 days ago	16.8s	ipykerne...	-
<input checked="" type="checkbox"/>	enthused-dolphin-321	15 days ago	16.8s	ipykerne...	-
<input checked="" type="checkbox"/>	dashing-wren-505	15 days ago	16.8s	ipykerne...	-
<input checked="" type="checkbox"/>	illustrious-asp-1	15 days ago	16.8s	ipykerne...	-
<input checked="" type="checkbox"/>	capable-ox-810	15 days ago	16.8s	ipykerne...	-
<input type="checkbox"/>	stylish-dog-669	15 days ago	18.8s	ipykerne...	sklearn, 1 more
<input type="checkbox"/>	languid-lamb-943	15 days ago	18.9s	ipykerne...	sklearn, 1 more
<input type="checkbox"/>	sedate-fish-899	16 days ago	17.2s	ipykerne...	sklearn, 1 more
<input type="checkbox"/>	learned-roo-680	16 days ago	13.2s	ipykerne...	-
<input type="checkbox"/>	languid-mare-145	16 days ago	13.2s	ipykerne...	-
<input type="checkbox"/>	intrigued-panda-958	16 days ago	13.1s	ipykerne...	-
<input type="checkbox"/>	likeable-vole-605	16 days ago	13.5s	ipykerne...	-
<input type="checkbox"/>	mysterious-bird-958	17 days ago	10.3s	ipykerne...	-
<input type="checkbox"/>	illustrious-tern-69	17 days ago	11.0s	ipykerne...	-

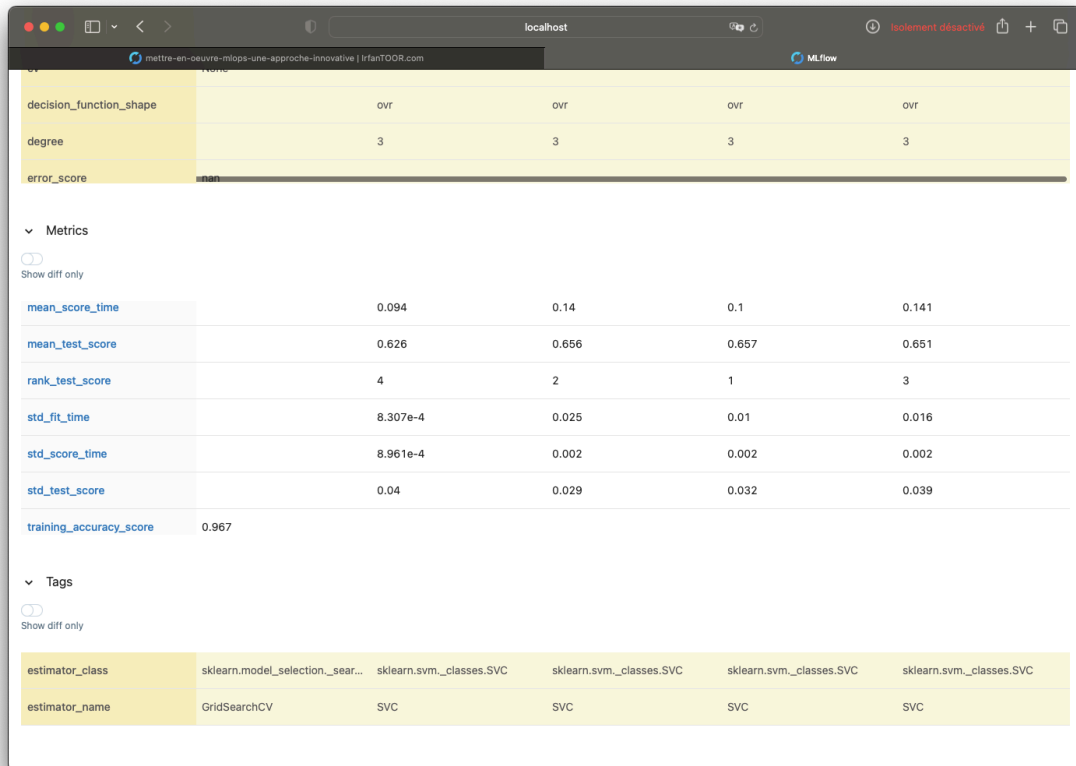
Load more

MLFlow Comparison :

Different runs can be compared graphically, their metrics and run details can also be compared in tabular form.







The screenshot shows the MLflow UI interface. At the top, there's a browser window with the URL 'localhost' and a tab titled 'mettre-en-oeuvre-mlops-une-approche-innovative | IrfanTOOR.com'. The MLflow logo is visible in the top right corner. The main content area displays a table of metrics and tags for a specific run. The 'Metrics' section is expanded, showing a table with columns for various metrics and their values. The 'Tags' section is also expanded, showing a table with columns for estimator_class and estimator_name. The browser's address bar shows 'localhost' and the page title is 'mettre-en-oeuvre-mlops-une-approche-innovative | IrfanTOOR.com'.

metric	value	metric	value	metric	value	metric	value
decision_function_shape	ovr	decision_function_shape	ovr	decision_function_shape	ovr	decision_function_shape	ovr
degree	3	degree	3	degree	3	degree	3
error_score	nan	error_score	nan	error_score	nan	error_score	nan

▼ Metrics

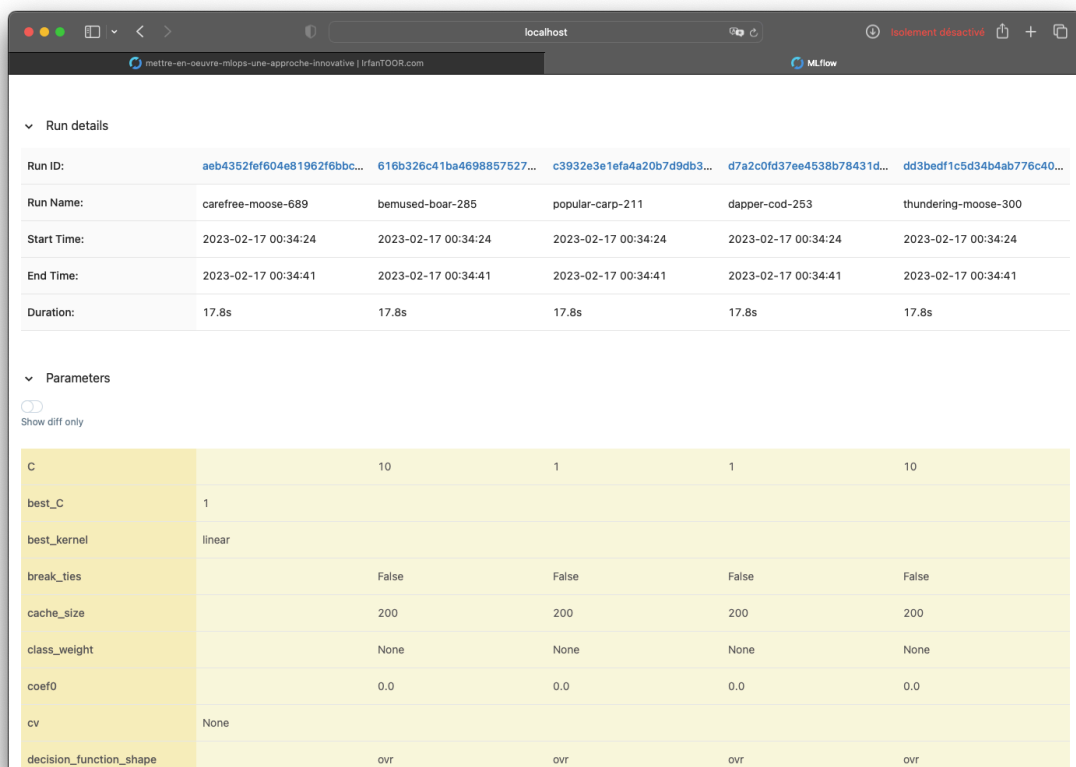
Show diff only

metric	value	metric	value	metric	value	metric	value
mean_score_time	0.094	mean_score_time	0.14	mean_score_time	0.1	mean_score_time	0.141
mean_test_score	0.626	mean_test_score	0.656	mean_test_score	0.657	mean_test_score	0.651
rank_test_score	4	rank_test_score	2	rank_test_score	1	rank_test_score	3
std_fit_time	8.307e-4	std_fit_time	0.025	std_fit_time	0.01	std_fit_time	0.016
std_score_time	8.961e-4	std_score_time	0.002	std_score_time	0.002	std_score_time	0.002
std_test_score	0.04	std_test_score	0.029	std_test_score	0.032	std_test_score	0.039
training_accuracy_score	0.967	training_accuracy_score		training_accuracy_score		training_accuracy_score	

▼ Tags

Show diff only

tag	value	tag	value	tag	value	tag	value
estimator_class	sklearn.model_selection_scar...	estimator_class	sklearn.svm_classes.SVC	estimator_class	sklearn.svm_classes.SVC	estimator_class	sklearn.svm_classes.SVC
estimator_name	GridSearchCV	estimator_name	SVC	estimator_name	SVC	estimator_name	SVC



The screenshot shows the MLflow UI interface. At the top, there's a browser window with the URL 'localhost' and a tab titled 'mettre-en-oeuvre-mlops-une-approche-innovative | IrfanTOOR.com'. The MLflow logo is visible in the top right corner. The main content area displays a table of run details and parameters for a specific run. The 'Run details' section is expanded, showing a table with columns for Run ID, Run Name, Start Time, End Time, and Duration. The 'Parameters' section is also expanded, showing a table with columns for various parameters and their values. The browser's address bar shows 'localhost' and the page title is 'mettre-en-oeuvre-mlops-une-approche-innovative | IrfanTOOR.com'.

Run ID:	Run Name:	Start Time:	End Time:	Duration:
aeb4352fef604e81962f6bbc...	carefree-moose-689	2023-02-17 00:34:24	2023-02-17 00:34:41	17.8s
616b326c41ba4698857527...	bemused-boar-285	2023-02-17 00:34:24	2023-02-17 00:34:41	17.8s
c3932e3e1efa4a20b7d9db3...	popular-carp-211	2023-02-17 00:34:24	2023-02-17 00:34:41	17.8s
d7a2c0fd37ee4538b78431d...	dapper-cod-253	2023-02-17 00:34:24	2023-02-17 00:34:41	17.8s
dd3bedf1c5d34b4ab776c40...	thundering-moose-300	2023-02-17 00:34:24	2023-02-17 00:34:41	17.8s

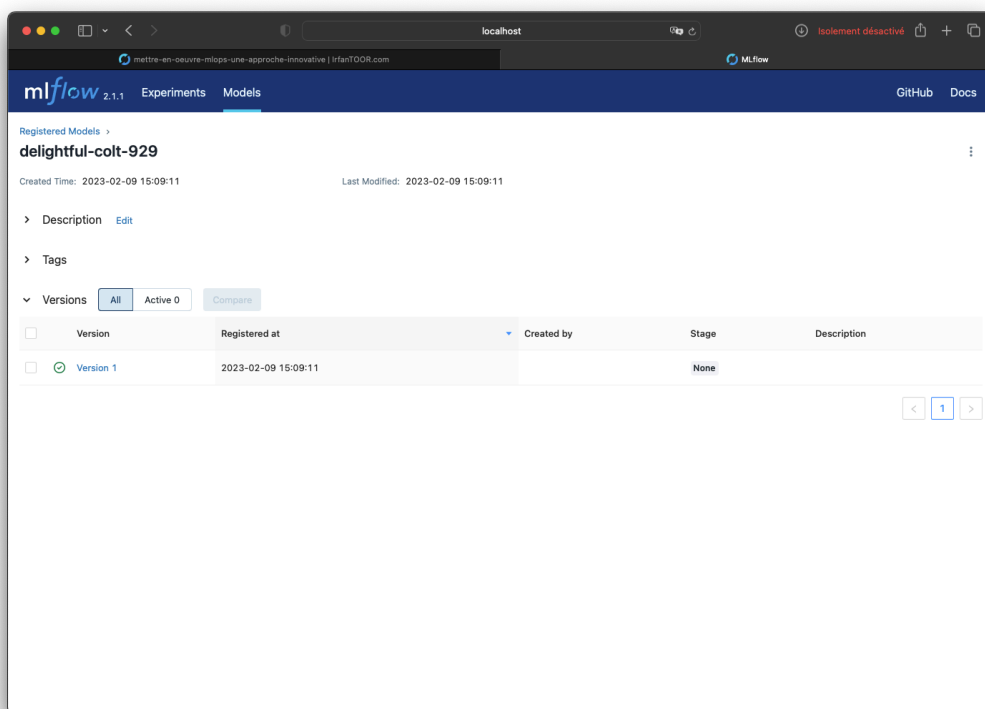
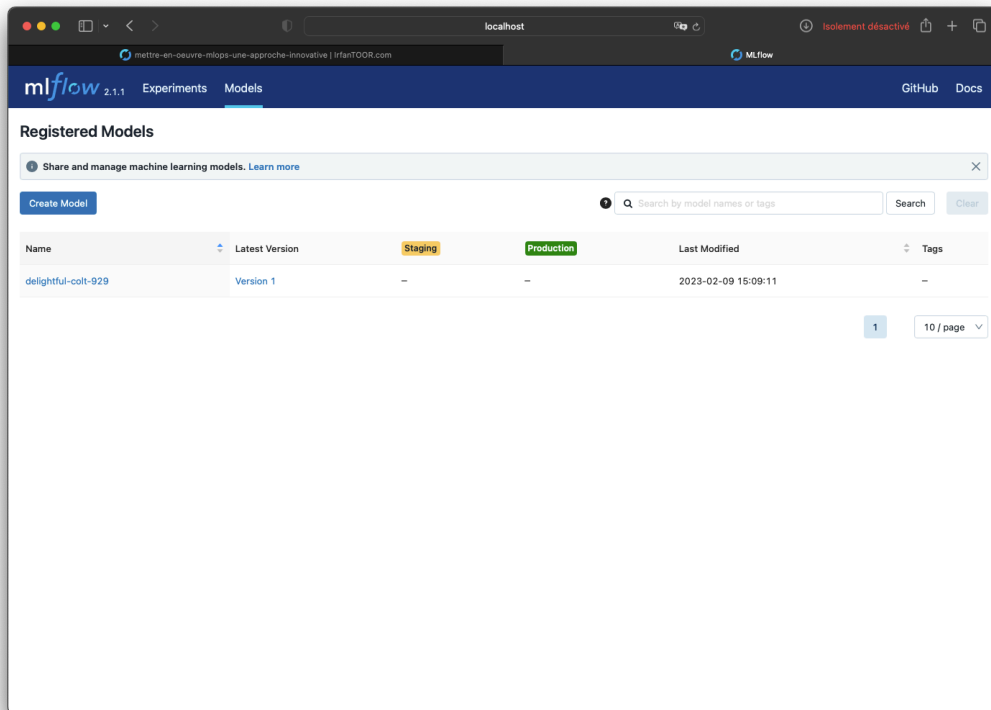
▼ Parameters

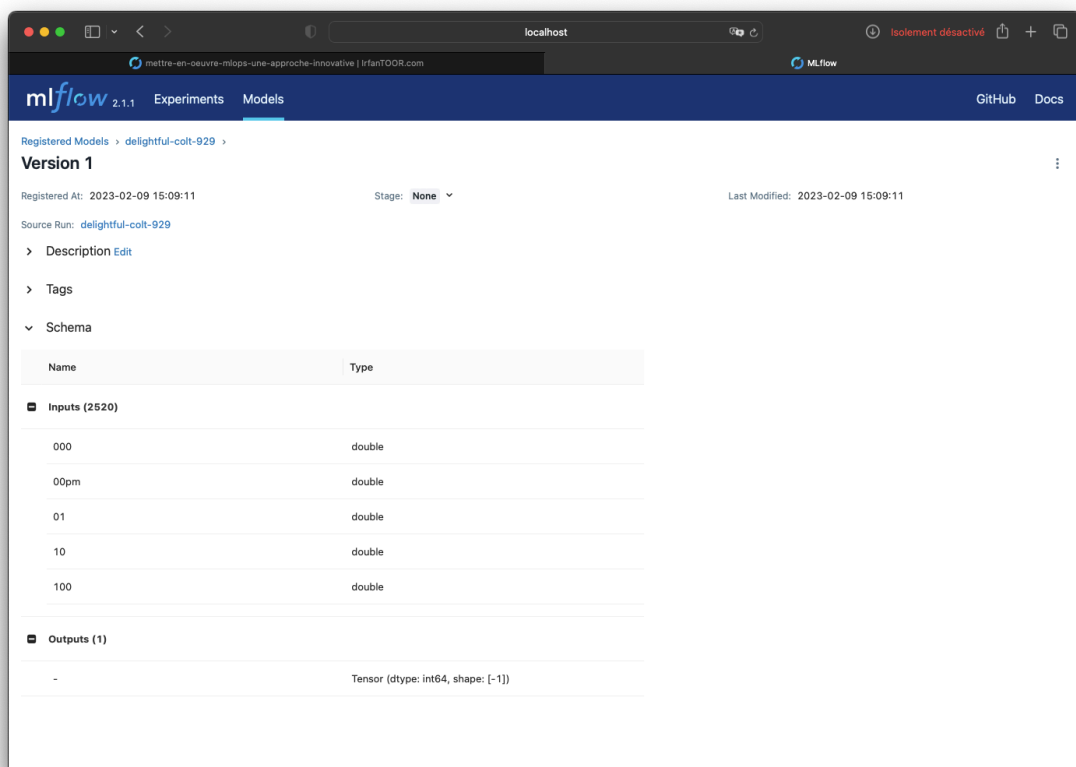
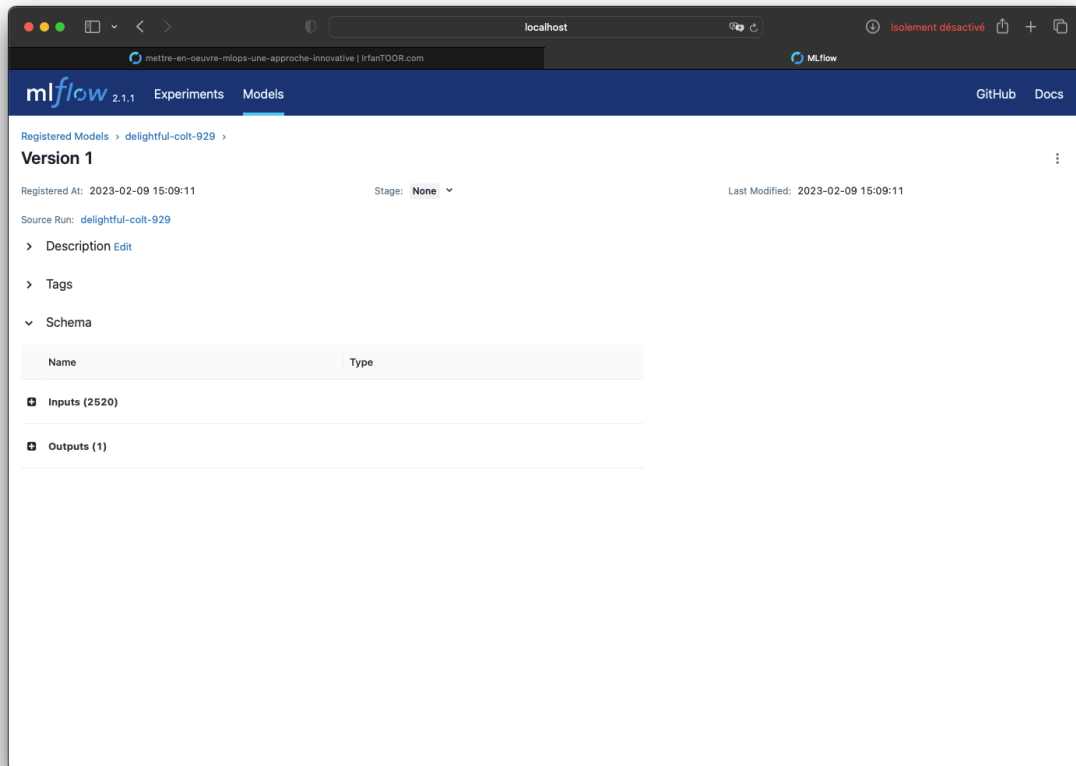
Show diff only

parameter	value	parameter	value	parameter	value	parameter	value
C	10	C	1	C	1	C	10
best_C	1	best_C		best_C		best_C	
best_kernel	linear	best_kernel		best_kernel		best_kernel	
break_ties	False	break_ties	False	break_ties	False	break_ties	False
cache_size	200	cache_size	200	cache_size	200	cache_size	200
class_weight	None	class_weight	None	class_weight	None	class_weight	None
coef0	0.0	coef0	0.0	coef0	0.0	coef0	0.0
cv	None	cv		cv		cv	
decision_function_shape	ovr	decision_function_shape	ovr	decision_function_shape	ovr	decision_function_shape	ovr

MLFlow Models

The models can be registered so that these can be easily staged for production.





Model lineage helps tracking the model and their respective parameters and results etc.

The screenshot shows the MLflow Experiments page for an experiment named "delightful-colt-929". The page displays various metadata including Run ID, Date, Source, Git Commit, User, Status, and Lifecycle Stage. A sidebar on the left lists artifacts such as MLmodel, conda.yaml, estimator.pkl, and training files. The main content area shows the MLflow Model details, including a description and a "Make Predictions" section with a code snippet for loading the model as a Spark UDF.

delightful-colt-929

Run ID: bde9be51f74f4ff5b3147807e6562e89 Date: 2023-02-04 23:30:35 Source: ipykernel_launcher.py

Git Commit: 70dabc35a14d370d306715fada9dba8c1631f771 User: it Duration: 16.9s

Status: FINISHED Lifecycle Stage: active

> Description Edit

> Parameters (11)

> Metrics (8)

> Tags (2)

▼ Artifacts

- best_estimator
 - MLmodel
 - conda.yaml
 - estimator.pkl
 - python_env.yaml
 - requirements.txt
- model
 - cv_results.csv
 - estimator.html
 - training_confusion_matrix.png
 - training_precision_recall_curve.png
 - training_roc_curve.png

Full Path: mlflow-artifacts/682780464809541461/bde9be51f74f4ff5b3147807e6562e89/artifacts/best_estimator

delightful-colt-929, v1
Registered on 2023/02/09

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 (2520)	
000	double
00pm	double

Make Predictions

Predict on a Spark DataFrame:

```
import mlflow
from pyspark.sql.functions import struct, col
logged_model = 'runs:/bde9be51f74f4ff5b3147807e6562e89/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=logged_model, result_type='double')
```

The screenshot shows the MLflow Experiments page for the same experiment, but with the "Parameters" section expanded. It displays a table of 11 parameters, including best_var_smoothing, cv, error_score, estimator, n_jobs, param_grid, pre_dispatch, refit, return_train_score, and scoring.

delightful-colt-929

Run ID: bde9be51f74f4ff5b3147807e6562e89 Date: 2023-02-04 23:30:35 Source: ipykernel_launcher.py

Git Commit: 70dabc35a14d370d306715fada9dba8c1631f771 User: it Duration: 16.9s

Status: FINISHED Lifecycle Stage: active

> Description Edit

▼ Parameters (11)

Name	Value
best_var_smoothing	1.0
cv	None
error_score	nan
estimator	GaussianNB()
n_jobs	None
param_grid	{'var_smoothing': array([1.00000000e+00, 8.11130831e-01, 6.57933225e-01, 5.33669923e-01, 4.32876128e-01, 3.51119173e-01, 2.84803587e-01, 2.31012970e-01, 1.87381742e-01, 1.51991108e-01, 1.23284674e-01, 1.00000000e-01, 8.11130831e-02, 6.57933225e-02, 5.33669923e-02, 4.32876128e-02, 3.51119173e-02, 2.84803587e-02, 2.31012970e-02, 1.87381742e-02, 1.51991108e-02, 1.23284674e-02, 1.00000000e-02, 8.11130831e-03, 6.57933225e-03, 5.33669923e-03, 4.32876128e-03, 3.51119173e-03, 2.84803587e-03, 2.31012970e-03, 1.87381742e-03, 1.51991108e-03, 1.23284674e-03, 1.00000000e-03, 8.11130831e-04, 6.57933225e-04, 5.33669923e-04, 4.32876128e-04, 3.51119173e-04, 2.84803587e-04, 2.31012970e-04, 1.87381742e-04, 1.51991108e-04, 1.23284674e-04, 1.00000000e-04, 8.11130831e-05, 6.57933225e-05, 5.33669923e-05, 4.32876128e-05, 3.51119173e-05, 2.84803587e-05, 2.31012970e-05, 1.87381742e-05, 1.51991108e-05, 1.23284674e-05, 1.00000000e-05, 8.11130831e-06, 6.57933225e-06, 5.33669923e-06, 4.32876128e-06, 3.51119173e-06, 2.84803587e-06, 2.31012970e-06, 1.87381742e-06, 1.51991108e-06, 1.23284674e-06, 1.00000000e-06, 8.11130831e-07, 6.57933225e-07, 5.33669923e-07, 4.32876128e-07, 3.51119173e-07, 2.84803587e-07, 2.31012970e-07, 1.87381742e-07, 1.51991108e-07, 1.23284674e-07, 1.00000000e-07, 8.11130831e-08, 6.57933225e-08, 5.33669923e-08, 4.32876128e-08, 3.51119173e-08, 2.84803587e-08, 2.31012970e-08, 1.87381742e-08, 1.51991108e-08, 1.23284674e-08, 1.00000000e-08, 8.11130831e-09, 6.57933225e-09, 5.33669923e-09, 4.32876128e-09, 3.51119173e-09, 2.84803587e-09, 2.31012970e-09, 1.87381742e-09, 1.51991108e-09, 1.23284674e-09, 1.00000000e-09, 8.11130831e-10, 6.57933225e-10, 5.33669923e-10, 4.32876128e-10, 3.51119173e-10, 2.84803587e-10, 2.31012970e-10, 1.87381742e-10, 1.51991108e-10, 1.23284674e-10, 1.00000000e-10, 8.11130831e-11, 6.57933225e-11, 5.33669923e-11, 4.32876128e-11, 3.51119173e-11, 2.84803587e-11, 2.31012970e-11, 1.87381742e-11, 1.51991108e-11, 1.23284674e-11, 1.00000000e-11, 8.11130831e-12, 6.57933225e-12, 5.33669923e-12, 4.32876128e-12, 3.51119173e-12, 2.84803587e-12, 2.31012970e-12, 1.87381742e-12, 1.51991108e-12, 1.23284674e-12, 1.00000000e-12, 8.11130831e-13, 6.57933225e-13, 5.33669923e-13, 4.32876128e-13, 3.51119173e-13, 2.84803587e-13, 2.31012970e-13, 1.87381742e-13, 1.51991108e-13, 1.23284674e-13, 1.00000000e-13, 8.11130831e-14, 6.57933225e-14, 5.33669923e-14, 4.32876128e-14, 3.51119173e-14, 2.84803587e-14, 2.31012970e-14, 1.87381742e-14, 1.51991108e-14, 1.23284674e-14, 1.00000000e-14, 8.11130831e-15, 6.57933225e-15, 5.33669923e-15, 4.32876128e-15, 3.51119173e-15, 2.84803587e-15, 2.31012970e-15, 1.87381742e-15, 1.51991108e-15, 1.23284674e-15, 1.00000000e-15, 8.11130831e-16, 6.57933225e-16, 5.33669923e-16, 4.32876128e-16, 3.51119173e-16, 2.84803587e-16, 2.31012970e-16, 1.87381742e-16, 1.51991108e-16, 1.23284674e-16, 1.00000000e-16, 8.11130831e-17, 6.57933225e-17, 5.33669923e-17, 4.32876128e-17, 3.51119173e-17, 2.84803587e-17, 2.31012970e-17, 1.87381742e-17, 1.51991108e-17, 1.23284674e-17, 1.00000000e-17, 8.11130831e-18, 6.57933225e-18, 5.33669923e-18, 4.32876128e-18, 3.51119173e-18, 2.84803587e-18, 2.31012970e-18, 1.87381742e-18, 1.51991108e-18, 1.23284674e-18, 1.00000000e-18, 8.11130831e-19, 6.57933225e-19, 5.33669923e-19, 4.32876128e-19, 3.51119173e-19, 2.84803587e-19, 2.31012970e-19, 1.87381742e-19, 1.51991108e-19, 1.23284674e-19, 1.00000000e-19, 8.11130831e-20, 6.57933225e-20, 5.33669923e-20, 4.32876128e-20, 3.51119173e-20, 2.84803587e-20, 2.31012970e-20, 1.87381742e-20, 1.51991108e-20, 1.23284674e-20, 1.00000000e-20, 8.11130831e-21, 6.57933225e-21, 5.33669923e-21, 4.32876128e-21, 3.51119173e-21, 2.84803587e-21, 2.31012970e-21, 1.87381742e-21, 1.51991108e-21, 1.23284674e-21, 1.00000000e-21, 8.11130831e-22, 6.57933225e-22, 5.33669923e-22, 4.32876128e-22, 3.51119173e-22, 2.84803587e-22, 2.31012970e-22, 1.87381742e-22, 1.51991108e-22, 1.23284674e-22, 1.00000000e-22, 8.11130831e-23, 6.57933225e-23, 5.33669923e-23, 4.32876128e-23, 3.51119173e-23, 2.84803587e-23, 2.31012970e-23, 1.87381742e-23, 1.51991108e-23, 1.23284674e-23, 1.00000000e-23, 8.11130831e-24, 6.57933225e-24, 5.33669923e-24, 4.32876128e-24, 3.51119173e-24, 2.84803587e-24, 2.31012970e-24, 1.87381742e-24, 1.51991108e-24, 1.23284674e-24, 1.00000000e-24, 8.11130831e-25, 6.57933225e-25, 5.33669923e-25, 4.32876128e-25, 3.51119173e-25, 2.84803587e-25, 2.31012970e-25, 1.87381742e-25, 1.51991108e-25, 1.23284674e-25, 1.00000000e-25, 8.11130831e-26, 6.57933225e-26, 5.33669923e-26, 4.32876128e-26, 3.51119173e-26, 2.84803587e-26, 2.31012970e-26, 1.87381742e-26, 1.51991108e-26, 1.23284674e-26, 1.00000000e-26, 8.11130831e-27, 6.57933225e-27, 5.33669923e-27, 4.32876128e-27, 3.51119173e-27, 2.84803587e-27, 2.31012970e-27, 1.87381742e-27, 1.51991108e-27, 1.23284674e-27, 1.00000000e-27, 8.11130831e-28, 6.57933225e-28, 5.33669923e-28, 4.32876128e-28, 3.51119173e-28, 2.84803587e-28, 2.31012970e-28, 1.87381742e-28, 1.51991108e-28, 1.23284674e-28, 1.00000000e-28, 8.11130831e-29, 6.57933225e-29, 5.33669923e-29, 4.32876128e-29, 3.51119173e-29, 2.84803587e-29, 2.31012970e-29, 1.87381742e-29, 1.51991108e-29, 1.23284674e-29, 1.00000000e-29, 8.11130831e-30, 6.57933225e-30, 5.33669923e-30, 4.32876128e-30, 3.51119173e-30, 2.84803587e-30, 2.31012970e-30, 1.87381742e-30, 1.51991108e-30, 1.23284674e-30, 1.00000000e-30, 8.11130831e-31, 6.57933225e-31, 5.33669923e-31, 4.32876128e-31, 3.51119173e-31, 2.84803587e-31, 2.31012970e-31, 1.87381742e-31, 1.51991108e-31, 1.23284674e-31, 1.00000000e-31, 8.11130831e-32, 6.57933225e-32, 5.33669923e-32, 4.32876128e-32, 3.51119173e-32, 2.84803587e-32, 2.31012970e-32, 1.87381742e-32, 1.51991108e-32, 1.23284674e-32, 1.00000000e-32, 8.11130831e-33, 6.57933225e-33, 5.33669923e-33, 4.32876128e-33, 3.51119173e-33, 2.84803587e-33, 2.31012970e-33, 1.87381742e-33, 1.51991108e-33, 1.23284674e-33, 1.00000000e-33, 8.11130831e-34, 6.57933225e-34, 5.33669923e-34, 4.32876128e-34, 3.51119173e-34, 2.84803587e-34, 2.31012970e-34, 1.87381742e-34, 1.51991108e-34, 1.23284674e-34, 1.00000000e-34, 8.11130831e-35, 6.57933225e-35, 5.33669923e-35, 4.32876128e-35, 3.51119173e-35, 2.84803587e-35, 2.31012970e-35, 1.87381742e-35, 1.51991108e-35, 1.23284674e-35, 1.00000000e-35, 8.11130831e-36, 6.57933225e-36, 5.33669923e-36, 4.32876128e-36, 3.51119173e-36, 2.84803587e-36, 2.31012970e-36, 1.87381742e-36, 1.51991108e-36, 1.23284674e-36, 1.00000000e-36, 8.11130831e-37, 6.57933225e-37, 5.33669923e-37, 4.32876128e-37, 3.51119173e-37, 2.84803587e-37, 2.31012970e-37, 1.87381742e-37, 1.51991108e-37, 1.23284674e-37, 1.00000000e-37, 8.11130831e-38, 6.57933225e-38, 5.33669923e-38, 4.32876128e-38, 3.51119173e-38, 2.84803587e-38, 2.31012970e-38, 1.87381742e-38, 1.51991108e-38, 1.23284674e-38, 1.00000000e-38, 8.11130831e-39, 6.57933225e-39, 5.33669923e-39, 4.32876128e-39, 3.51119173e-39, 2.84803587e-39, 2.31012970e-39, 1.87381742e-39, 1.51991108e-39, 1.23284674e-39, 1.00000000e-39, 8.11130831e-40, 6.57933225e-40, 5.33669923e-40, 4.32876128e-40, 3.51119173e-40, 2.84803587e-40, 2.31012970e-40, 1.87381742e-40, 1.51991108e-40, 1.23284674e-40, 1.00000000e-40, 8.11130831e-41, 6.57933225e-41, 5.33669923e-41, 4.32876128e-41, 3.51119173e-41, 2.84803587e-41, 2.31012970e-41, 1.87381742e-41, 1.51991108e-41, 1.23284674e-41, 1.00000000e-41, 8.11130831e-42, 6.57933225e-42, 5.33669923e-42, 4.32876128e-42, 3.51119173e-42, 2.84803587e-42, 2.31012970e-42, 1.87381742e-42, 1.51991108e-42, 1.23284674e-42, 1.00000000e-42, 8.11130831e-43, 6.57933225e-43, 5.33669923e-43, 4.32876128e-43, 3.51119173e-43, 2.84803587e-43, 2.31012970e-43, 1.87381742e-43, 1.51991108e-43, 1.23284674e-43, 1.00000000e-43, 8.11130831e-44, 6.57933225e-44, 5.33669923e-44, 4.32876128e-44, 3.51119173e-44, 2.84803587e-44, 2.31012970e-44, 1.87381742e-44, 1.51991108e-44, 1.23284674e-44, 1.00000000e-44, 8.11130831e-45, 6.57933225e-45, 5.33669923e-45, 4.32876128e-45, 3.51119173e-45, 2.84803587e-45, 2.31012970e-45, 1.87381742e-45, 1.51991108e-45, 1.23284674e-45, 1.00000000e-45, 8.11130831e-46, 6.57933225e-46, 5.33669923e-46, 4.32876128e-46, 3.51119173e-46, 2.84803587e-46, 2.31012970e-46, 1.87381742e-46, 1.51991108e-46, 1.23284674e-46, 1.00000000e-46, 8.11130831e-47, 6.57933225e-47, 5.33669923e-47, 4.32876128e-47, 3.51119173e-47, 2.84803587e-47, 2.31012970e-47, 1.87381742e-47, 1.51991108e-47, 1.23284674e-47, 1.00000000e-47, 8.11130831e-48, 6.57933225e-48, 5.33669923e-48, 4.32876128e-48, 3.51119173e-48, 2.84803587e-48, 2.31012970e-48, 1.87381742e-48, 1.51991108e-48, 1.23284674e-48, 1.00000000e-48, 8.11130831e-49, 6.57933225e-49, 5.33669923e-49, 4.32876128e-49, 3.51119173e-49, 2.84803587e-49, 2.31012970e-49, 1.87381742e-49, 1.51991108e-49, 1.23284674e-49, 1.00000000e-49, 8.11130831e-50, 6.57933225e-50, 5.33669923e-50, 4.32876128e-50, 3.51119173e-50, 2.84803587e-50, 2.31012970e-50, 1.87381742e-50, 1.51991108e-50, 1.23284674e-50, 1.00000000e-50, 8.11130831e-51, 6.57933225e-51, 5.33669923e-51, 4.32876128e-51, 3.51119173e-51, 2.84803587e-51, 2.31012970e-51, 1.87381742e-51, 1.51991108e-51, 1.23284674e-51, 1.00000000e-51, 8.11130831e-52, 6.57933225e-52, 5.33669923e-52, 4.32876128e-52, 3.51119173e-52, 2.84803587e-52, 2.31012970e-52, 1.87381742e-52, 1.51991108e-52, 1.23284674e-52, 1.00000000e-52, 8.11130831e-53, 6.57933225e-53, 5.33669923e-53, 4.32876128e-53, 3.51119173e-53, 2.84803587e-53, 2.31012970e-53, 1.87381742e-53, 1.51991108e-53, 1.23284674e-53, 1.00000000e-53, 8.11130831e-54, 6.57933225e-54, 5.33669923e-54, 4.32876128e-54, 3.51119173e-54, 2.84803587e-54, 2.31012970e-54, 1.87381742e-54, 1.51991108e-54, 1.23284674e-54, 1.00000000e-54, 8.11130831e-55, 6.57933225e-55, 5.33669923e-55, 4.32876128e-55, 3.51119173e-55, 2.84803587e-55, 2.31012970e-55, 1.87381742e-55, 1.51991108e-55, 1.23284674e-55, 1.00000000e-55, 8.11130831e-56, 6.57933225e-56, 5.33669923e-56, 4.32876128e-56, 3.51119173e-56, 2.84803587e-56, 2.31012970e-56, 1.87381742e-56, 1.51991108e-56, 1.23284674e-56, 1.00000000e-56, 8.11130831e-57, 6.57933225e-57, 5.33669923e-57, 4.32876128e-57, 3.51119173e-57, 2.84803587e-57, 2.31012970e-57, 1.87381742e-57, 1.51991108e-57, 1.23284674e-57, 1.00000000e-57, 8.11130831e-58, 6.57933225e-58, 5.33669923e-58, 4.32876128e-58, 3.51119173e-58, 2.84803587e-58, 2.31012970e-58, 1.87381742e-58, 1.51991108e-58, 1.23284674e-58, 1.00000000e-58, 8.11130831e-59, 6.57933225e-59, 5.33669923e-59, 4.32876128e-59, 3.51119173e-59, 2.84803587e-59, 2.31012970e-59, 1.87381742e-59, 1.51991108e-59, 1.23284674e-59, 1.00000000e-59, 8.11130831e-60, 6.57933225e-60, 5.33669923e-60, 4.32876128e-60, 3.51119173e-60, 2.84803587e-60, 2.31012970e-60, 1.87381742e-60, 1.51991108e-60, 1.23284674e-60, 1.00000000e-60, 8.11130831e-61, 6.57933225e-61, 5.33669923e-61, 4.32876128e-61, 3.51119173e-61, 2.84803587e-61, 2.31012970e-61, 1.87381742e-61, 1.51991108e-61, 1.23284674e-61, 1.00000000e-61, 8.11130831e-62, 6.57933225e-62, 5.33669923e-62, 4.32876128e-62, 3.51119173e-62, 2.84803587e-62, 2.31012970e-62, 1.87381742e-62, 1.51991108e-62, 1.23284674e-62, 1.00000000e-62, 8.11130831e-63, 6.57933225e-63, 5.33669923e-63, 4.32876128e-63, 3.51119173e-63, 2.84803587e-63, 2.31012970e-63, 1.87381742e-63,

The screenshot shows the MLflow Experiments page for an experiment named 'delightful-colt-929'. The page displays various metadata and metrics for the experiment.

Experiment Details:

- Run ID: bde9be51f74f4ff5b3147807e6562e89
- Date: 2023-02-04 23:30:35
- Source: ipykernel_launcher.py
- Git Commit: 70dabc35a14d370d306715fada9dba8c1631f771
- User: it
- Duration: 16.9s
- Status: FINISHED
- Lifecycle Stage: active

Metrics (8):

Name	Value
best_cv_score	0.646
training_accuracy_score	0.969
training_f1_score	0.969
training_log_loss	0.121
training_precision_score	0.969
training_recall_score	0.969
training_roc_auc	0.994
training_score	0.969

Tags (2)

Artifacts

The screenshot shows the MLflow Model page for a model named 'best_estimator'. The page displays the model's schema and provides code snippets for making predictions using the logged model.

Model Details:

- Full Path: mlflow-artifacts/682780464809541461/bde9be51f74f4ff5b3147807e6562e89/artifacts/best_estimator
- Registered on: 2023/02/09

Model schema

Input and output schema for your model. [Learn more](#)

Name	Type
Inputs (2520)	
000	double
00pm	double
01	double
10	double
100	double
Outputs (1)	
-	Tensor (dtype: int64, shape: [-1])

Make Predictions

Predict on a Spark DataFrame:

```
import mlflow
from pyspark.sql.functions import struct, col
logged_model = 'runs:/bde9be51f74f4ff5b3147807e6562e89/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=logged_model, result_type='double')

# Predict on a Spark DataFrame.
df.withColumn('predictions', loaded_model(struct(*map(col, df.columns))))
```

Predict on a Pandas DataFrame:

```
import mlflow
logged_model = 'runs:/bde9be51f74f4ff5b3147807e6562e89/best_estimator'

# Load model as a PyFuncModel.
loaded_model = mlflow.pyfunc.load_model(logged_model)

# Predict on a Pandas DataFrame.
import pandas as pd
loaded_model.predict(pd.DataFrame(data))
```

3. COMPARISON

	Simple	Avancé	+Word2Vec	+Glov.6B	roBERTa (TweetEval)
Preprocessing pipeline	irfantoor/sentiment				
Tokenizer/ Vectorizer	Tfidf- Vectorizer	Keras	Keras	Keras	Twitter- roBERTa- base
Type Vecteur	Sparse	Dense	Dense	Dense	Dense
Dimensions des vecteurs	2475	128	128	100	768
Modèle-Type	Linear	Deep	Deep	Deep	Deep
Modèle	GaussianNB	LSTM	LSTM	LSTM	Twitter- roBERTa- base
Embedding	-	-	Word2Vec Shakespea re	Glove.6B .100	twitter- roberta- base- sentiment- latest
Nombres de vecteurs (emb.)			~111K	400K	~124M
Nombres des Tokens (Emb.)	-	-	9239	51037	50265
Entrainement	Oui	Oui	Oui	Oui	Non
Temps d'entraînement		~84 sec.	~48 sec.	~45 sec.	-
Taille de sample	1000	100000	100000	100000	-
Optimization/ Epoch	Grid- Search	3	4	4	-
Accuracy	0.667	0.8606	0.7976	0.7982	0.71
Val-accuracy	-	0.7288	0.7500	0.7449	-

4. PREDICTIONS

4.1 Simple Model

Phrase	Sentiment
Its a test, He'll Hello World!!! ;-) OFF You Go !! https://irfantoor.com via @irfantoor #contact 123	positif
than would Hello world, its a @test Tweet, follow me at http://irfantoor.com , AMAZING!	positif
Its another great test tweet, so SWEET!, follow me at https://irfantoor.com , AMAZING !	positif
enjoying the inspiring and amazing World !!! I AM AMAZED	positif
up all night ..baby had her injections ..poor little thing	négatif
my iPad...apple.again @apple #stop	négatif
I am sleepy	négatif
I love the world	positif
I'm tired	négatif
I enjoy being here	positif
Tsunami destroyed half of the Island	neutre

4.2 Model roBERTa (TweetEval)

Phrase	Sentiment	score
I like you, I love you	positive	0.9231
Good night 😊	positive	0.9625
Highway to hell	negative	0.8268
She is buying a stairway to heaven	neutral	0.5513
What time is it?	neutral	0.9074
Covid cases are increasing fast!	negative	0.7236
Its a test, He'll Hello World!!! ;-) OFF You Go !! https://irfantoor.com via @irfantoor #contact 123	positive	0.7888
than would Hello world, its a @test Tweet, follow me at http://irfantoor.com , AMAZING!	positive	0.9850
Its another great test tweet, so SWEET!, follow me at https://irfantoor.com , AMAZING !	positive	0.9900
enjoying the inspiring and amazing World !!! I AM AMAZED	positive	0.9897
up all night ..baby had her injections ..poor little thing	negative	0.8044
my iPad...apple.again @apple #stop	negative	0.8731
I am sleepy	neutral	0.6288
Tsunami destroyed half of the Island	negative	0.8572

5. CONCLUSION

It's a complex and time-consuming process to first choose a model, then adding the training layers according to the type of data and then optimizing the model. There are no particular standards defining the vector lengths of embeddings, but models are seemingly trying to stick to the de-facto and/or the precedence established by the pioneers of the models in different categories.

In the meantime it is important to note that the <https://huggingface.co/> is an exceptional resource for the pre-trained models or embeddings, for eliminating the time and huge computing resources required to train the models. These pretrained models and embeddings can be fine-tuned with our own data to handle a specific task.

6. DECLARATION

I am not affiliated to any of the organisations, tools or technologies, referred or used while doing the experiments, with an intended direct or indirect financial interest, and I have tried my level best to be rational and unbiased regarding the comparison of the discussed models.

7. ACKNOWLEDGEMENTS

I thank **twitter** for the availability of the dataset, all of the people involved in the production and availability of *python*, the m/l libraries specially *Numpy*, *Pandas*, *Scikit-learn*, *Tensorflow*, *Keras*, *GaussianNB*, *MLFlow*, *LSTM*, *Huggingface* for making available the pretrained NLP models like **BERT** etc. and twitter for the availability of the tweets so that a comparison of different models for a *sentiment analysis* could be done in such a short time.

I personally thank the team behind *OpenClassrooms* for adding the innovative references to the courses and specially to my mentor **Mr. Panayotis Papoutsis** for his invaluable guidelines and advices.

8. REFERENCE

- [A platform for the machine learning lifecycle | MLflow](#)
- [MLflow Documentation — MLflow 2.1.1 documentation](#)
- [MLOps Zoomcamp](#)
- <https://irfantoor.com/mlops-and-comparison-of-different-models>