



TENSORFLOW EXTENDED

Introduction to Big Data – Group 4 – Machine Learning tool

MEMBERS



20127484
Ng Tư Duy



20127383
Lê Ngọc Tường



20127588
Ng Tấn Phát

20127374
Ng Đức Trường



TABLE OF CONTENTS

01

Overview

02

TFX
PIPELINE

03

TFX
COMPONENTS

04

Conclusion



01

OVERVIEW

INTRODUCTION



TensorFlow Extended (TFX) là một nền tảng phát triển mã nguồn mở của Google, được thiết kế để xây dựng các ứng dụng học máy sản xuất và quản lý quá trình thực hiện mô hình học máy.

TFX bao gồm một bộ các công cụ, thư viện và khung công việc cho phép các nhà phát triển và chuyên gia dữ liệu xây dựng, huấn luyện, triển khai và quản lý các mô hình học máy. Nó cung cấp cho người dùng các khả năng tiền xử lý dữ liệu, huấn luyện mô hình, kiểm tra và đánh giá mô hình, triển khai mô hình và quản lý quá trình sản xuất.



TensorFlow Extended



“Với TFX, các nhà phát triển và chuyên gia dữ liệu có thể tập trung vào việc phát triển các mô hình học máy chất lượng cao, đồng thời giảm thiểu thời gian và công sức trong việc quản lý các công việc liên quan đến tiền xử lý dữ liệu, huấn luyện mô hình và triển khai mô hình...”

WHAT IS APACHE BEAM ?

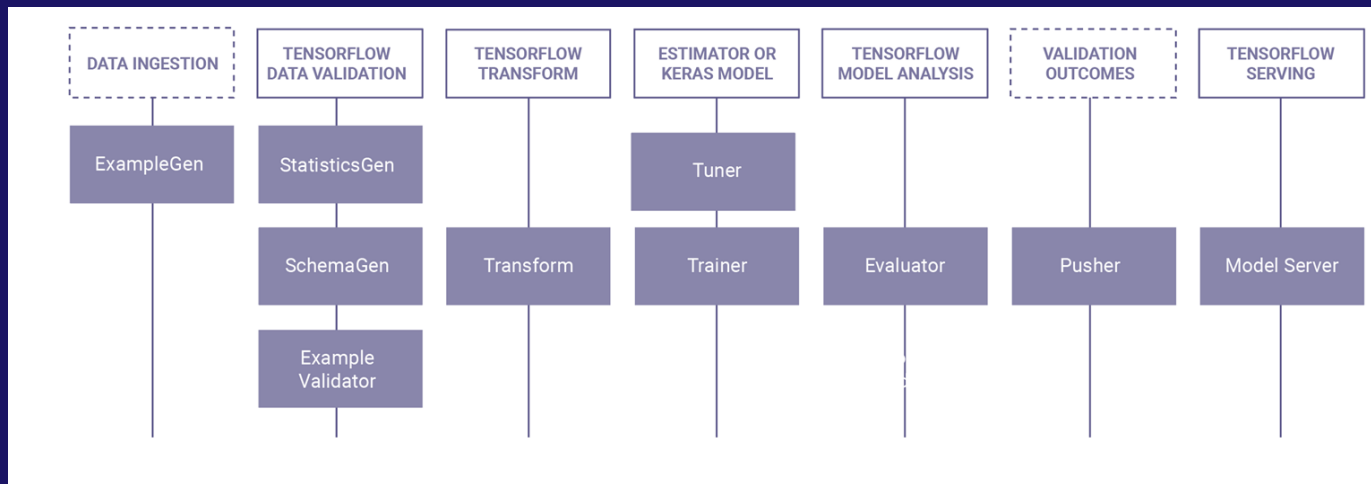
Apache Beam là một framework mã nguồn mở cho xử lý dữ liệu phân tán. Nó cung cấp một API đơn giản để xây dựng các pipeline xử lý dữ liệu phức tạp trên các hệ thống phân tán như Apache Spark, Google Cloud Dataflow...

Apache Beam được thiết kế để hỗ trợ việc xử lý dữ liệu liên tục và dữ liệu theo lô (batch), cũng như các loại dữ liệu khác nhau như văn bản, hình ảnh, video, âm thanh và dữ liệu định dạng phức tạp. Nó cung cấp các công cụ và thư viện để thực hiện các tác vụ như rút trích đặc trưng, xử lý dữ liệu thô, tối ưu hóa hiệu suất và triển khai các ứng dụng phân tán.

TFX sử dụng Apache Beam như một công cụ để thực hiện các tác vụ xử lý dữ liệu và huấn luyện mô hình trên các tập dữ liệu lớn và phân tán.



WHAT IS TENSORFLOW EXTENDED??



TFX là một end-to-end Machine Learning pipeline. Đây là một chuỗi các bước xử lý dữ liệu và huấn luyện mô hình từ dữ liệu đầu vào cho đến triển khai mô hình vào môi trường sản xuất.



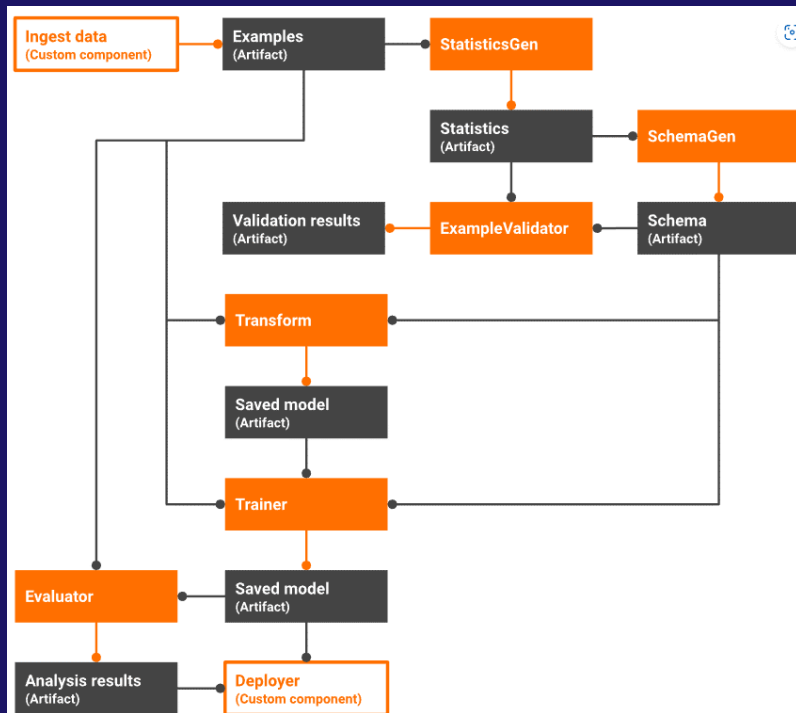
02

TFX PIPELINES



- **Data Ingestion:** lấy dữ liệu từ các nguồn khác nhau và chuyển đổi chúng thành dữ liệu đầu vào đồng nhất...
- **TensorFlow Data Validation:** Kiểm tra tính chính xác và tính đầy đủ của dữ liệu, phát hiện và xử lý các giá trị thiếu và ngoại lai và đảm bảo tính nhất quán của dữ liệu
- **TensorFlow Transform:** Là quá trình chuyển đổi dữ liệu đầu vào thành dữ liệu được chuẩn hóa phù hợp với mô hình học máy.
- **Estimator or Keras Model:** Là 2 lựa chọn cho thành phần Trainer. Đây là các framework để xây dựng mô hình ML trên Tensorflow.
- **Data Model Analysis:** đánh giá chất lượng của mô hình học máy. Nó cung cấp các công cụ để tính toán các chỉ số đánh giá và biểu diễn chúng dưới dạng biểu đồ hoặc báo cáo.
- **Validation Outcomes:** là kết quả của quá trình kiểm tra mô hình học máy trên các tập dữ liệu đầu vào khác nhau. Kết quả này được sử dụng để đánh giá tính hiệu quả của mô hình và đảm bảo tính nhất quán của nó khi được triển khai vào sản xuất.
- **TensorFlow Serving:** Triển khai các mô hình đã được huấn luyện trên các hệ thống sản xuất và cung cấp một API cho phép các ứng dụng khác sử dụng.

TFX PIPELINES



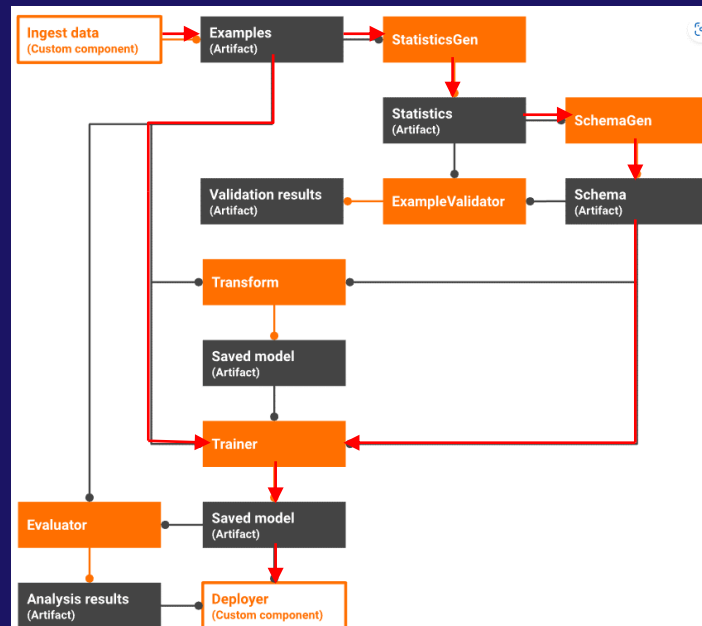
Pipeline trong TFX là một quy trình triển khai được đóng gói và có thể chạy trên các nền tảng khác nhau như Apache Airflow, Apache Beam, và Kubeflow Pipelines. Một pipeline bao gồm các thành phần và tham số đầu vào.

Các thành phần của pipeline sẽ tạo ra các đầu ra (artifacts) và thường phụ thuộc vào các đầu vào được tạo ra bởi các thành phần trước đó. Thứ tự thực thi các thành phần được xác định bằng cách tạo một đồ thị có hướng không tuần hoàn (directed acyclic graph) của các phụ thuộc đầu vào.

TFX PIPELINES

Ví dụ, một pipeline có thể bao gồm các bước sau:

1. Lấy dữ liệu trực tiếp từ hệ thống độc quyền bằng cách sử dụng thành phần tùy chỉnh để đưa dữ liệu vào pipeline.
2. Tính toán thống kê cho dữ liệu huấn luyện bằng cách sử dụng thành phần tiêu chuẩn StatisticsGen.
3. Tạo schema dữ liệu sử dụng thành phần tiêu chuẩn SchemaGen.
4. Kiểm tra dữ liệu huấn luyện để phát hiện các lỗi sử dụng thành phần tiêu chuẩn ExampleValidator.
5. Thực hiện kỹ thuật Feature Engineering trên tập dữ liệu sử dụng thành phần tiêu chuẩn Transform.
6. Huấn luyện mô hình sử dụng thành phần tiêu chuẩn Trainer.
7. Đánh giá mô hình đã được huấn luyện bằng cách sử dụng thành phần tiêu chuẩn Evaluator.
8. Nếu mô hình đã vượt qua đánh giá, pipeline sẽ lưu mô hình đã huấn luyện vào hệ thống triển khai độc quyền bằng cách sử dụng thành phần tùy chỉnh.



BUILDING TFX PIPELINES

Step 1

Định nghĩa pipeline

Step 2

Cấu hình pipeline

Step 3

Thực hiện pipeline

Step 4

Biên dịch pipeline

Step 5

Chạy pipeline

Step 6

Giám sát pipeline

Create a pipeline

```
def _create_pipeline(pipeline_name: str, pipeline_root: str,
                    data_root: str, module_file: str,
                    serving_model_dir: str, metadata_path: str)
    -> tfx.dsl.Pipeline:

    example_gen = tfx.components.CsvExampleGen(...)
    trainer = tfx.components.Trainer(...)
    pusher = tfx.components.Pusher(...)

    components = [example_gen, trainer, pusher]

    return tfx.dsl.Pipeline(
        pipeline_name=pipeline_name,
        pipeline_root=pipeline_root,
        metadata_connection_config=tfx.orchestration.metadata
        .sqlite_metadata_connection_config(metadata_path),
        components=components)
```

Định nghĩa hàm tạo pipeline.

Trong đó cấu hình từng thành phần trong pipeline.

- **pipeline_name**: tên của pipeline (phải là duy nhất).
- **pipeline_root**: đường dẫn gốc của kết quả đầu ra của pipeline.
- **metadata_connection_config**: cấu hình kết nối cho ML metadata.
- **components**: một list các thành phần cấu thành nên quy trình làm việc của pipeline.



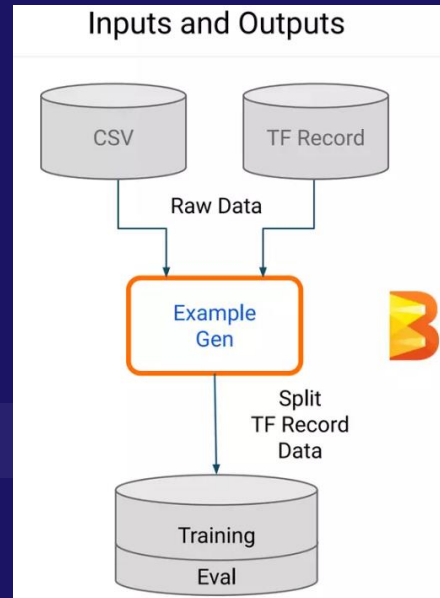
03

TFX COMPONENTS

Component: ExampleGen

- Component **ExampleGen** trong TFX có chức năng đọc dữ liệu vào pipeline từ các nguồn khác nhau và chuyển đổi chúng thành định dạng Example record / Sequence Ex record. Và cũng có thể tạo ra các tập dữ liệu train và eval bằng cách chia dữ liệu đầu vào theo tỉ lệ đã chỉ định.
- Trong một pipeline thực tế, ExampleGen thường là bước đầu tiên để đọc dữ liệu vào pipeline, chuẩn hóa định dạng và chuẩn bị cho các thành phần khác trong pipeline.

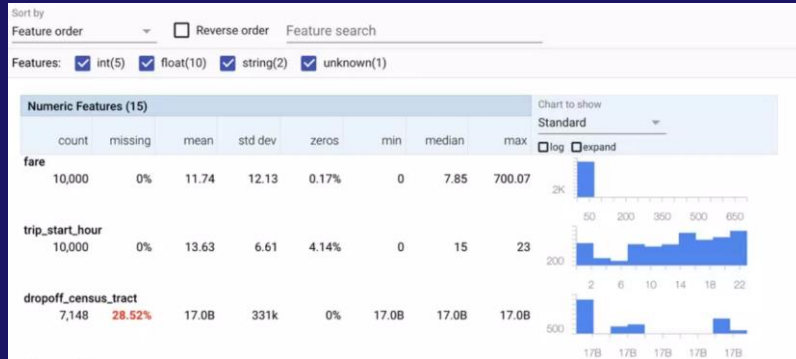
```
examples = csv_input(os.path.join(data_root, 'simple'))  
example_gen = CsvExampleGen(input_base=examples)
```



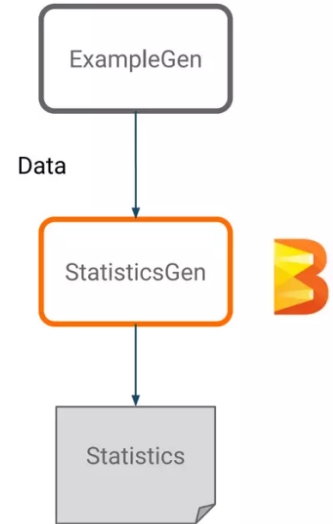
Component: StatisticsGen

- Component **StatisticsGen** trong TFX có chức năng tính toán các thống kê quan trọng của dữ liệu đầu vào, bao gồm số lượng, trung bình, phương sai, giá trị lớn nhất và nhỏ nhất, để giúp người dùng hiểu hơn về dữ liệu và kiểm tra tính hợp lệ của dữ liệu đầu vào.
- StatisticsGen có thể hoạt động trên các định dạng dữ liệu khác nhau như CSV, JSON hoặc TensorFlow Record. Nó cũng hỗ trợ việc thêm các biến đổi dữ liệu tùy chỉnh để tính toán các thống kê phù hợp với yêu cầu của người dùng.

```
statistics_gen =  
    StatisticsGen(input_data=example_gen.outputs.examples)
```



Inputs and Outputs



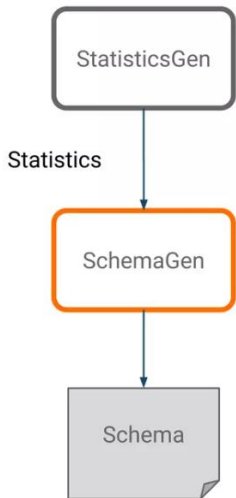
Component: SchemaGen

- **Component SchemaGen** trong TFX là component được sử dụng để tạo ra schema cho dữ liệu đầu vào. Schema là mô tả cấu trúc của dữ liệu, bao gồm tên, kiểu dữ liệu và hình dạng của các tính năng trong dữ liệu.
- **Component SchemaGen** giúp đảm bảo rằng dữ liệu đầu vào là hợp lệ và phù hợp với mong đợi trước khi sử dụng trong các component downstream của pipeline TFX, chẳng hạn như ExampleValidator và Transform.

```
infer_schema = SchemaGen(stats=statistics_gen.outputs.output)
```

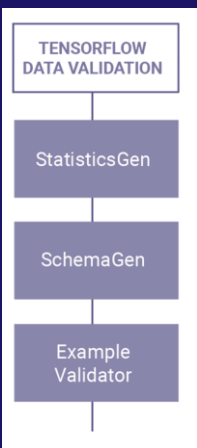
	Type	Presence	Valency	Domain
Feature name				
'fare'	FLOAT	required	single	-
'trip_start_hour'	INT	required	single	-
'pickup_census_tract'	BYTES	optional		-
'dropoff_census_tract'	FLOAT	optional	single	-
'company'	STRING	optional	single	'company'
'trip_start_timestamp'	INT	required	single	-

Inputs and Outputs



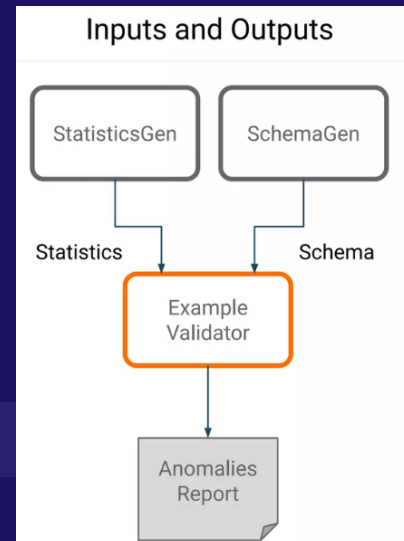
Component: Example Validator

- Component `ExampleValidator` trong TFX được sử dụng để kiểm tra và đánh giá tính hợp lệ của dữ liệu huấn luyện. Nó sẽ xác định các giá trị của các trường đầu vào của dữ liệu huấn luyện có nằm trong các giới hạn được xác định trước đó hay không, và tạo ra một bản báo cáo về các giá trị ngoại lai, dữ liệu thiếu, hoặc không hợp lệ. Các báo cáo này sẽ giúp cho nhà phát triển có thể xác định được các lỗi trong dữ liệu huấn luyện và thực hiện các chỉnh sửa cần thiết để cải thiện chất lượng của mô hình.



```
validate_stats = ExampleValidator(  
    stats=statistics_gen.outputs.output,  
    schema=infer_schema.outputs.output)
```

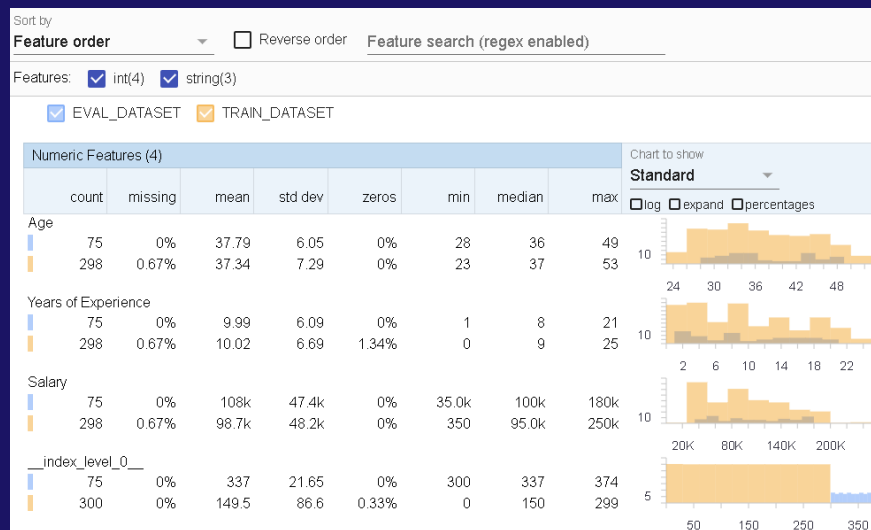
Anomaly short description		Anomaly long description
Feature name		
'payment_type'	Unexpected string values	Examples contain values missing from the schema: Prcard (<1%).
'company'	Unexpected string values	Examples contain values missing from the schema: 2092 - 61288 Sbeih company (<1%), 2192 - 73487 Zeymane Corp (<1%), 2192 - Zeymane Corp (<1%), 2823 - 73307 Seung Lee (<1%), 3094 - 24059 G.L.B. Cab Co (<1%), 3319 - CD Cab Co (<1%), 3385 - Eman Cab (<1%), 3897 - 57856 Ilie Malec (<1%), 4053 - 40193 Adwar H. Nikola (<1%), 4197 - Royal Star (<1%), 585 - 88805 Valley Cab Co (<1%), 5874 - Sergey Cab Corp. (<1%), 6057 - 24657 Richard Addo (<1%), 6574 - Babylon Express Inc. (<1%), 6742 - 83735 Tasha ride inc (<1%).



Lib: TensorFlow Data Validation

Một vài function phổ biến:

- `generate_statistics_from_csv`: tạo ra số liệu thống kê từ dữ liệu file CSV.
- `infer_schema`: tạo ra schema từ số liệu thống kê đã có.



`visualize_statistics`: trực quan hóa số liệu thống kê.

	Type	Presence	Valency	Domain
Feature name				
'Age'	INT	optional	single	-
'Gender'	STRING	optional	single	'Gender'
'Education Level'	STRING	optional	single	'Education Level'
'Job Title'	BYTES	optional	single	-
'Years of Experience'	INT	optional	single	-
'Salary'	INT	optional	single	-
'__index_level_0__'	INT	required		-
Values				
Domain				
'Gender'	'Female', 'Male'			
'Education Level'	'Bachelor's', 'Master's', 'PhD'			

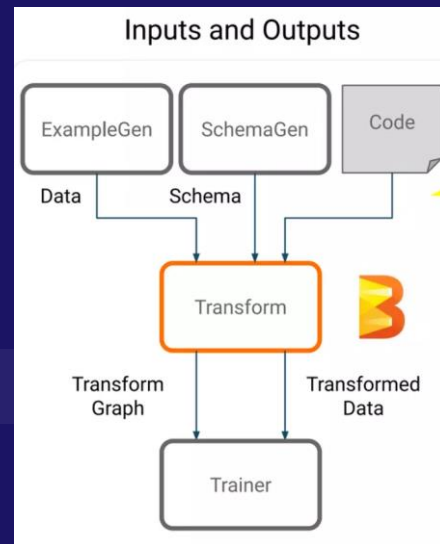
`validate_statistics`: tham chiếu đến schema để kiểm tra dị thường trong dữ liệu.

Component: Transform

- **Component Transform** trong TFX pipeline có chức năng chuyển đổi và tiền xử lý dữ liệu đầu vào trước khi đưa vào huấn luyện mô hình. Nó cũng có thể thực hiện một số tác vụ như chọn lọc và kết hợp các tính năng (feature) để tạo ra các tính năng mới. Đầu vào của component Transform là dữ liệu huấn luyện và Schema được tạo bởi SchemaGen, đầu ra là dữ liệu tiền xử lý mới (transformed data) và transform_fn. Transform_fn sẽ được sử dụng trong quá trình tiền xử lý dữ liệu ở các bước tiếp theo, bao gồm huấn luyện và serving model.

```
transform = Transform(  
    input_data=example_gen.outputs.examples,  
    schema=infer_schema.outputs.output,  
    module_file=taxi_module_file)
```

```
for key in _DENSE_FLOAT_FEATURE_KEYS:  
    outputs[_transformed_name(key)] = transform.scale_to_z_score(  
        _fill_in_missing(inputs[key]))  
# ...
```



Lib: TensorFlow Transform

Một vài function phổ biến:

- Scaling functions: scale dữ liệu số đến phạm vi xác định.

E.g. `tft.scale_to_0_1`

0	age	12	hours-per-week
39	0.493151	40	0.479592
50	0.219178	13	0.500000
38	0.438356	40	0.397959
53	0.465753	40	0.397959
28	0.328767	40	0.479592
37	0.342466	40	0.397959
49	0.095890	40	0.397959
52	0.095890	16	0.397959
31	0.095890	45	0.397959

- Mapping functions: map dữ liệu category thành vocabulary.

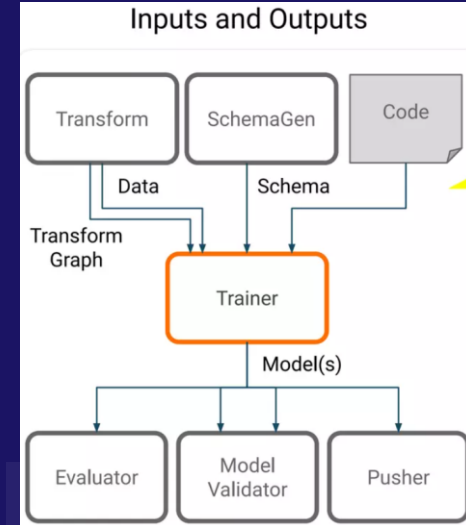
E.g. `tft.compute_and_apply_vocabulary`,

7	relationship	3	education
Not-in-family	0	Bachelors	0
Husband	0	Bachelors	2
Not-in-family	1	HS-grad	1
Husband	0	11th	4
Wife	0	Bachelors	0
Wife	3	Masters	0
Not-in-family	1	9th	1
Husband	3	HS-grad	13
		Masters	3

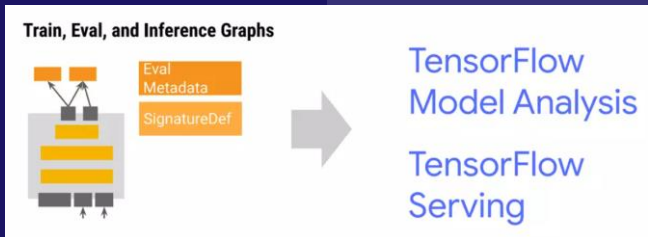
Component: Trainer

- Trainer Component training mô hình machine learning trên dữ liệu đã được chuẩn bị và xử lý bởi các thành phần trước đó trong pipeline. Trong quá trình huấn luyện, Trainer component sẽ sử dụng các thông số được cấu hình để huấn luyện mô hình. Ngoài ra, nó cũng có khả năng lưu trữ và xuất ra các tệp checkpoint của mô hình và trọng số tương ứng (saved model) để có thể sử dụng lại cho các mục đích sau này.

Kết quả đầu ra của Trainer component là một mô hình đã được huấn luyện sẵn sàng được sử dụng cho việc dự đoán.

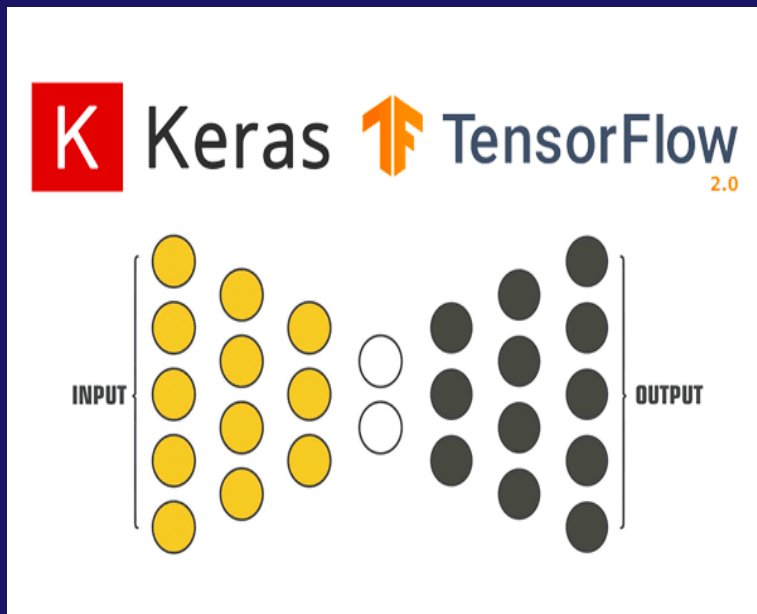


```
trainer = Trainer(  
    module_file=taxi_module_file,  
    transformed_examples=transform.outputs.transformed_examples,  
    schema=infer_schema.outputs.output,  
    transform_output=transform.outputs.transform_output,  
    train_steps=10000,  
    eval_steps=5000,  
    warm_starting=True)
```

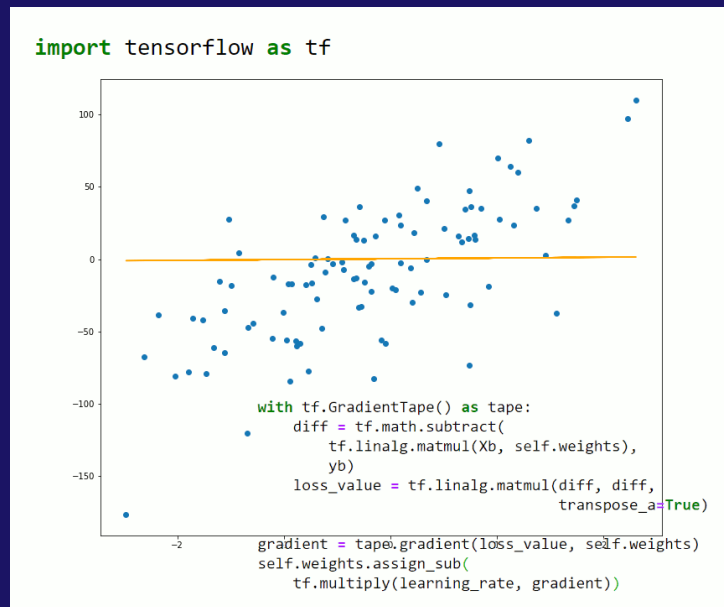


Lib: TensorFlow Training

Một vài module phổ biến: `tf.saved_model`: lưu model đã được train.



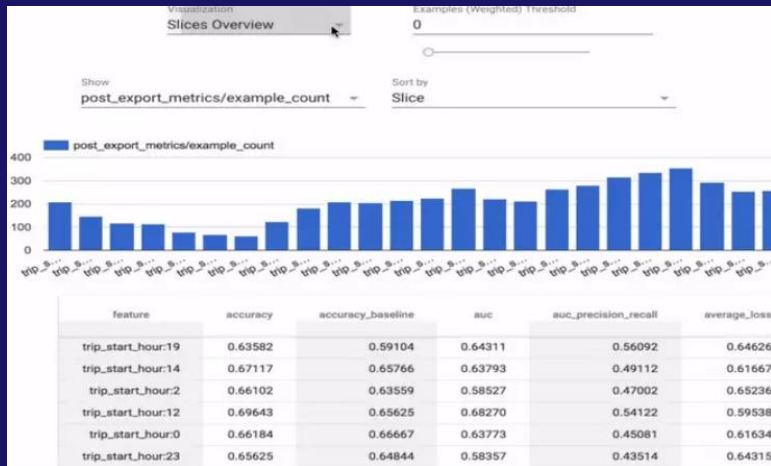
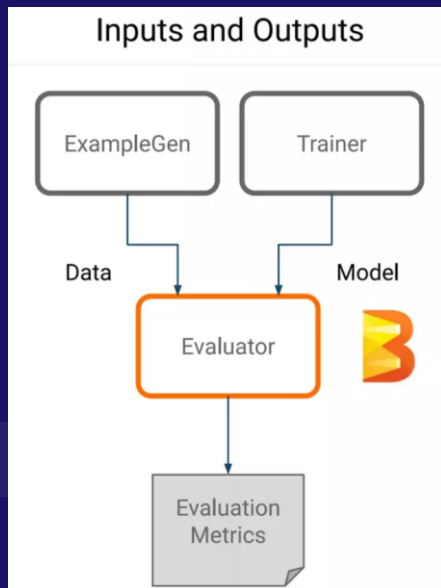
`tf.keras`: thực hiện API cho keras - huấn luyện các mô hình DNN.



`tf.linalg`: thực hiện các phép toán tuyến tính trên các tensor như tích chập, ma trận nghịch đảo, ...

Component: Evaluator

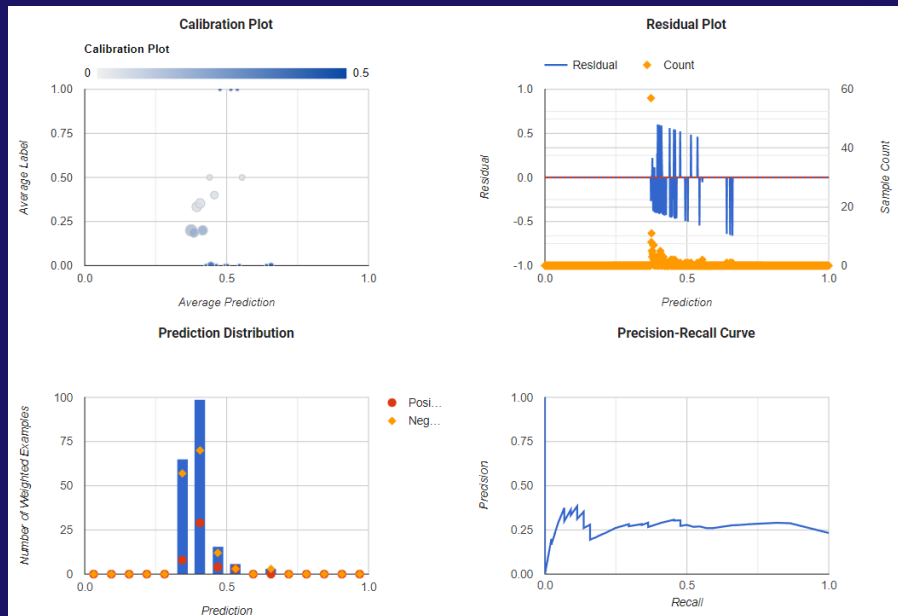
- **Component Evaluator** đánh giá chất lượng mô hình huấn luyện. Evaluator sử dụng các metric đã được định nghĩa để tính toán như accuracy, precision, recall, AUC, F1-score,... Và đưa ra kết quả của mô hình trên tập dữ liệu kiểm tra hoặc tập dữ liệu mới.
- Nếu mô hình đạt được chất lượng đủ tốt, Evaluator sẽ tạo ra các metadata để lưu trữ kết quả đánh giá mô hình. Nếu mô hình không đạt được chất lượng đủ tốt, nó sẽ không được triển khai.



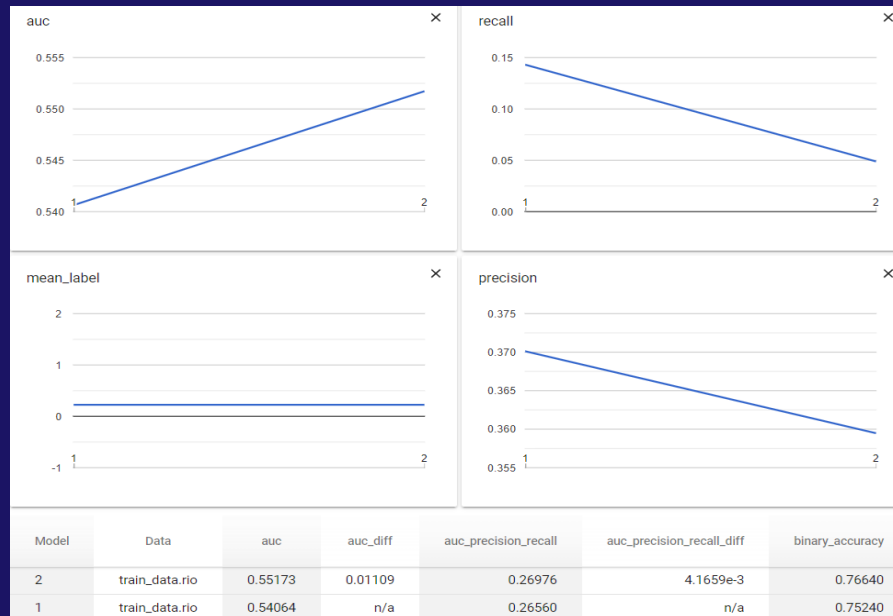
```
model_analyzer = Evaluator(  
    examples=examples_gen.outputs.output,  
    eval_spec=taxi_eval_spec,  
    model_exports=trainer.outputs.output)
```

Lib: TensorFlow Model Analysis

Một vài function phổ biến:



`tfma.view.render_plot()` được sử dụng để hiển thị biểu đồ dữ liệu của kết quả đánh giá mô hình được lưu trữ



`tfma.view.render_time_series` sử dụng để hiển thị sự thay đổi hoặc xu hướng của các metric theo thời gian

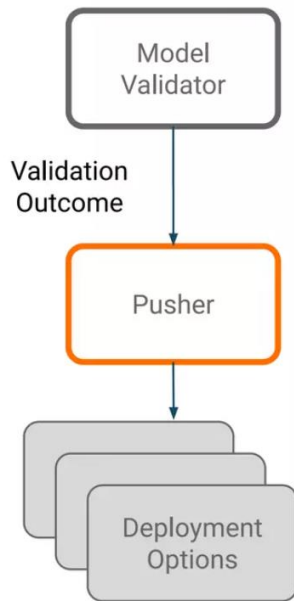
Component: Pusher

- **Component Pusher** được sử dụng để triển khai (deploy) một mô hình đã được huấn luyện lên một hệ thống khác để sử dụng trong thực tế. Component này sẽ đóng gói mô hình vào một định dạng chuẩn để có thể được sử dụng bởi TFServing hoặc các hệ thống khác.
- Nó cũng có thể tạo ra các tệp metadata để giúp theo dõi mô hình đã được triển khai như thế nào. Các thông số cấu hình như địa chỉ và cổng đích của hệ thống triển khai được chỉ định thông qua input parameter của component Pusher.

```
pusher = Pusher(  
    model_export=trainer.outputs.output,  
    model_blessing=model_validator.outputs.blessing,  
    serving_model_dir=serving_model_dir)
```

- Block push on validation outcome
- Push destinations supported today
 - Filesystem (TensorFlow Lite, TensorFlow JS)
 - TensorFlow Serving

Inputs and Outputs



Lib: TensorFlow Serving

Một vài cách gọi API phổ biến:

- Run serving: run
- với tensorflow_model_server hoặc có thể run với docker đã pull tensorflow/serving.
- Thực hiện gọi predict: bằng cách curl (POST) đến API của model.
- Ngoài ra, cũng có thể thực hiện tự build ModelServer.

```
2023-04-18 20:55:49.017065: I external/org_tensorflow/tensorflow/cc/saved_model/loader.cc:213] Running initialization op
on SavedModel bundle at path: /mnt/c/Users/HOME/Downloads/models/penguin_full_model/1681738315
2023-04-18 20:55:49.041077: I external/org_tensorflow/tensorflow/cc/saved_model/loader.cc:305] SavedModel load for tags
{ serve }; Status: success: OK. Took 354088 microseconds.
2023-04-18 20:55:49.048617: I tensorflow_serving/servables/tensorflow/saved_model_warmup_util.cc:62] No warmup data file
found at /mnt/c/Users/HOME/Downloads/models/penguin_full_model/1681738315/assets.extra/tf_serving_warmup_requests
2023-04-18 20:55:49.240841: I tensorflow_serving/core/loader_harness.cc:95] Successfully loaded servable version {name:
penguin_model version: 1681738315}
2023-04-18 20:55:49.242845: I tensorflow_serving/model_servers/server_core.cc:486] Finished adding/updating models
2023-04-18 20:55:49.243065: I tensorflow_serving/model_servers/server.cc:118] Using InsecureServerCredentials
2023-04-18 20:55:49.243107: I tensorflow_serving/model_servers/server.cc:383] Profiler service is enabled
2023-04-18 20:55:49.256803: I tensorflow_serving/model_servers/server.cc:409] Running gRPC ModelServer at 0.0.0.0:8500
...
2023-04-18 20:55:49.260649: I tensorflow_serving/model_servers/server.cc:430] Exporting HTTP/REST API at:localhost:8501
...
[evhttp_server.cc : 245] NET_LOG: Entering the event loop ...
```

```
(base) (lengoctuong@kali-linux)-[~]
$ curl -d '{"signature_name": "serving_default", "instances": [{"b64": "Cm8KFQoLYm9keV9tYXNzX2cSBhoECgKYKgobCg9jdWxtZW
5fZGVvdGhfbW0SCBICGcTNzIBBChsKEWZsaXBwZXJfbG9uZ3RoX21tEgYaBAoC1QEKHAoQY3VsbnVudX2lbmd0aF9tbRIIEgYKBjQZR0I="}]}' -X POST
http://localhost:8501/v1/models/penguin_model:predict
{
  "predictions": [[-6.95050526, -3.03205, 2.62595367]
]
}
(base) (lengoctuong@kali-linux)-[~]
$
```



04

Conclusion

Advantages and Disadvantages

Ưu điểm





- Tính toàn diện: Là một nền tảng end to end, TFX cung cấp nhiều công cụ để giải quyết hầu hết mọi vấn đề trong việc triển khai mô hình Machine learning.
- Hoạt động tốt trên dữ liệu có quy mô lớn và phức tạp.
- Tính linh hoạt: Cho phép tùy chỉnh các thành phần pipeline và thêm các thành phần mới để đáp ứng nhu cầu đặc biệt của dự án.
- Tính linh động: Có thể chạy trên nhiều nền tảng khác nhau, bao gồm Google Cloud, Apache Beam, Apache Airflow và Kubeflow Pipelines.

Advantages and Disadvantages

Nhược điểm

- Cần nhiều thời gian để học tập: TFX là một nền tảng phức tạp với nhiều thành phần và chức năng, với những người mới thì để sử dụng TFX một cách hiệu quả sẽ phải mất một thời gian nhất định để làm quen.
- Tuy có khả năng tùy chỉnh linh hoạt các thành phần của mình, tuy nhiên để điều chỉnh hiệu quả thì người dùng cũng cần phải có một số kinh nghiệm để thực hiện.
- Tích hợp: Với các công nghệ, nền tảng không phải của Google thì việc sử dụng TFX sẽ gặp thêm một số khó khăn.

Other solutions

Platform	Description	Pros	Cons
TFX  TensorFlow	Platform end-to-end của Google giúp xây dựng các pipeline học máy có quy mô lớn trên Tensorflow.	Tích hợp chặt chẽ với TensorFlow và cung cấp các thành phần để xử lý dữ liệu, đào tạo và đánh giá mô hình, và triển khai mô hình.	Để sử dụng tốt yêu cầu phải có kinh nghiệm. Gặp vài hạn chế khi sử dụng trên các nền tảng không phải của Google.
Kubeflow Pipelines  Kubeflow	Platform mã nguồn mở để xây dựng và triển khai quy trình học máy end-to-end trên Kubernetes.	Khả năng mở rộng và linh hoạt cao do dựa vào Kubernetes để quản lý tất cả việc thực thi mã, quản lý tài nguyên và mạng.	Để sử dụng tốt yêu cầu phải có kinh nghiệm.
Apache Airflow  Apache Airflow	Platform mã nguồn mở để lập lịch và giám sát các quy trình làm việc dưới dạng mã Python.	Có một thư viện lớn các công cụ.	Giao diện người dùng có thể không phản hồi nhanh và không phù hợp cho những mô hình học máy đặc biệt.
MLflow  mlflow	Platform mã nguồn mở để quản lý vòng đời của quy trình học máy.	Tương thích với nhiều nền tảng học máy khác nhau.	Các dịch vụ về model serving chưa thực sự tốt.



Application

Cung cấp các công cụ cần thiết.

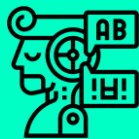
Tự động hóa quy trình.

Xây dựng các pipeline cần sự linh hoạt.

Triển khai mô hình trong thực tế.



Application: Research and development



Xử lý ngôn ngữ tự nhiên
(NLP)



Xây dựng các mô hình
phân loại.



Xử lý hình ảnh



Xử lý giọng nói

Application: Industry



THANKS !

Do you have any questions?

