VMware Tanzu Greenplum

Greenplum MLOps on K8s

林士傑　Jay Lin

Tanzu Data, VMware Taiwan

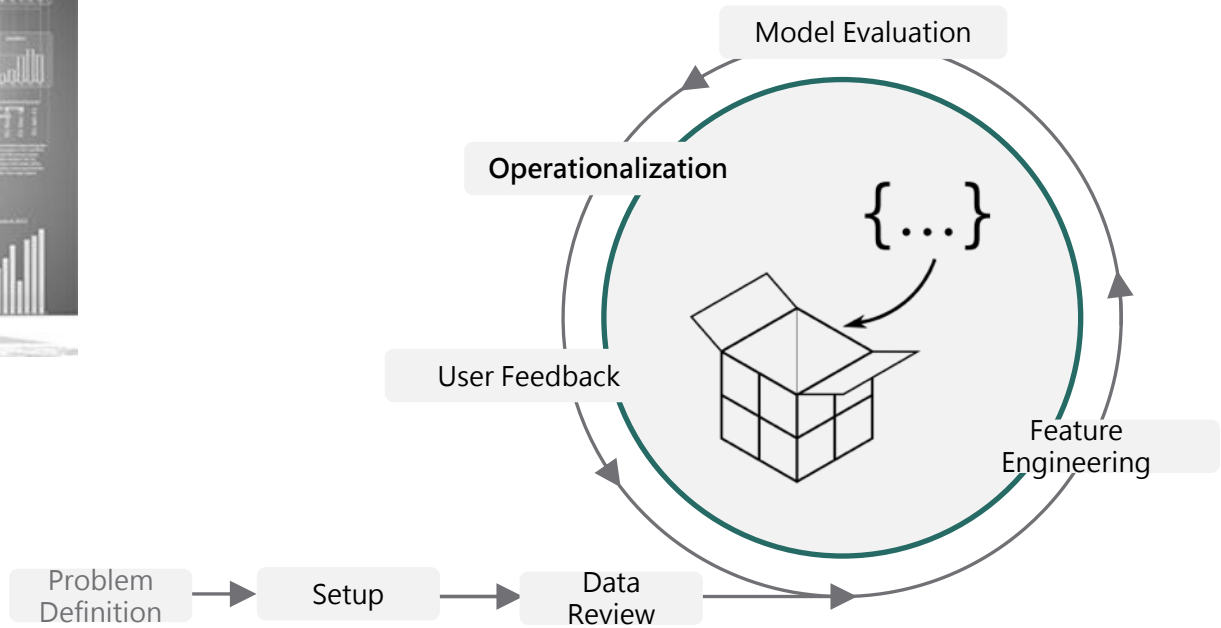
資深技術顧問

2022/10/19

Agenda

- 資料科學與機器學習作業自動化 (MLOps)
- Greenplum 支持 MLOps 的關鍵要素
  - MADlib & In-DB Analytics
  - PL/Container、PL/Python, & Containerized Deployment
  - Storing & Control Mechanism of ML Models
- “事件驅動”(Event-Driven)模型預測機制
  - Demo

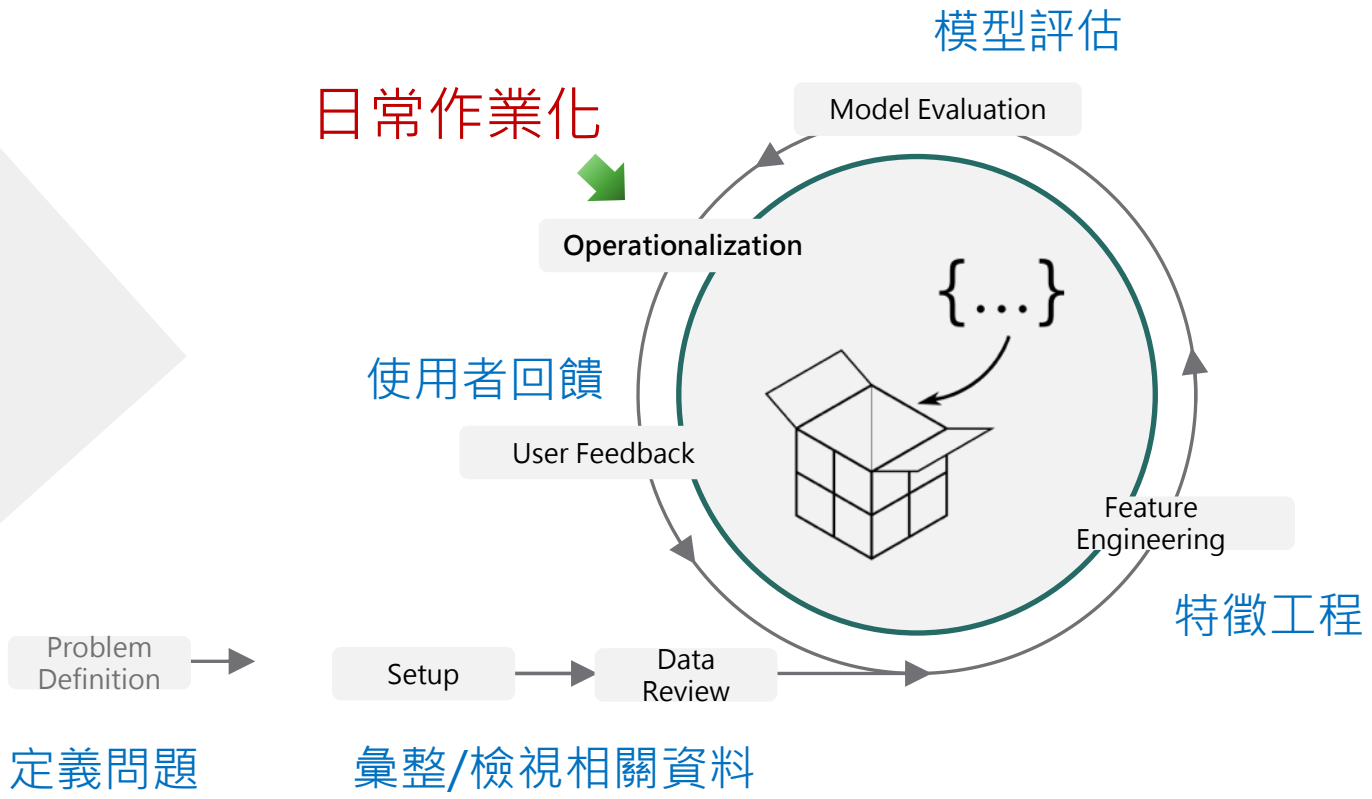
資料科學的作業流程


Model Evaluation
Operationalization
{...}
User Feedback
Feature Engineering
Problem Definition
Setup
Data Review

ML-Ops ! (Machine Learning Models Operationalization)

ML模型作業化

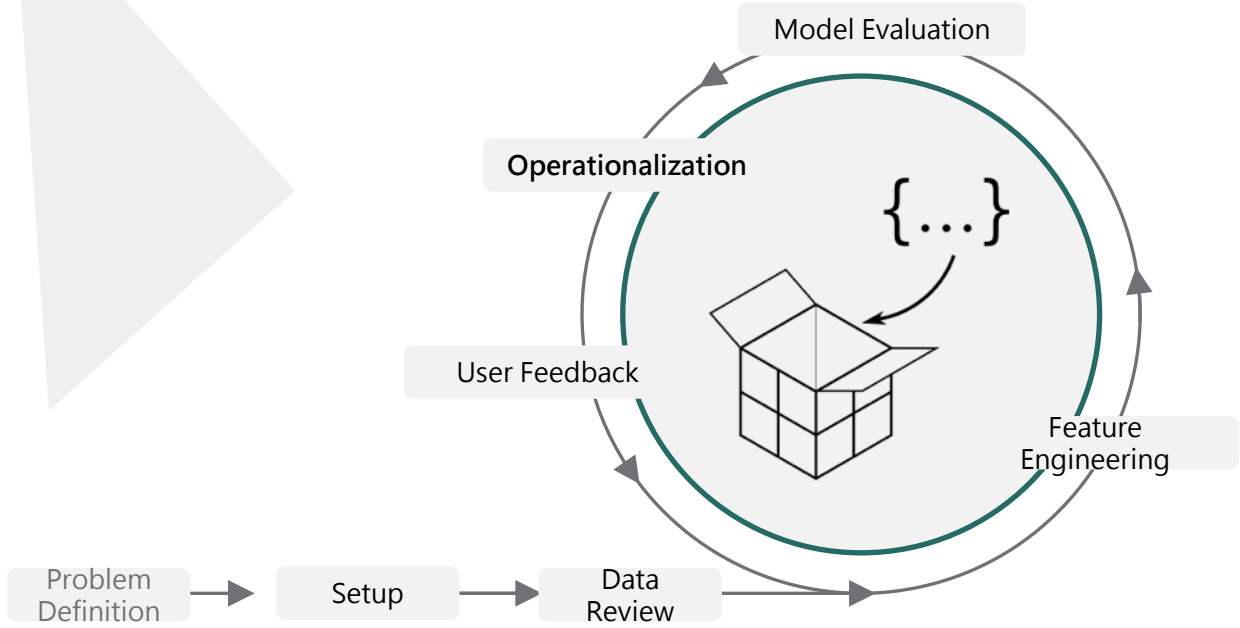
是將資料科學模型部署到生產環境以供其他軟體持續使用的過程



ML-Ops 普遍遭遇的挑戰

Common challenges with model operationalization:

- 掌握產線上的正式數據
- 以擴張規模與強化性能為目的的工程
- 模型的呈送/發佈
- 管理、編排已部署的模型
- 資料科學家非開發人員或平台專家



作業化 ML模型的幾種形式

批次訓練
批次預測/評分

EXAMPLE

Tax Return Fraud (退稅申報詐欺): 為退稅申報資料庫做評分 – 每晚批次作業 – 為可能的欺詐性退稅申報做註記，以提供審計需要

PotsgreSQL/Greenplum with MADlib supports this pattern

批次訓練
事件驅動預測/評分

EXAMPLE

Real Time Transaction Fraud (實時交易詐欺): 從既有歷史資料來源去訓練出一個 ML 分類模型：以判斷 新的一筆借貸交易是否有詐欺的可能性

PostgreSQL/Greenplum with MADlib and RTSMADlib supports this pattern

事件驅動訓練
事件驅動預測/評分

EXAMPLE

Online Advertising (線上廣告): 藉由演算法來即時挑選/測試廣告投放，以期最大化點擊率

Highly specialized – low number of enterprise use cases

ML 模型預測服務的容器化部署 (以 RTSMADlib 為例)

將 Apache MADlib & Greenplum PL/Python 等 ML 工具所實作的工作流程 做 **容器化部署**，為低延遲、事件驅動預測機制奠定基礎

```
$ rtsmadlibflow --deploy --target kubernetes --type madlib-model -- input modelspec.json
```

容器化部署套件的關鍵優勢

- 輕量級 & 簡易部署 (Easy to deploy & light weight)
- 高規模拓展性的 REST & 串流化 (Highly scalable REST and Streaming)
- 端點到端點的ML工作流程 (End-to-end ML workflow)
- 低延遲的推論/預測 (Low latency inference/predictions)
- 特徵轉換 (Feature Transformations)
- Shift & Load 部署模式，無須異動程式碼 (Shift and load deployment pattern, without code change)
- 支持任何風格的 K8s (Any flavor of K8s Supported)
- 一行指令完成部署

自動化部署(ML模型)的作業程序

使用者手動作業

自動化的後端作業程序

Model Training

Greenplum+MADlib 使用特定表格資料、訓練出 ML 模型

Model deploy

將訓練好的模型做適度版控與儲存

Docker pull

Greenplum+PL/Container 可協助模型歸檔、管控模型的訓練作業邏輯與調用程序

Extract Model

依不同使用需求從 Greenplum 擷取相對應 ML模型

Load Model

將模型與作業程序同時加載至容器中，然後進行指定的模型預測程序

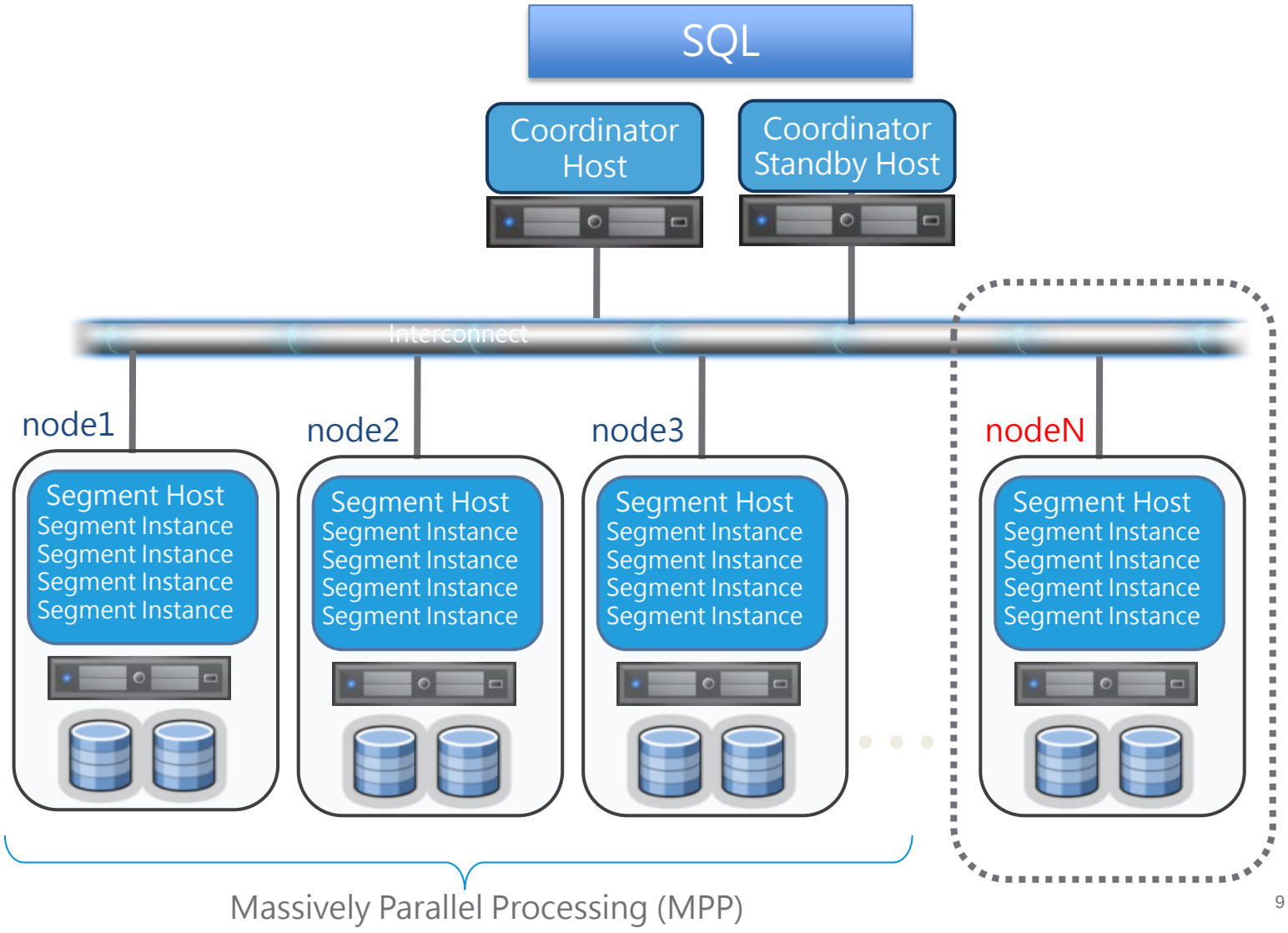
Deploy

將”事件驅動預測”機制作為服務部署於容器中，後續再以REST API 形式提供服務予用戶

Greenplum MPP Shared Nothing Architecture

專為可擴充性和速度性能設計的架構

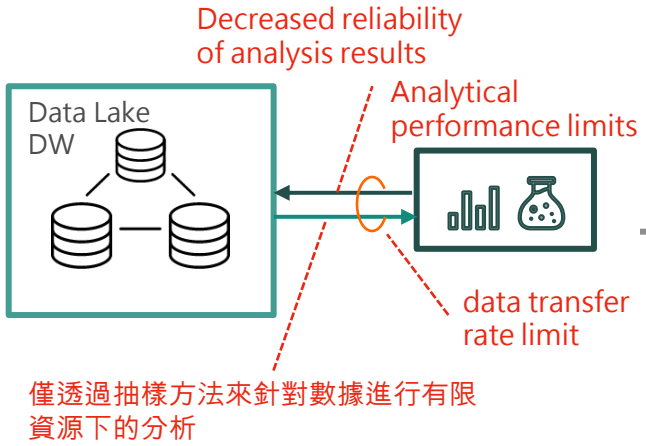
- MPP高併發平行運算架構
- ANSI and PostgreSQL相容SQL
- 運算容量輕易從 terabytes 擴充到 petabytes
- 相較於數據湖 (e.g. Hadoop)，更容易部署整合
- 支援許多資料科學家所需的分析演算法工具
- 支援多種資料格式儲存型態遠端進程存取



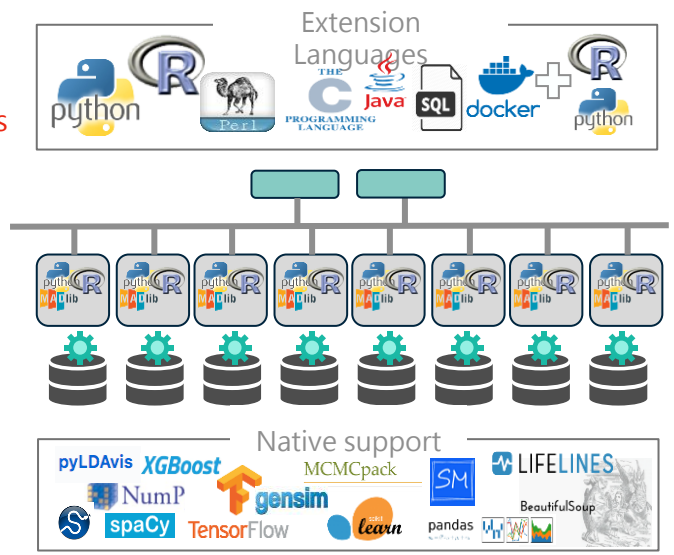
Greenplum 支持 MLOps 的關鍵因素

MADlib 與 In-DB 分析技術 (1)

Typical single-node analysis method



Greenplum's In-DB Analysis Method



- 在不搬動大量數據的前提下，針對所有數據進行分析，而非僅針對抽樣數據進行分析，以提高分析結果的信賴程度
- 高效的平行處理分析演算法

Defect analysis (瑕疵分析使用案例)

Perform all-stage analysis and improve analysis quality
Detect Suspicious Sensors by Correlation/Pattern

Defect analysis case

- 分析量測數據以分辨的統計分析是否呈現偏離現象
- 重新制定新的管理計畫以解決上述偏離問題

Join sensor quality data

Split sensor data by condition

Data normalization

Remove meaningless data conditions

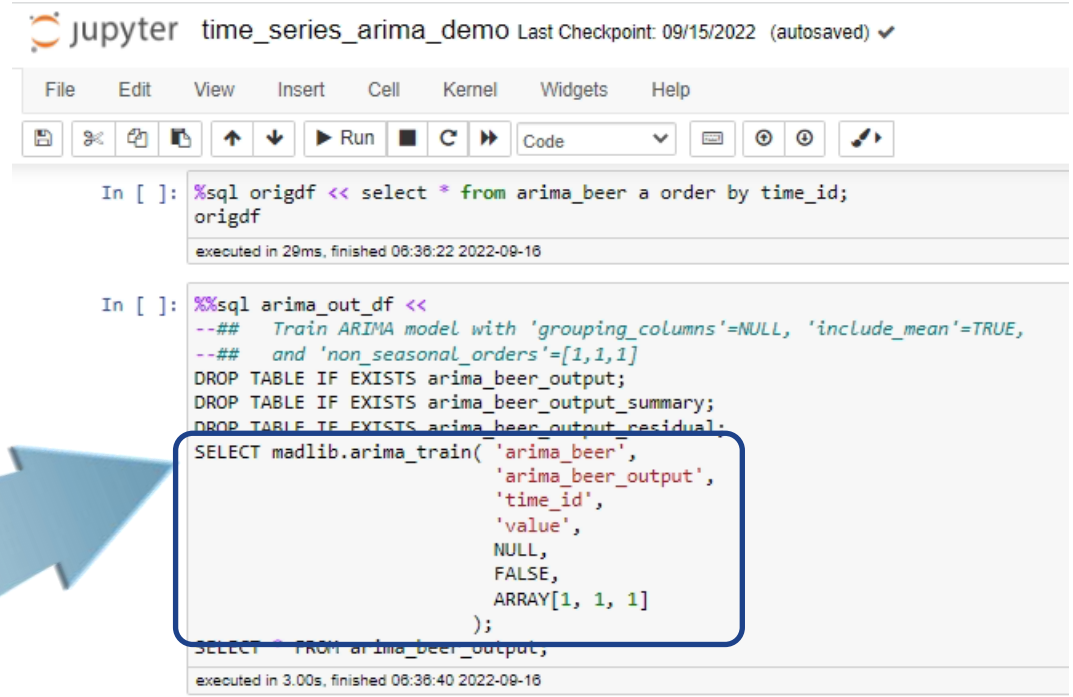
Correlation hypothesis verification, correlation coefficient calculation & sensor selection

Facility Engineer Verification and Action

Greenplum 支持 MLOps 的關鍵因素

MADlib 與 In-DB 分析技術 (2)

MADlib 函式庫分類與功能清單



- 函數的使用、輸入輸出都以資料表的形式存在
- 採用數據庫內分析技術、充分利用海量平行運算 (MPP) 的關鍵優勢，讓分析效率大量提升

Greenplum 支持 MLOps 的關鍵因素

提供撰寫程序式語言的彈性 (1)

Example

PL/R example

```
CREATE OR REPLACE FUNCTION rf_predict_plr
 id int[], y float8[], x1 float8[], x2 float8[])
RETURNS SETOF rf_predict_type AS
$$
  library(randomForest)

  m1<- randomForest(y ~ x1 + x2)
  temp_m1<- data.frame(id, predict(m1))
  return(temp_m1)
$$
LANGUAGE 'plr';
```

R source

```
SELECT gender, UNNEST(id) AS id,
    UNNEST(s_weight_predicted) AS s_weight_predicted
FROM (
    SELECT  gender,
      (rf_predict_plr(id_arr, y_arr, x1_arr, x2_arr)).*
    FROM abalone_array
  ) a
ORDER BY id;
```

Call PL/R function

PL/python example

```
CREATE OR REPLACE FUNCTION rf_predict_plpy
 (id_arr int[], y_arr float8[], x1_arr float8[], x2_arr float8[])
RETURNS rf_predict_type AS
$$
  import numpy as np
  from sklearn.ensemble import RandomForestRegressor
  id = np.array(id_arr).T
  y = np.array([y_arr]).T
  X = np.array([x1_arr, x2_arr]).T
  rf_regr = RandomForestRegressor(max_depth = 2,
                    max_features = "auto",
                    n_estimators = 200,
                    random_state = 1004)
  rf_regr_model = rf_regr.fit(X, y)
  y_pred = rf_regr_model.predict(X)
  return {'id': id, 's_weight_predicted': y_pred}
$$
LANGUAGE 'plpythonu';
```

Python source

```
SELECT gender, UNNEST(id) AS id,
    UNNEST(s_weight_predicted) AS s_weight_predicted
FROM (
    SELECT  gender,
      (rf_predict_plpy(id_arr, y_arr, x1_arr, x2_arr)).*
    FROM abalone_array
  ) a
ORDER BY id;
```

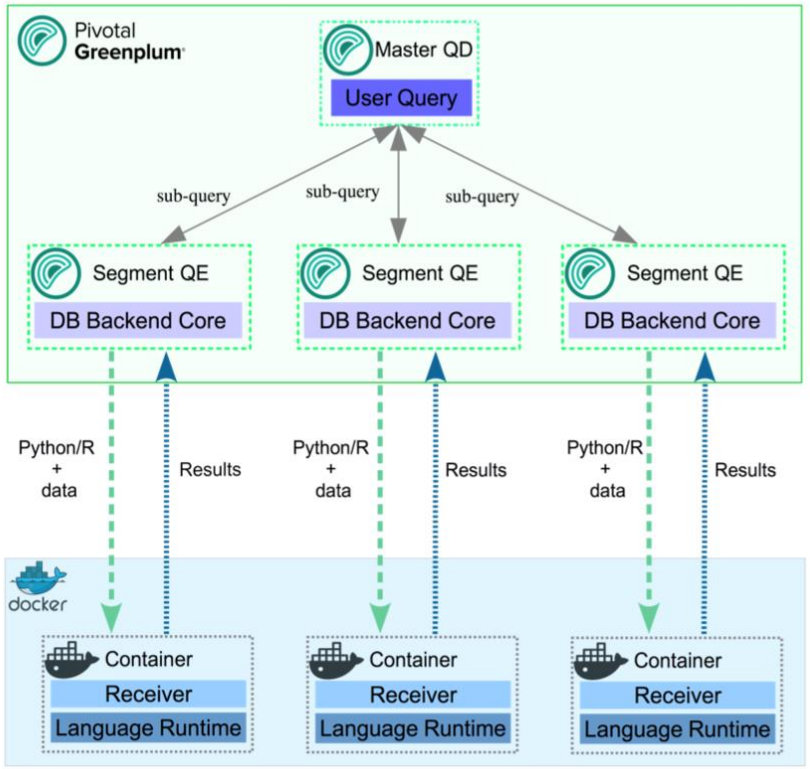
Call PL/Python function

Parallel processing for each data node in Greenplum

Greenplum 支持 ML-Ops 的關鍵因素

提供撰寫程序式語言的彈性 (2)

PL/Container Mechanism



Coding Example by using PL/Container

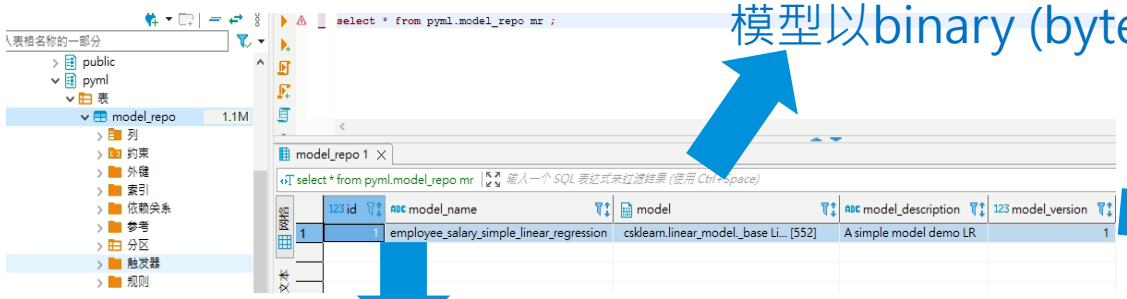
```
In [6]: %%sql

create or replace function plcpymldemo.employee_salary_LR_model() returns bytea as
$$
    # container: plc_python3_shared
    """
     Usage: simple linear regression demo
     """
    import numpy as np
    import pandas as pd
    from pickle import dumps
    from sklearn.linear_model import LinearRegression
    ''' load training data from view '''
    tableData = plpy.execute('select years_of_experience, salary from plcpymldemo.employee_salary_lr_training')
    frame = []
    for rec in tableData:
        frame.append(rec)
    df = pd.DataFrame(frame)
    ''' dependent variable, i.e years_of_experience '''
    x = df.iloc[:, :-1].values
    # independent variable, i.e salary
    y = df.iloc[:, 1].values
    ''' fit model '''
    regressor = LinearRegression()
    regressor.fit(x, y)
    return dumps(regressor)
$$ language plcontainer;
executed in 20ms, finished 14:04:52 2022-10-11

 * postgresql://gpadmin:***@172.18.105.136:5432/fpgdb
Done.

Out[6]: []
```

- 進一步提供撰寫 Python 3.x 程式，使其能於容器中獨立執行的機制
- 雖共享 Greenplum 叢集的運算資源、仍遵循容器環境獨立於Greenplum 的規範，運算邏輯以 Greenplum Function 的形式存在於叢集之中
- 亦持續採用數據庫內分析技術並充分利用海量平行運算(MPP) 的運作機制

Greenplum 支持 MLOps 的關鍵因素

ML 模型的管控機制 (1)

模型的管控



模型以binary (bytea) 形式存在於管控表之中

模型於管控表中也可以有版本與生效期間

模型有名稱也有大綱描述 (description)

藉由既定的作業程序進行模型的存取

```
In [6]: %%sql

create or replace function plcpymldemo.employee_salary_LR_model() returns bytea as
$$
    # container: plc_python3_shared
    """
    Usage: simple linear regression demo
    """
    import numpy as np
    import pandas as pd
    from pickle import dumps
    from sklearn.linear_model import LinearRegression
    ''' load training data from view
    tableData = plpy.execute('select years_of_experience, salary from plcpymldemo.employee_salary_lr_training')
    frame = []
    for rec in tableData:
        frame.append(rec)
    df = pd.DataFrame(frame)
    ''' dependent variable, i.e years_of_experience '''
    x = df.iloc[:, :-1].values
    # independent variable, i.e salary
    y = df.iloc[:, 1].values
    ''' fit model '''
    regressor = LinearRegression()
    regressor.fit(x, y)
    return dumps(regressor)
$$ language plcontainer;
executed in 20ms, finished 14:04:52 2022-10-11
 * postgresql://gpadmin:***@172.18.105.136:5432/fpgdb
Done.

Out[6]: []
```

將產生的模型以BINARY形式回傳

訓練出來的模型透過pickle套件輸出為binary物件

欲使用模型進行預測前，須告知模型名稱與版本

```
In [9]: %sql
CREATE OR REPLACE FUNCTION plcpymldemo.employee_salary_lr_model_driver(model_na
RETURNS void
AS $$
    # container: plc_python3_shared
    from pickle import loads
    import pandas as pd
    import numpy as np

    ## This function is used to run the model by loading it from repositor
    # The input need is the model_name and model_version in the model rep
    and the payload table from where the input to model is read. The ca
    payload in to the table and invoke this function.
    example usage is;
    select plcpymldemo.employee_salary_lr_model_driver('employee_salary
    """
    ' Read model from table and deserialize .....'
    splan = plpy.prepare('SELECT model FROM pyml.model_repo WHERE model_name =
    rv = plpy.execute(splan, [model_name, model_version])
    model = loads(rv[0]['model'])
    iqry  = 'insert into ' + output_table + '(years_of_exp, predicted_salary) v
    iplan = plpy.prepare(iqry, ["float", "float"]);
    ' Read years of experiences from input table .....'
    tableData = plpy.execute('SELECT * FROM %s ;' % (input_table))
```

透過pickle套件將二進位轉換為模型物件

Greenplum 支持 MLOps 的關鍵因素

ML 模型的管控機制 (2)

Json 格式的部署設定檔，決定容器化部署後該服務與 Greenplum 的互動模式 (包括連線機制)

與事件驅動服務搭配

以RTSMADlib 為例

```
$ rtsmadlibflow –action deploy  --target kubernetes --type plpy-model -- inputJson linear_regression.json
```

```
> rts4madlib
No arguments passed!
Usage:->
--------------------------------------------------------------------------------
rts4madlib --name unique_name --type type --action action --target target --inputJson file
 name -> module name
 action -> deploy|undeploy
 type -> flow|madlib-model|plpy-model|feature-engine|featurecache|batch
 target -> docker|kubernetes
 inputJson -> path to input json for model **only if action is deploy**
--------------------------------------------------------------------------------
```

```
 "plpyrest.pydeps": "numpy==1.14.6,scipy==1.4.0,pandas==0.25.3,scikit-learn==0.22",
 "modeldb-datasource.jdbc-url" : "jdbc:postgresql://172.18.105.136:5432/fpgdb",
 "modeldb-datasource.username" : "gpadmin",
 "modeldb-datasource.password" : "                ",
 "plpyrest.modelreposchema" : "pyml",
 "plpyrest.modelrepotable" : "model_repo",
 "plpyrest.modelname" : "employee_salary_simple_linear_regression",
 "plpyrest.modelversion" : 1,
 "plpyrest.modeldescription" : "linear regression model with 1 dependent variable demo",
 "plpyrest.payloadtable" : "employee_salary_predict_model_input",
 "plpyrest.resultstable" : "employee_salary_predict_model_output",
 "plpyrest.modelschema" : "plcpymldemo",
 "plpyrest.modeldriverfunction" : "employee_salary_lr_model_driver",
 "plpyrest.modelquery" : "select plcpymldemo.employee_salary_lr_model_driver('employee_salary_s
imple_linear_regression', 1, 'plcpymldemo.employee_salary_predict_model_input', 'plcpymldemo.em
ployee_salary_predict_model_output')"
}
~
~
~
"linear_regression.json" 17L, 1002C written                          17,1          Bot
```

事件驅動的模型預測

```
$ curl -v -H "Content-Type:application/json" http://192.168.99.100:30123/actuator/info
```

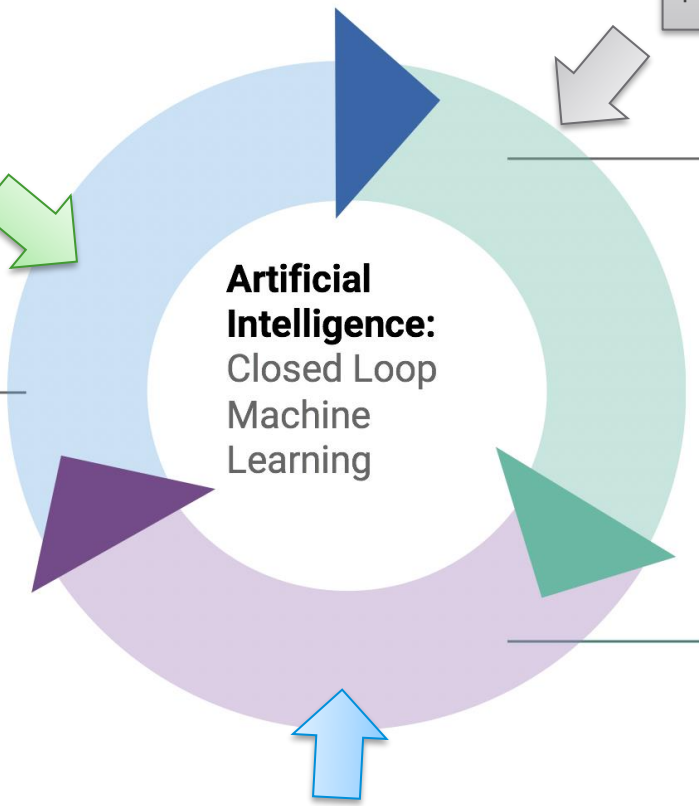
REST 服務為其中一種以’事件觸發’要求GP提供預測服務的形式，可進一步與串流服務搭配，提供快速、大量的預測需求

事件驅動 (Event Driven)與預測/評分服務容器化

模型在大數據環境與MLOps持續運轉下逐漸優化

大數據平台提供海量數據做實驗，更易驗證模型效度，並保證分析功能的擴充性

大數據平台提供ML模型管控機制、提供線上即時預測能力、並容許高負載、高併發(concurrent) 形式運作



Modeling at Scale

Heavy compute tasks such as model training across big data

Pivotal Greenplum

MADlib PostGIS python R TensorFlow

Experimentation

Initial code development and testing, model experimentation on samples.

PostgreSQL

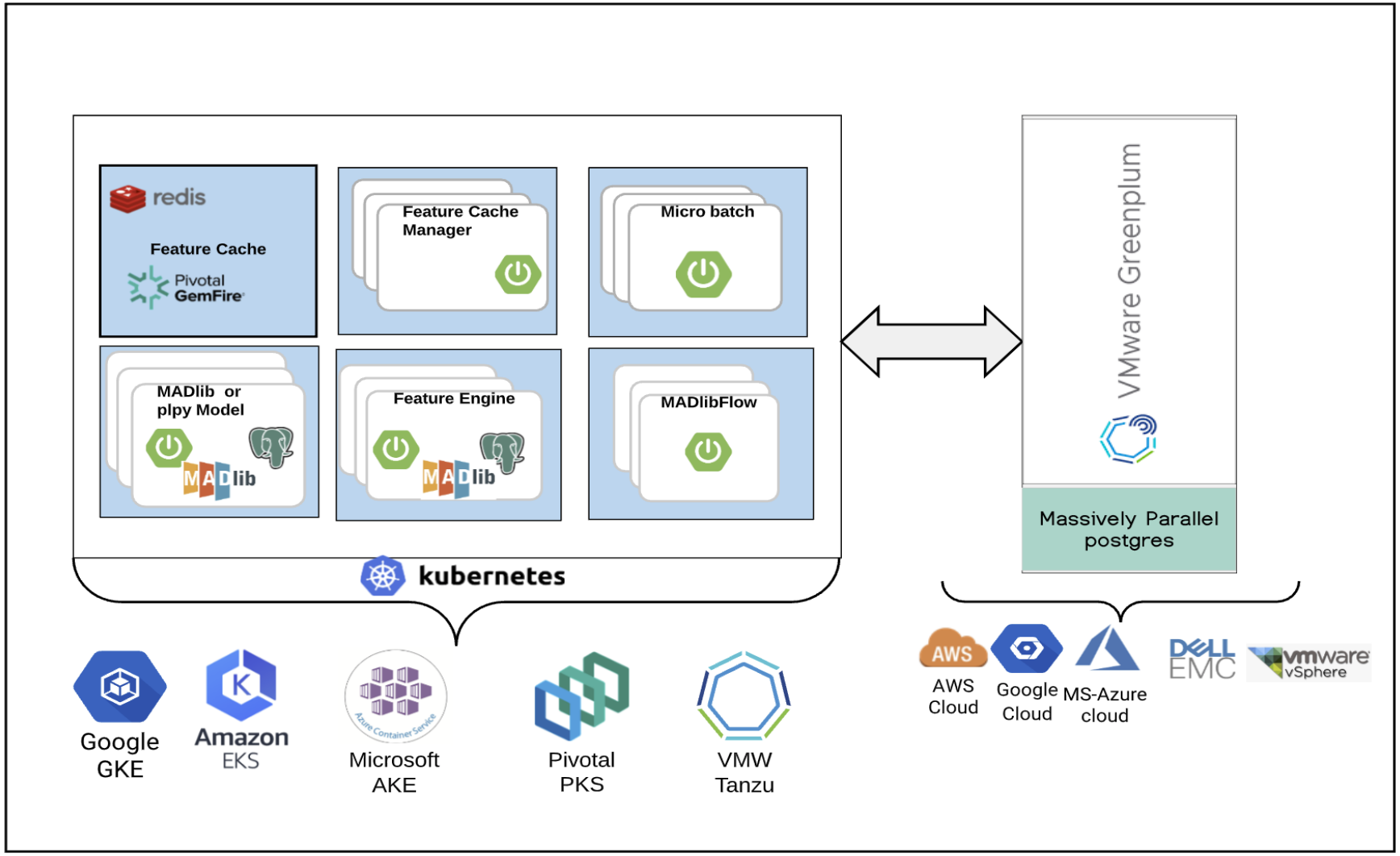
MADlib PostGIS python R TensorFlow

Deployment

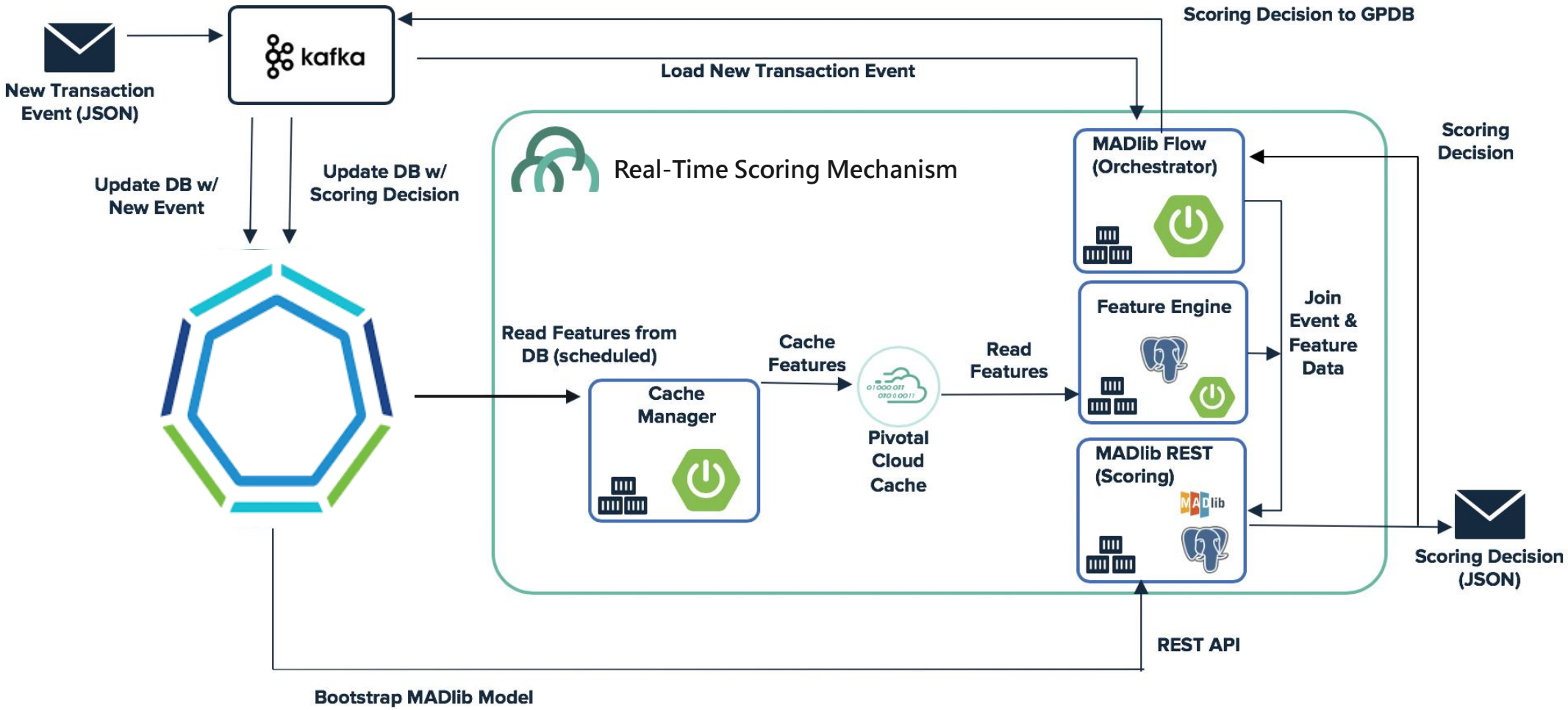
Production deployment of models to feed downstream applications and reports

持續為預測結果與事實比較，提供反饋以決定是否進行模型重新訓練

事件驅動評分(RTS)服務之相關元件

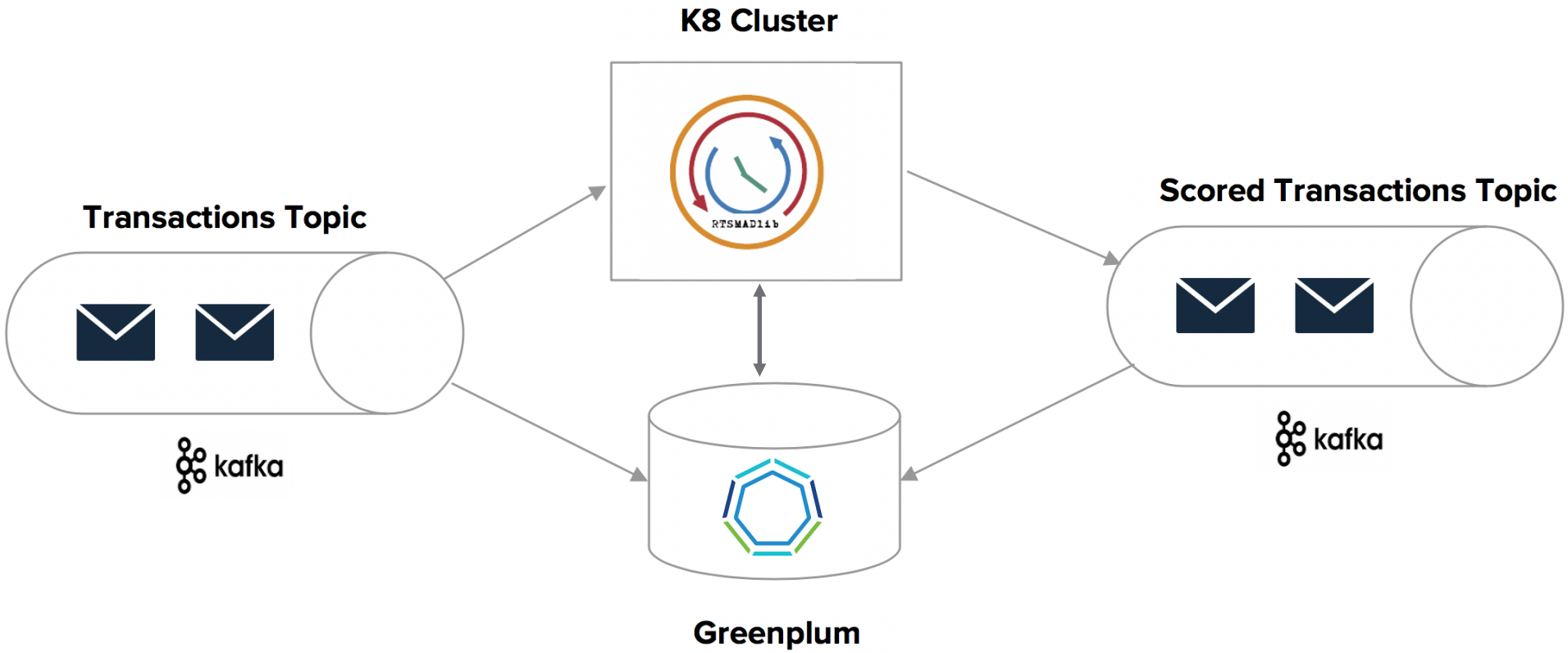


事件驅動評分用例 (配合串流服務)



Demo

Case Study: “信用卡詐欺交易檢測” 資料流



Demo

Case Study: “信用卡詐欺交易檢測” 資料流

1. 信用卡詐欺檢測模型 建置在 Greenplum
2. 信用卡詐欺模型部署 (MADlib 流程)，包括：
    i. 模型
    ii. 特徵引擎
    iii. 特徵快取 *(refreshable via REST)*
3. Kafka 為事件驅動評分的串流處理
    i. One Kafka producer
    ii. One Kafka streams consumer

References

1. Greenplum
2. PL/Container
3. Apache MADlib
4. MADlib Machine Learning model Jupyter notebooks
5. Real Time Scoring for MADLIB Github
6. RTSMADlib Jupyter notebook examples

感謝聆聽 敬請指教