



# MLOps 再入門！Vertex AI で広がる MLOps の世界と実践例

小野 友也

Google Cloud

カスタマー エンジニア

春日 瑛

株式会社ブレイド

Lead Analytics Engineer



# MLOps & Vertex AI

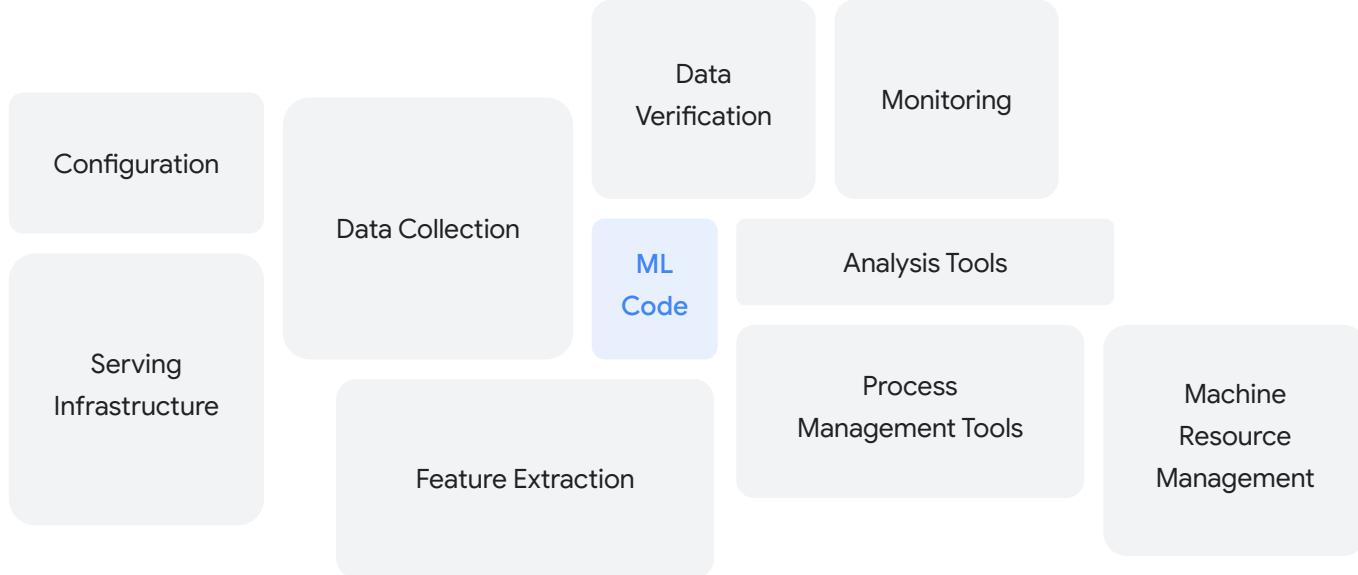
# スピーカー自己紹介



小野 友也

Google Cloud  
カスタマーエンジニア

Google Cloud のカスタマー エンジニア。大手 Sler で証券会社向けのアプリケーション開発の現場にて提案、開発～リリース、運用まで幅広く経験を積む。その後、インフラエンジニアとしてプライベートクラウドの構築～運用、サービス開発を経験。2019 年より現職。新しいテクノロジーを活用できる機会を常に楽しみにしている。岡山県倉敷市出身。趣味はテニスと囲碁。



# 機械学習の実運用における課題

## 機械学習の課題

- データ、特徴量、モデル、
- パイプライン、実験等のガバナンス
- CI/CD
- Training-serving skew
- データ検証とモデル分析
- ML 公平性と説明可能性

## DevOps の課題

- スケーラビリティと可用性
- ポータビリティ
- 再現可能性
- モジュール性
- 監視とアラート
- セキュリティ
- サーバレス対応



# MLOps

---

機械学習システムの開発と運用を  
統合するプラクティス

**Vertex AI** は  
機械学習システムの  
開発から実運用までを支える  
統合プラットフォーム



Vertex AI



# Vertex AI

## Applications

Vision and Video

Conversation

Language

Structured Data

## Custom machine learning

Workbench

AutoML

NAS

Prediction

ML Metadata

Data Labeling

Training

Explainable AI

Feature Store

Model Monitoring

Experiments

Vizier (Optimization)

Pipelines

Matching Engine



# Vertex AI

## Applications

Vision and Video

Conversation

Language

Structured Data

## Custom machine learning

Workbench

AutoML

NAS

Prediction

ML Metadata

Data Labeling

Training

Explainable AI

Feature Store

Model Monitoring

Experiments

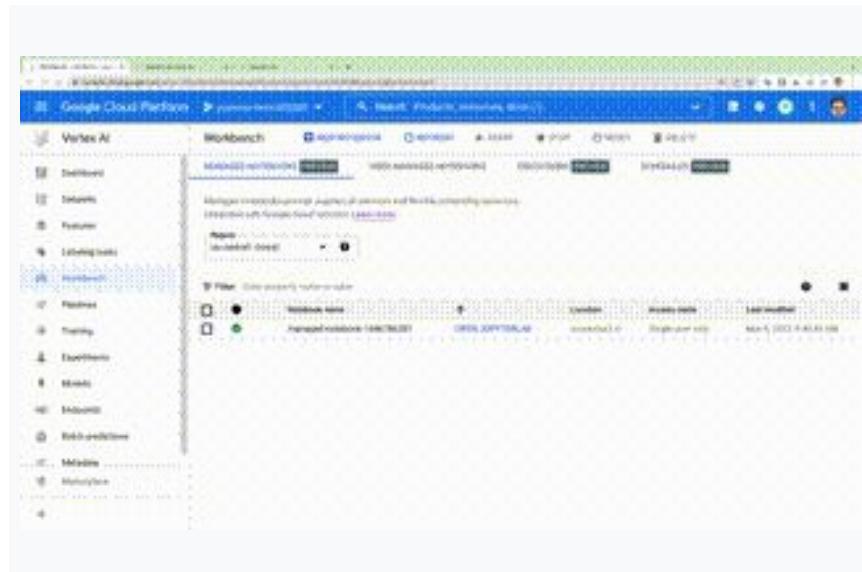
Vizier (Optimization)

Pipelines

Matching Engine

# Vertex AI Workbench PREVIEW

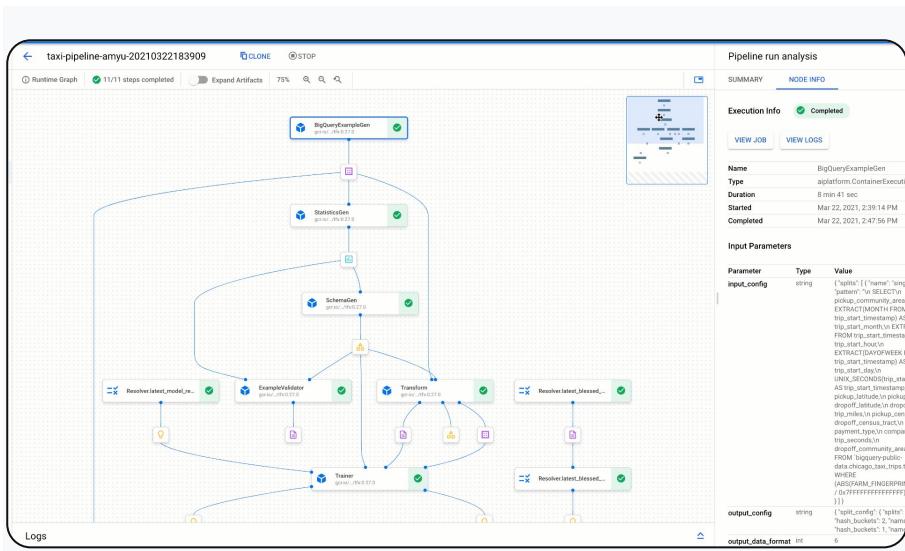
- データ分析、マシンラーニングのワークフロー向けのノートブック インターフェース
  - 高速なプロトタイプとモデル開発
  - マネージド or ユーザ管理
  - より簡単なデータ探索と分析
    - BigQuery、Cloud Storage や Spark 環境とシームレスに連携



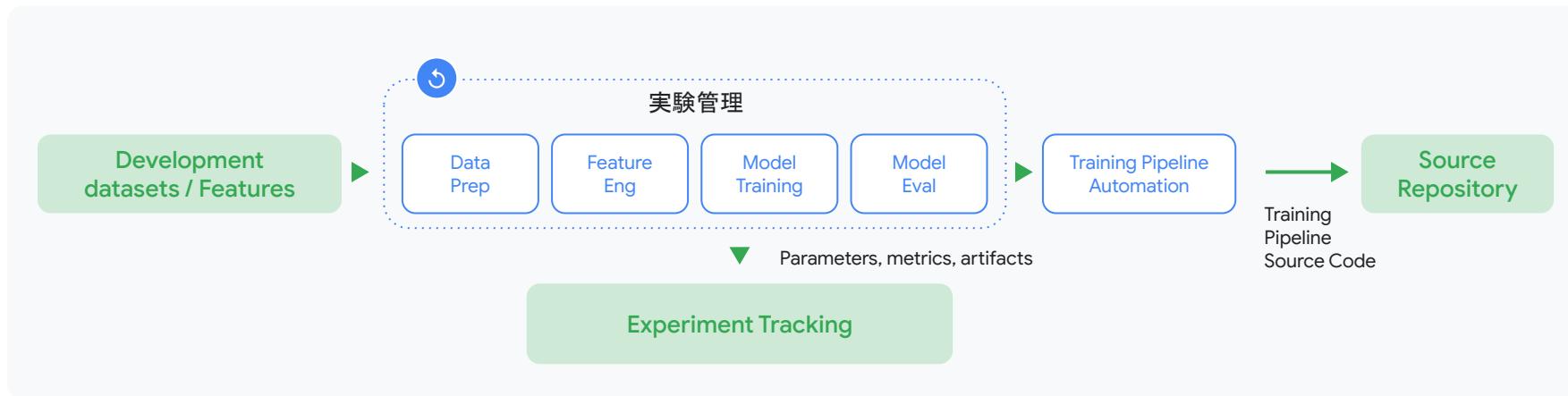
\* managed notebooks instance は Preview です

# Vertex AI Pipelines

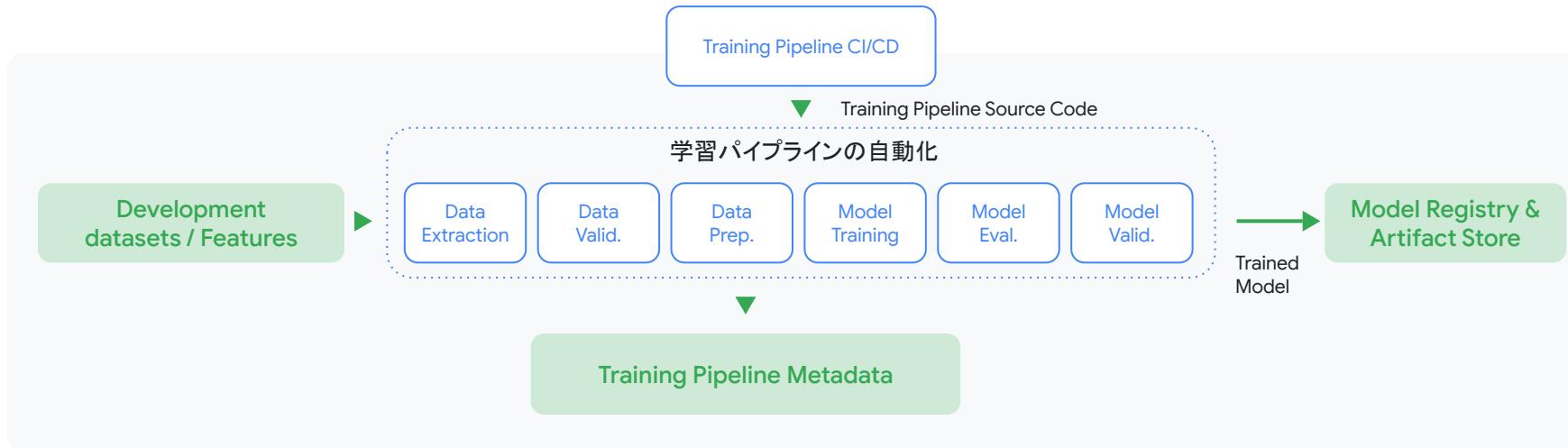
- ML パイプラインの サーバレス 実行環境
  - Kubeflow Pipelines
  - TensorFlow Extended (TFX)
- 再現可能性、監査、ガバナンス
  - データ、特徴量、モデル、パイプライン、実験結果
- 様々な ML フレームワークに対応
  - TensorFlow, Pytorch, scikit-learn, XGBoost など



# Vertex AI Pipelines による実験管理



# Vertex AI Pipelines による継続学習



# スピーカー自己紹介



春日 瑛

株式会社ブレイド  
Lead Analytics Engineer

大学・大学院ではデータマイニングの研究に従事。2017年に株式会社ディー・エヌ・エーにAIエンジニアとして新卒入社し、タクシーの需要供給予測のアルゴリズム開発等に従事。

2019年に株式会社ブレイドに入社。KARTEで収集した膨大なユーザー行動データに統計や機械学習を活用し新たなインサイトを得る分析プロダクトの開発等を担当。

人間工学準専門家としてデスクワーク環境の改善を行う活動もしている。

# Agenda

- KARTEについて
- “Light” MLOps
- ワークフロー
- 今後の展望
- まとめ



# KARTEについて

# データによって 人の価値を最大化する

プレイドは、インターネットで欠如している  
ユーザデータを蓄積するミドルウェアとなり、  
人の価値を最大化するためのサービスを提供します。

株式会社プレイド

東京都中央区銀座 6-10-1 GINZA SIX 10F

設立: 2011年10月

従業員: 240名(2021年12月時点)

2011年10月 創業

2015年3月 CXプラットフォーム「KARTE」正式版を提供開始

2018年3月 「KARTE for App」を提供開始

2018年4月 既存投資家を中心とした社から総額27億円の資金調達を実施

2018年10月 「2018年日本テクノロジーFast 50」で3位受賞

2019年11月 Googleからの資金調達を実施

2020年7月 「KARTE Blocks」を提供開始

2020年12月 東証マザーズへ新規上場



# Customer Experience Platform

karte.io

The screenshot displays a dashboard with four main sections:

- Top Left:** A user profile with a gray circular icon, showing a notification count of 71. Text: "Facebook広告から流入しました たった今" and "Facebookから流入 キャンペーン広告を見た 3回以上購入、30日以上購入" (Facebook ad from flow, just now; viewed campaign ad, purchased 3+ times, 30+ days since purchase).
- Top Right:** A callout bubble stating: "Webサイトの訪問者の行動を顧客ごとにリアルタイムに解析" (Analyze website visitors' behavior in real-time for each customer).
- Middle Left:** Another user profile with a red circular icon, showing 94 notifications. Text: "商品詳細ページを閲覧しています たった今" and "ichiro.tanaka@example.com じっくり検討中 10分以上滞在しているお客様" (Viewing product detail page, just now; ichiro.tanaka@example.com, reviewing carefully, stayed over 10 minutes, customer).
- Middle Right:** A third user profile with a gray circular icon, showing 64 notifications. Text: "新着アイテムのページを閲覧しています たった今" and "木村一郎 ガジェット系商品をよく見る お気に入りから流入 大学生" (Viewing new item page, just now; Ichiro Kimura, frequently views gadget products, from favorite, university student).
- Bottom Left:** A modal window titled "よくあるご質問" (FAQ) with a question mark icon. It says: "vis-12345" and "よくあるご質問". Below it is a large orange button with white text: "一人一人に合わせた顧客体験を提供" (Provide personalized customer experiences for each individual).
- Bottom Right:** Two stacked cards:
  - Top Card:** "絶対、欲しい!" (Definitely want!) with a "CHECK" button. Text: "ずっと使えるアイテムだけを揃えました。この機会をお見逃しなく!" (We have gathered only items that last. Don't miss this opportunity!).
  - Bottom Card:** "ご希望の物件は見つかりませんでしたか?" (Did you find the property you were looking for?). Text: "検索条件を少し変更するだけで、あなたの理想的な物件に出会える可能性が高くなります。条件を変えて検索してみてください。"



---

**Customer  
Experience Platform**

karte.io

145 億 UU

累計ユーザー数 ※1

2.43 兆円

年間解析流通金額 ※2

135,000 over

秒間トラッキング数 ※3

0.x 秒/解析

解析速度

180 + PB

月間解析データ量

8 + PB

蓄積データ量

※1 ローンチ～2022年3月までの解析ユニークユーザー数の実績

※2 EC領域における解析流通金額。2021年3月～2022年2月までの  
単年の実績

※3 秒間解析イベント数

(閲覧、購入、クリックなど全計測イベントが対象。2022年3月最大値)

# KARTE 導入企業様

※2021年9月27日時点(順不同/一部抜粋)

## EC / 総合通販

AEON.com AlpenGroup BAYCREW'S STORE

ENOTECA GDO KOMEHYO

LUXA PAL CLOSET  
ONLINE STORE

QVC W THE SUIT COMPANY ZOZOTOWN  
三井ショッピングパーク  
& mall モノタロウ 大丸松坂屋百貨店

## 金融 / 保険

SMBC 三井住友銀行 MUFG 三菱UFJ銀行

MIZUHO SAISON POINT MALL  
セイゾンポイント

SBI SBI証券 セブン銀行

LiFENET WealthNavi  
イーデザイン損保 三井ダイレクト損保  
MSIG INSURANCE GROUP

## IT / サービス

AllAbout bellFace

:DeNA freee

NRI RECRUIT

Rentio SmartHR  
STORES Y!mobile

## コンテンツ

DELISH KITCHEN

Rakuten TV

サンデーラスボリ

ゼンジヤンゴ+  
テレ東BIZ

## メーカー

Canon

KIRIN

Nestle.  
Good food, Good life.

H House VOLVO

## 人材

BIZREACH en  
Wantedly リクナビ

## 旅行 / 交通

GO HIS  
JB JAL  
旅の予定、いつも。

## 美容 / ヘルスケア

@cosme SHOPPING KOSE  
POLA エステティック TBC

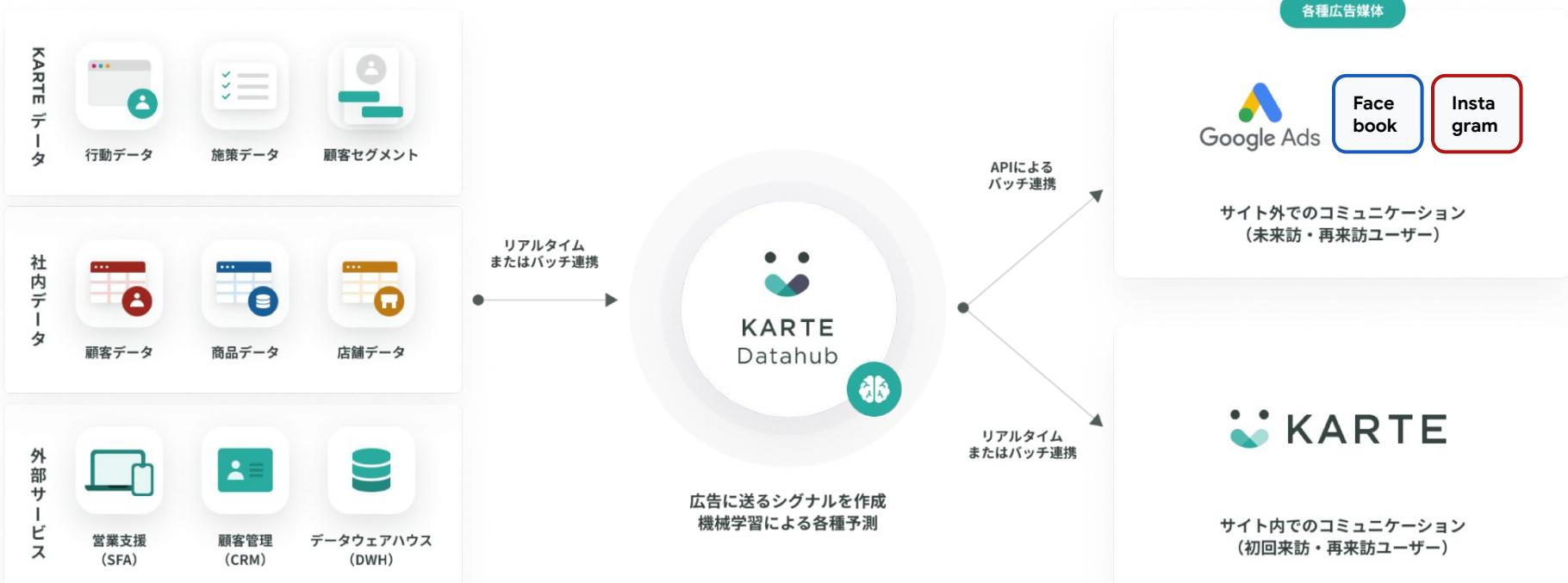
## 不動産 / 住宅情報

OPEN HOUSE LIFULL HOME'S  
三井不動産 三菱地所

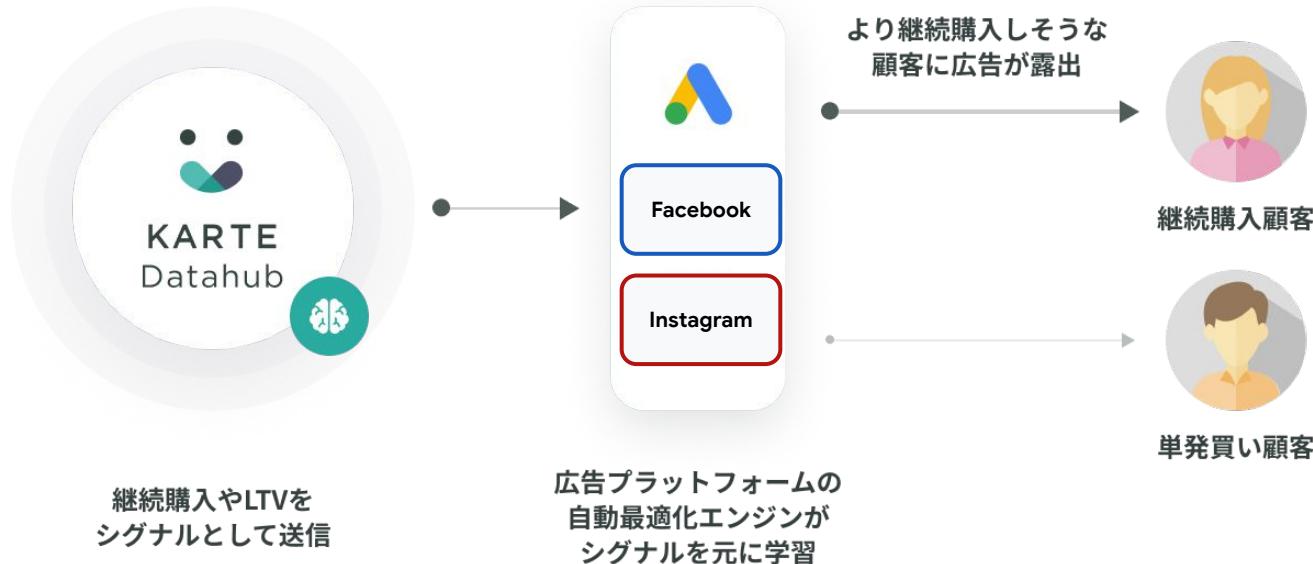
## 教育 / 学習

京都芸術大学  
通信教育課程  
スタディサプリ LEC 東京リーガルマインド

# KARTE Signals



# KARTE Signals



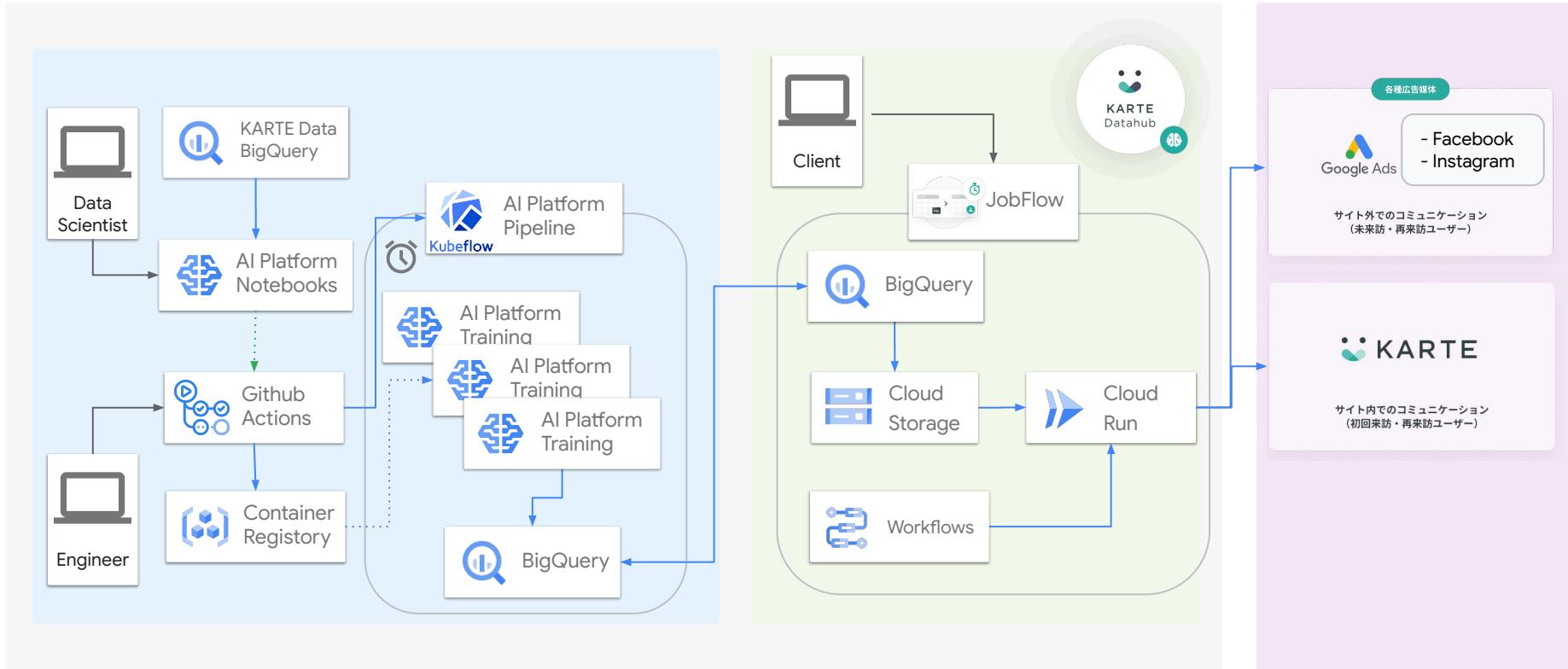


# “Light” MLOps

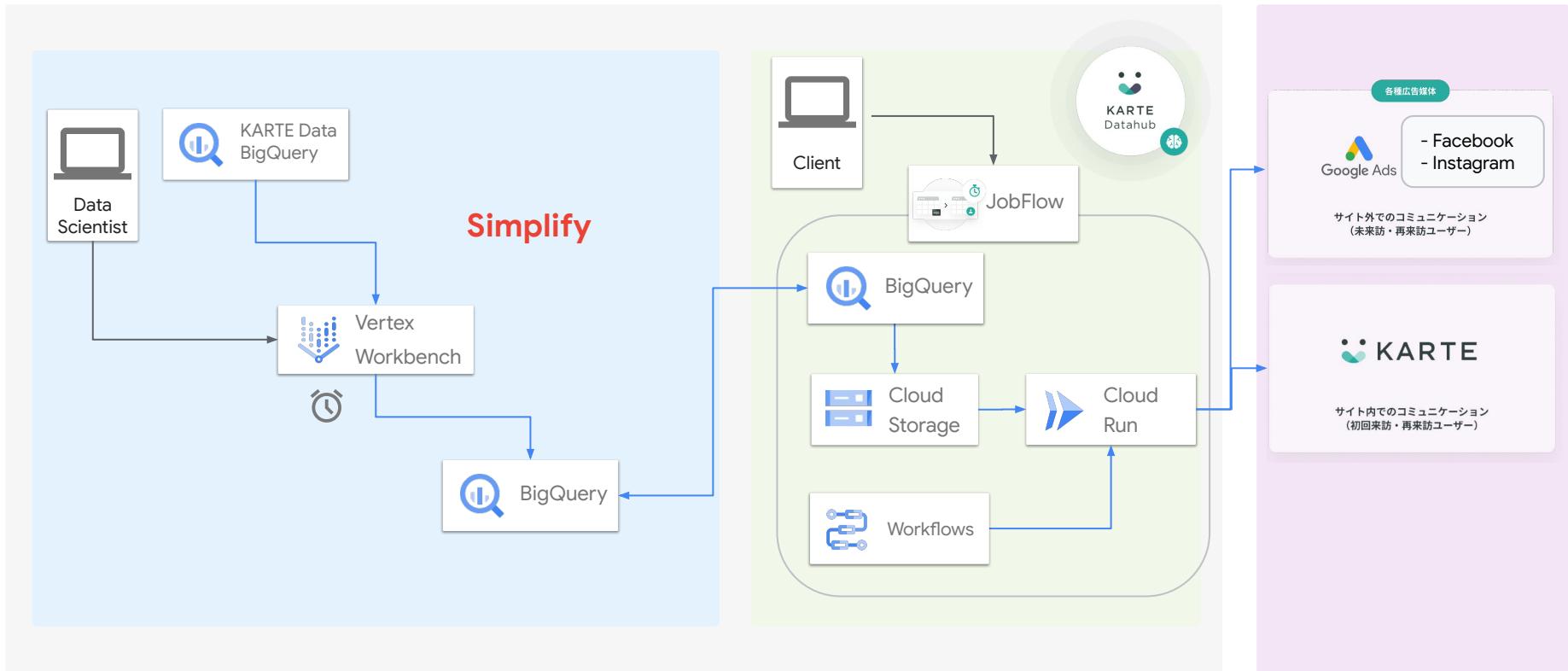
# MLOps vs “Light” MLOps

	MLOps	“Light” MLOps
エンジニア工数の削減	△	○
モデル開発集中度	△	○
複雑なワークフロー	○	△
Github 管理	○	△

# MLOps Architecture



# “Light” MLOps Architecture





# ワークフロー

# Data Scientist のワークフロー

## Notebook

Notebook で分析及び機械学習  
モデルを作成

## Schedule Execution

Notebook を定期実行する

## External Table Link

必要に応じて KARTE または  
外部のテーブルと連携する



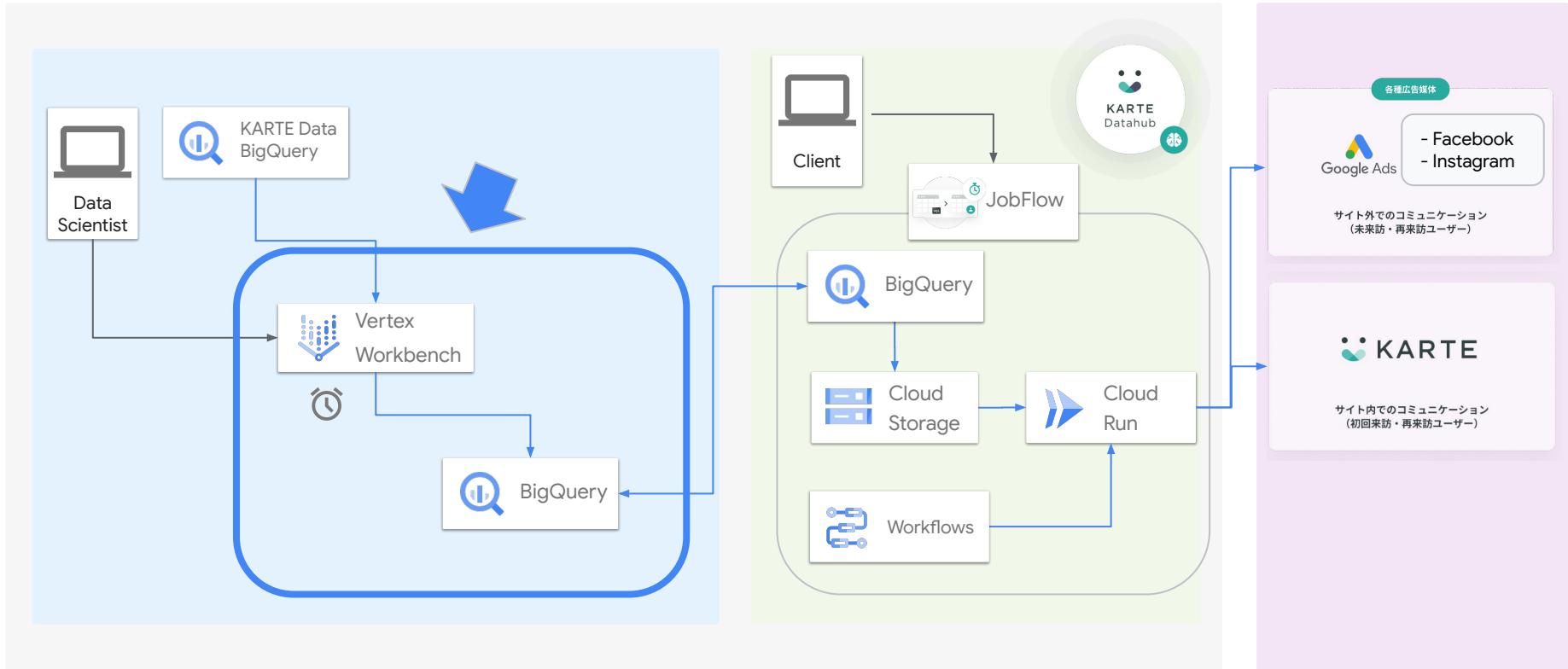
## Parameterized

変数を Parameter にして汎用化  
する

## Output Table

出力テーブルが定期更新されて  
いるか確認する

# “Light” MLOps Architecture



The screenshot shows a Jupyter Notebook interface with several code cells:

- Install**:

```
[1]: ! pip install category_encoders lightgbm -q
```
- Parameters**:

```
[2]: PROJECT = "dev-karte-ml"  
API_KEY = "demo"  
DATA_DAY_RANGE = 60  
FEATURE_DAY_RANGE = 7  
LABEL_DAY_RANGE = 45  
LABEL_BUY_COUNT = 3
```
- Preprocess**:

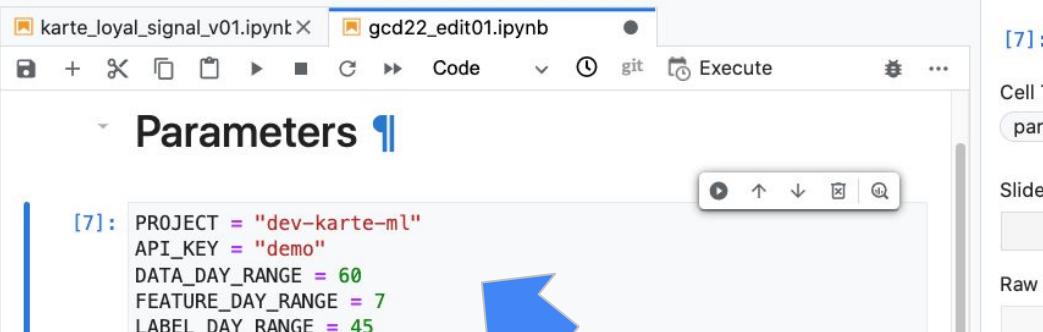
```
[3]: from google.cloud import bigquery  
from datetime import datetime, timedelta  
  
[4]: client = bigquery.Client(PROJECT)  
END_DATE = datetime.now()  
START_DATE = END_DATE - timedelta(days=DATA_DAY_RANGE)  
  
[5]: query = """  
WITH  
events AS (  
SELECT  
sync_date,  
session_id,  
is_new_session,  
user_id,  
visitor_id,  
event_name,  
view,  
buy,  
item,  
dimensions  
FROM  
`karte-data.karte_stream_{API_KEY}.krt_pockyevent_v1_*`  
WHERE  
_TABLE_SUFFIX BETWEEN FORMAT_TIMESTAMP('%Y%m%d',TIMESTAMP('{START_DATE}'))  
AND FORMAT_TIMESTAMP('%Y%m%d',TIMESTAMP('{END_DATE}'))  
AND event_name IN ("view",  
"buy",
```
- Train**:

```
[ ]:
```

Below the Preprocess section, there is a placeholder text: **Train, Inference, PostProcess ... etc**.

# Notebook

Componentごとに処理を構築し、  
Notebookを作成します



```
[7]: PROJECT = "dev-karte-ml"
API_KEY = "demo"
DATA_DAY_RANGE = 60
FEATURE_DAY_RANGE = 7
LABEL_DAY_RANGE = 45
LABEL_BUY_COUNT = 3
```

```
[8]: DATA_DAY_RANGE = int(DATA_DAY_RANGE)
FEATURE_DAY_RANGE = int(FEATURE_DAY_RANGE)
LABEL_DAY_RANGE = int(LABEL_DAY_RANGE)
LABEL_BUY_COUNT = int(LABEL_BUY_COUNT)
```

## Preprocess

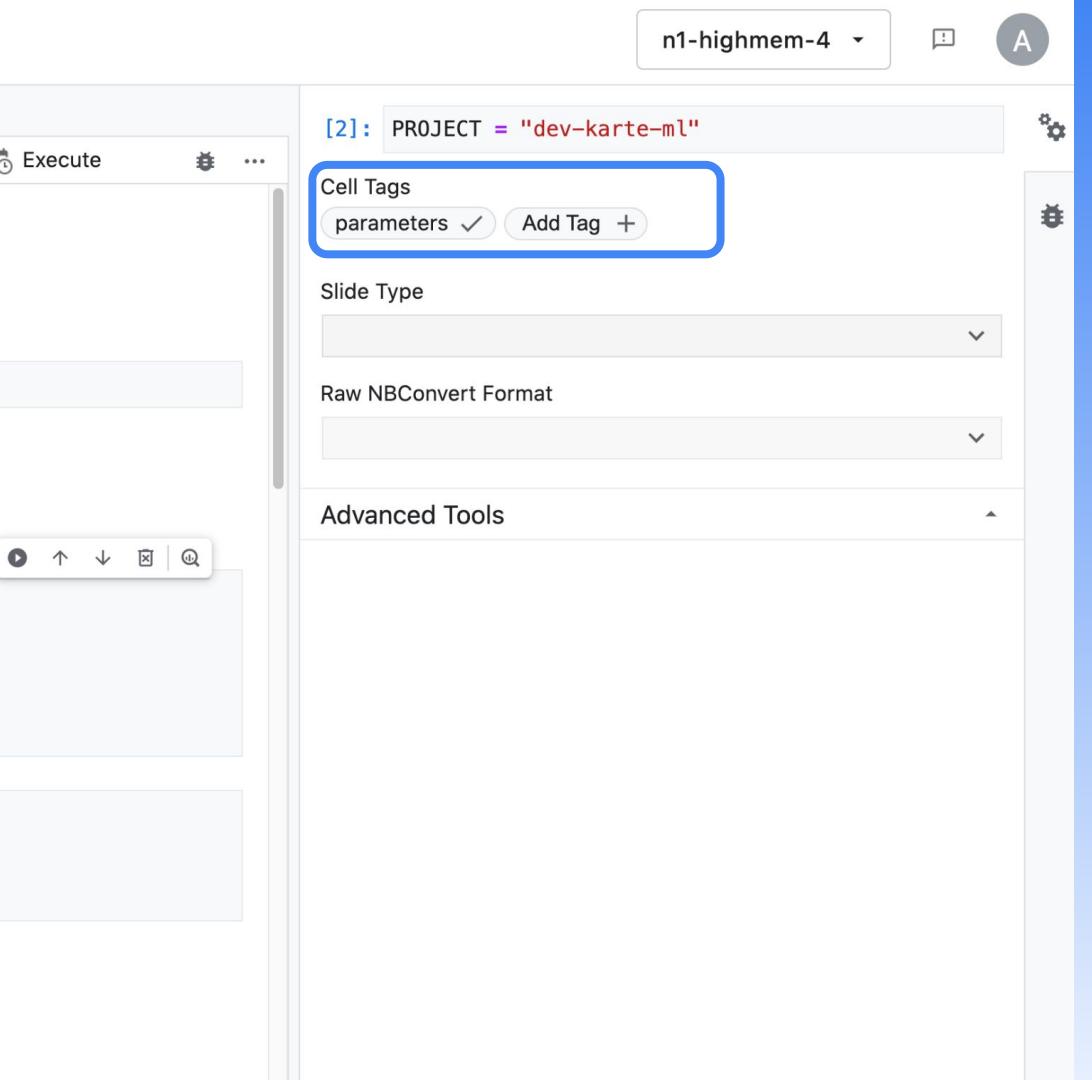
```
[3]: from google.cloud import bigquery
from datetime import datetime, timedelta
```

```
[4]: client = bigquery.Client(PROJECT)
END_DATE = datetime.now()
START_DATE = END_DATE - timedelta(days=DATA_DAY_RANGE)
```

```
[5]: query = """
WITH
    events AS (
        SELECT
            sync_date,
            session_id,
            is_new_session
```

# Pamameterized

Parameters を設定したい Cell を選択します



# Pamameterized

Cell に「Parameters」タグを付与することで、Execution する際に変更可能な Parameters を設定します

Settings Help

l\_signal\_v01.ipynb ×

gcd22\_edit01.ipynb

X



## Install

```
! pip install category_encoders lightgbm -q
```

## Parameters



```
PROJECT = "dev-karte-ml"  
API_KEY = "demo"  
DATA_DAY_RANGE = 60  
FEATURE_DAY_RANGE = 7  
LABEL_DAY_RANGE = 45  
LABEL_BUY_COUNT = 3
```

```
DATA_DAY_RANGE = int(DATA_DAY_RANGE)  
FEATURE_DAY_RANGE = int(FEATURE_DAY_RANGE)
```

# Schedule Execution

Execute ボタンをクリックします

Submit notebooks to Executor PREVIEW :

Create a one-time or recurring notebook execution using Vertex AI Training. Results are stored in a sharable Cloud Storage bucket. Charges apply for training and storage. [Learn more](#)

Notebook: signals\_prediction/gcd22\_edit01.ipynb

Schedule name: gcd22\_edit01\_1645687250618 (Up to 128 lowercase letters, numbers, or underscores.)

Cloud Storage bucket: signals\_prediction\_mock (Where results are stored. Select an existing bucket or create a new one.)

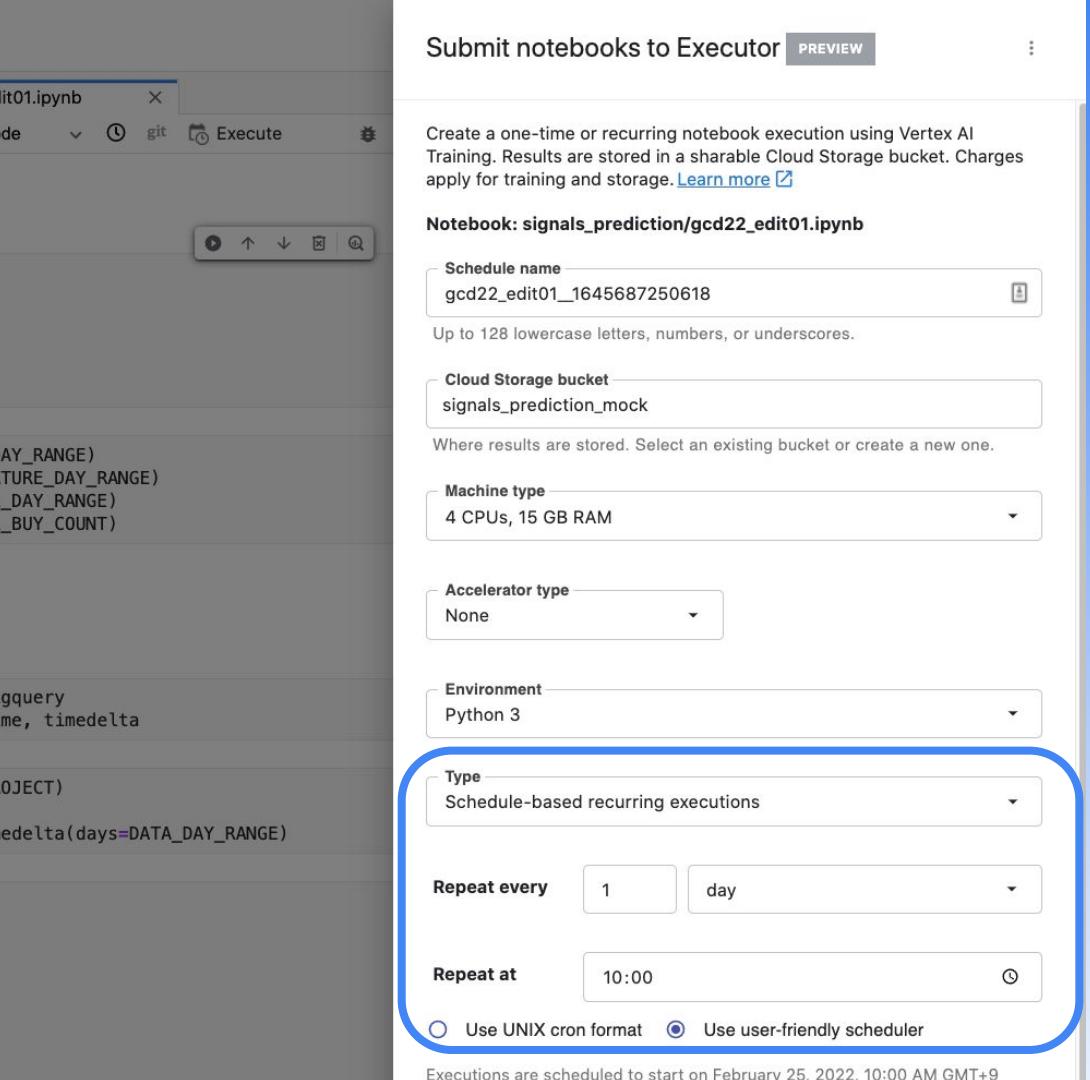
Machine type: 4 CPUs, 15 GB RAM

Accelerator type: None

Environment: Python 3

Type: Schedule-based recurring executions (Repeat every 1 day at 10:00. Use user-friendly scheduler)

Executions are scheduled to start on February 25, 2022, 10:00 AM GMT+9



# Schedule Execution

Schedule Execution を設定します

DATA\_DAY\_RANGE  
(FEATURE\_DAY\_RANGE)  
ABEL\_DAY\_RANGE  
ABEL\_BUY\_COUNT

t bigquery  
timestime, timedelta  
  
t (PROJECT)  
(  
timedelta(days=DATA\_DAY\_RANGE)

stream\_{API\_KEY}.krt\_pockyevent\_v1\_\*  
  
EN FORMAT\_TIMESTAMP('%Y%m%d',TIMESTAMP('{{P('Y%m%d',TIMESTAMP('{{END\_DATE}}'))  
"view",

Machine type  
4 CPUs, 15 GB RAM

Accelerator type  
None

Environment  
Python 3

Type  
Schedule-based recurring executions

Repeat every  
1 day

Repeat at  
10:00 ⏺  
 Use UNIX cron format  Use user-friendly scheduler

Executions are scheduled to start on February 25, 2022, 10:00 AM GMT+9

~ ADVANCED OPTIONS

Notebook parameterization

Input parameters (optional)  
API\_KEY=gcd22,DATA\_DAY\_RANGE=60,FEATURE\_DAY\_RANGE=7,ABEL\_DAY\_RANGE=45,LABEL\_BUY\_COUNT=3

Each parameter needs to be separated by commas (Example:a=x,b=y)

**SUBMIT** CANCEL



# Schedule Execution

最後に Submit します

File

Notebook Executor PREVIEW

Executions Schedules

Search

init\_dataset\_and\_table\_

Frequency: Every day at 9:00 AM GMT+9  
Latest execution: Mar 8, 2022, 9:00 AM

[VIEW LATEST EXECUTION RESULT](#)

karte\_loyal\_signal\_v01\_

Frequency: Every day at 10:00 AM GMT+9  
Latest execution: Mar 8, 2022, 10:00 AM

[VIEW LATEST EXECUTION RESULT](#)

init\_dataset\_and\_table\_v01\_test

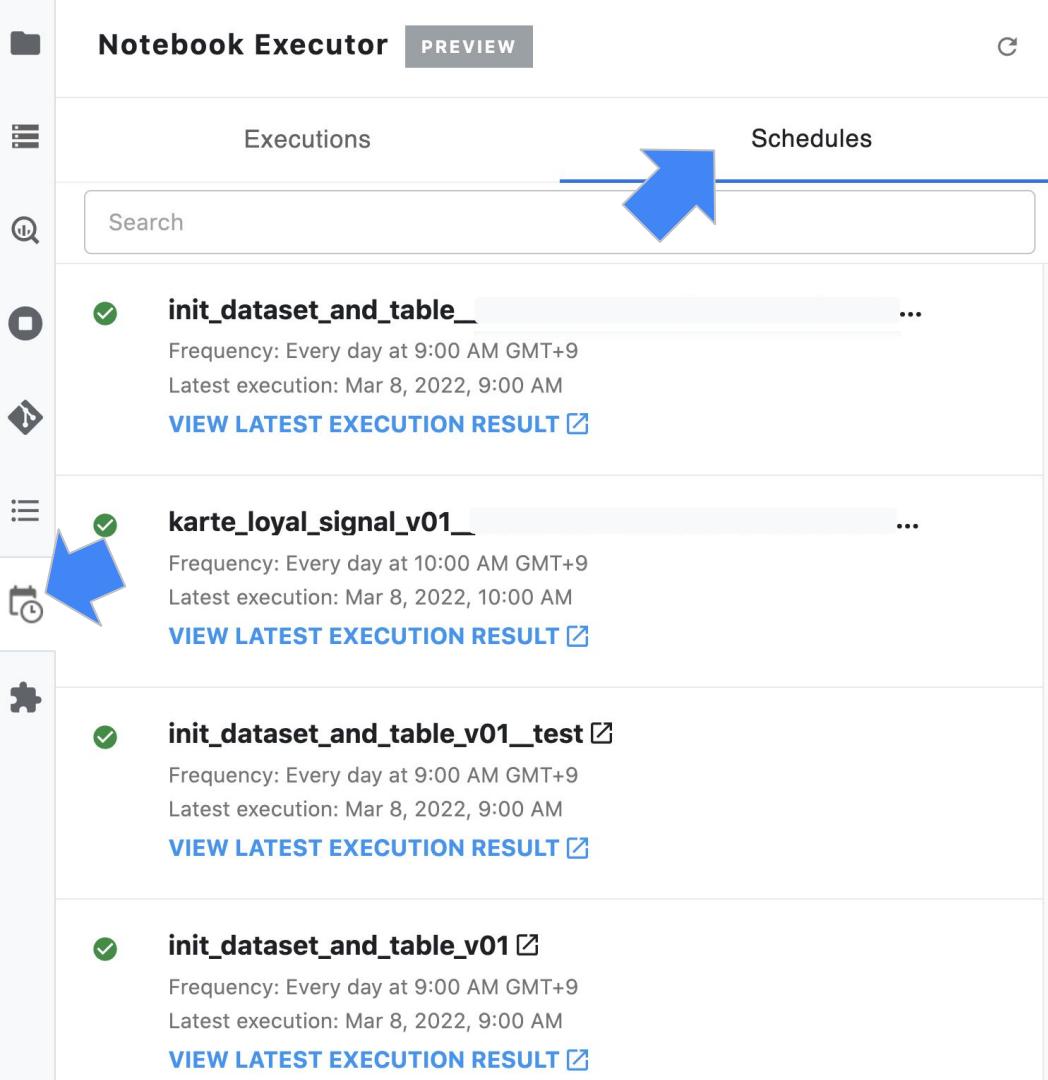
Frequency: Every day at 9:00 AM GMT+9  
Latest execution: Mar 8, 2022, 9:00 AM

[VIEW LATEST EXECUTION RESULT](#)

init\_dataset\_and\_table\_v01

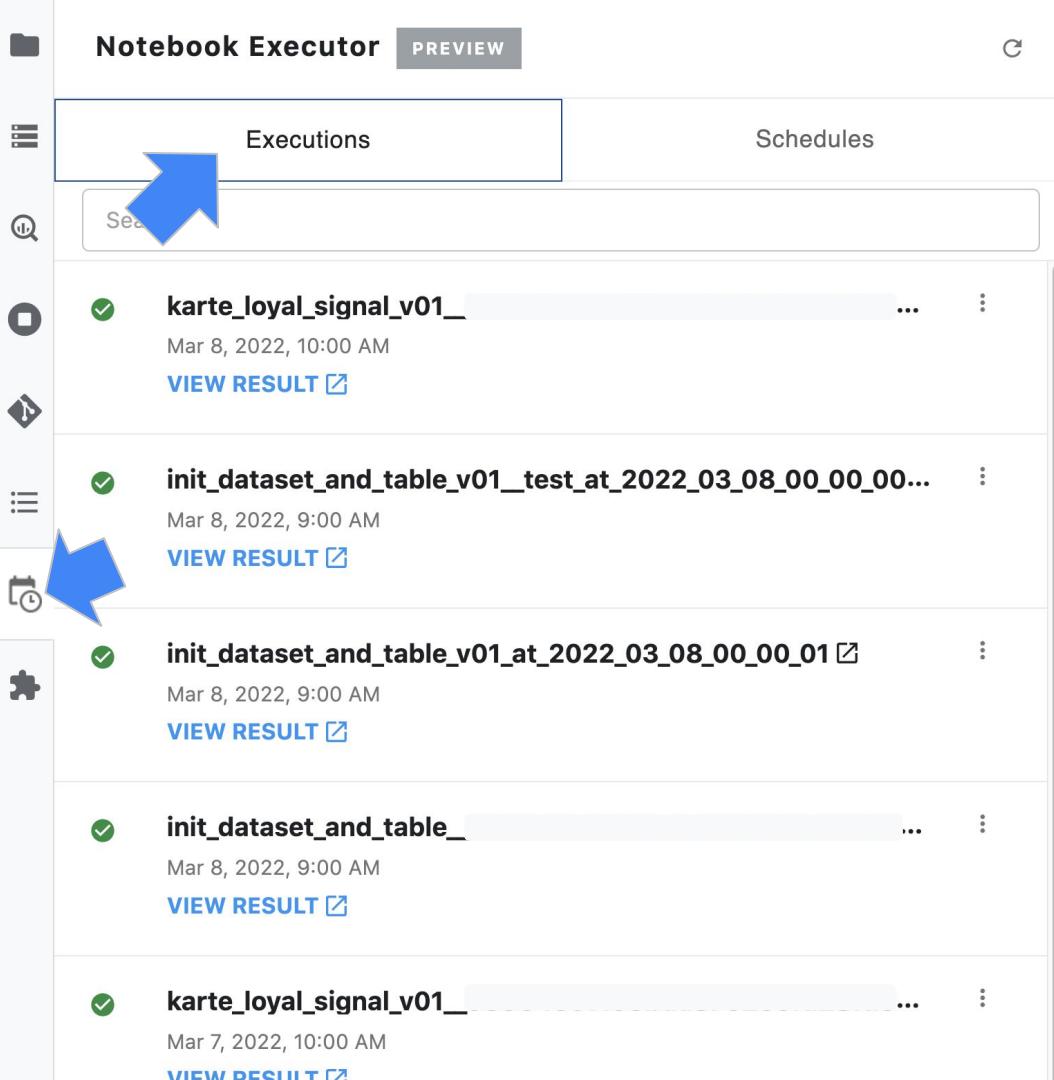
Frequency: Every day at 9:00 AM GMT+9  
Latest execution: Mar 8, 2022, 9:00 AM

[VIEW LATEST EXECUTION RESULT](#)



# Schedule Execution

「Schedules」のタブから設定したタスクが確認できます



# Schedule Execution

「Executions」のタブからタスクの実行結果がそれぞれ確認できます

The screenshot shows the 'Notebook Executor' interface with the 'PREVIEW' tab selected. On the left, there is a vertical sidebar with icons for file operations like Create, Open, Save, Copy, Paste, and Delete. The main area has two tabs: 'Executions' (selected) and 'Schedules'. Below the tabs is a search bar. The execution list displays five entries:

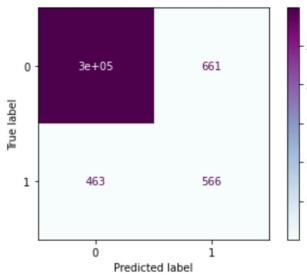
- karte\_loyal\_signal\_v01\_** (Mar 8, 2022, 10:00 AM) - A blue box highlights the 'VIEW RESULT' button.
- init\_dataset\_and\_table\_v01\_test\_at\_2022\_03\_08\_00\_00\_00...** (Mar 8, 2022, 9:00 AM) - A blue box highlights the 'VIEW RESULT' button.
- init\_dataset\_and\_table\_v01\_at\_2022\_03\_08\_00\_00\_01** (Mar 8, 2022, 9:00 AM) - A blue box highlights the 'VIEW RESULT' button.
- init\_dataset\_and\_table\_** (Mar 8, 2022, 9:00 AM) - A blue box highlights the 'VIEW RESULT' button.
- karte\_loyal\_signal\_v01\_** (Mar 7, 2022, 10:00 AM) - A blue box highlights the 'VIEW RESULT' button.

Each execution entry includes a green checkmark icon, a three-dot menu icon, and a 'VIEW RESULT' button with a magnifying glass icon.

# Schedule Execution

さらに「VIEW RESULT」から実行結果  
が簡単に確認できます

Out[19]:  
<sklearn.metrics.\_plot.confusion\_matrix.ConfusionMatrixDisplay at 0x7f1bf18b2590>

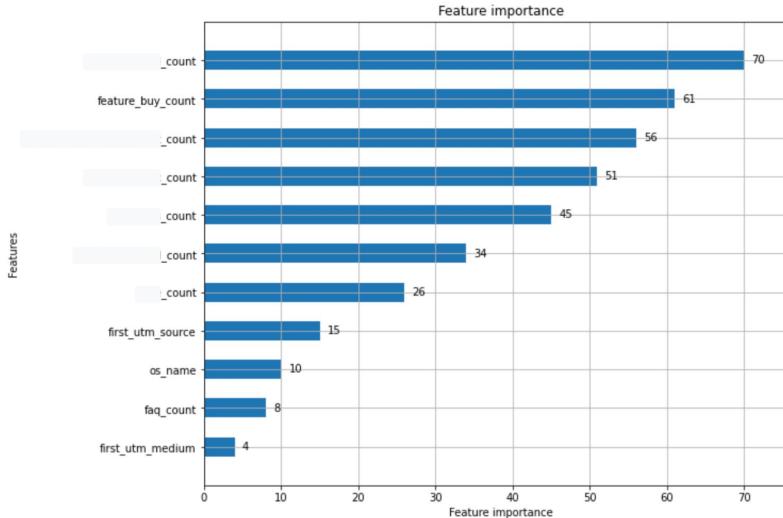


In [20]:  
fig, ax = plt.subplots(1, 1, figsize=(10, 8))

# [-1] means last value in pipeline model list.  
lgb.plot\_importance(lgb\_gscv.best\_estimator\_[-1], ax=ax, height=0.5)

Out[20]:

<AxesSubplot:title={'center':'Feature importance'}, xlabel='Feature importance', ylabel='Features'>



# Schedule Execution

Confusion Matrix や Feature Importance なども確認できます

This is a partitioned table. [Learn more](#)

SCHEMA

DETAILS

PREVIEW

## Table info

Table ID	dev-karte-ml:signals_prediction_mock.loyal_customer_v01
Table size	34.17 MB
Long-term storage size	0 B
Number of rows	484 991
Created	Feb 4, 2022, 3:24:29 PM UTC+9
Last modified	Feb 24, 2022, 10:04:06 AM UTC+9
Table expiration	NEVER
Data location	US
Description	
Table Type	Partitioned
Partitioned by	DAY
Partitioned on field	generated_sync_date
Partition expiration	
Partition filter	Not required



# Output Table

出力先のテーブルが Schedule 通りに更新されていることを確認します



user_id	label	predict_proba	generated_sync_date
0	0.3654801268186312	2022-03-08 01:05:08.440605 UTC	
0	0.3654801268186312	2022-03-08 01:05:08.440605 UTC	
0	0.3654801268186312	2022-03-08 01:05:08.440605 UTC	
0	0.3654801268186312	2022-03-08 01:05:08.440605 UTC	
0	0.3654801268186312	2022-03-08 01:05:08.440605 UTC	
0	0.13729852252718888	2022-03-08 01:05:08.440605 UTC	
0	0.13729852252718888	2022-03-08 01:05:08.440605 UTC	
1	0.9826242377621407	2022-03-08 01:05:08.440605 UTC	
1	0.5410283712352344	2022-03-08 01:05:08.440605 UTC	
1	0.6794460443412431	2022-03-08 01:05:08.440605 UTC	
1	0.6794460443412431	2022-03-08 01:05:08.440605 UTC	
1	0.6794460443412431	2022-03-08 01:05:08.440605 UTC	
1	0.7449196034860508	2022-03-08 01:05:08.440605 UTC	
1	0.7449196034860508	2022-03-08 01:05:08.440605 UTC	
1	0.7449196034860508	2022-03-08 01:05:08.440605 UTC	
1	0.7449196034860508	2022-03-08 01:05:08.440605 UTC	
1	0.7449196034860508	2022-03-08 01:05:08.440605 UTC	
1	0.7449196034860508	2022-03-08 01:05:08.440605 UTC	
1	0.7449196034860508	2022-03-08 01:05:08.440605 UTC	
1	0.7449196034860508	2022-03-08 01:05:08.440605 UTC	
1	0.999999852100367	2022-03-08 01:05:08.440605 UTC	

Rows per page:

100 ▾

1 - 100 of 158

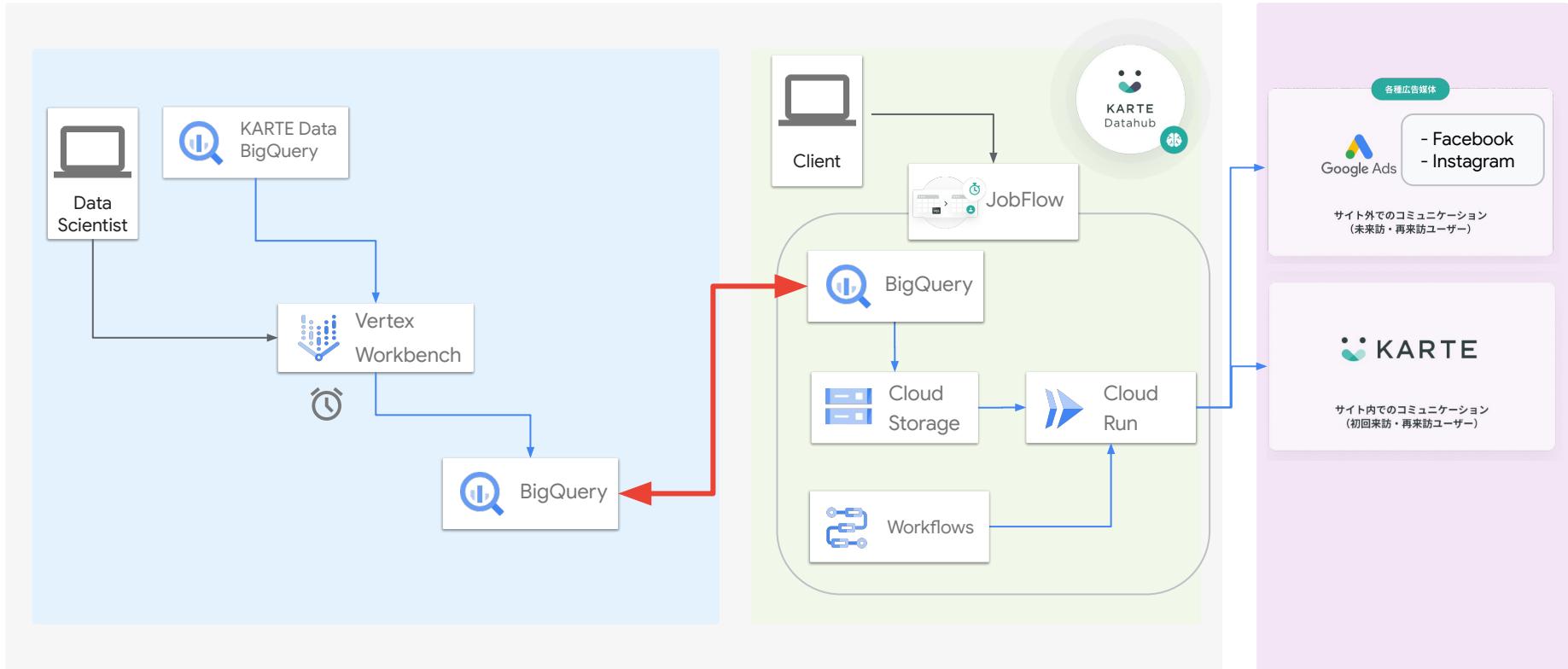
First page



## Output Table

ユーザーごとに CV するかどうかのラベルの予測値が更新されます

# “Light” MLOps Architecture



## 外部のデータセットを登録

事前にDatahubサービスアカウントに対しデータセットの参照権限を付与する必要があります。

データセットの登録には数分程度かかる場合があります。

### プロジェクトID

dev-karte-ml

### データセットID

signals\_prediction\_mock

### データセットの表示名

KARTE Signals

未入力で保存した場合、データセットIDのみの表示になります

キャンセル

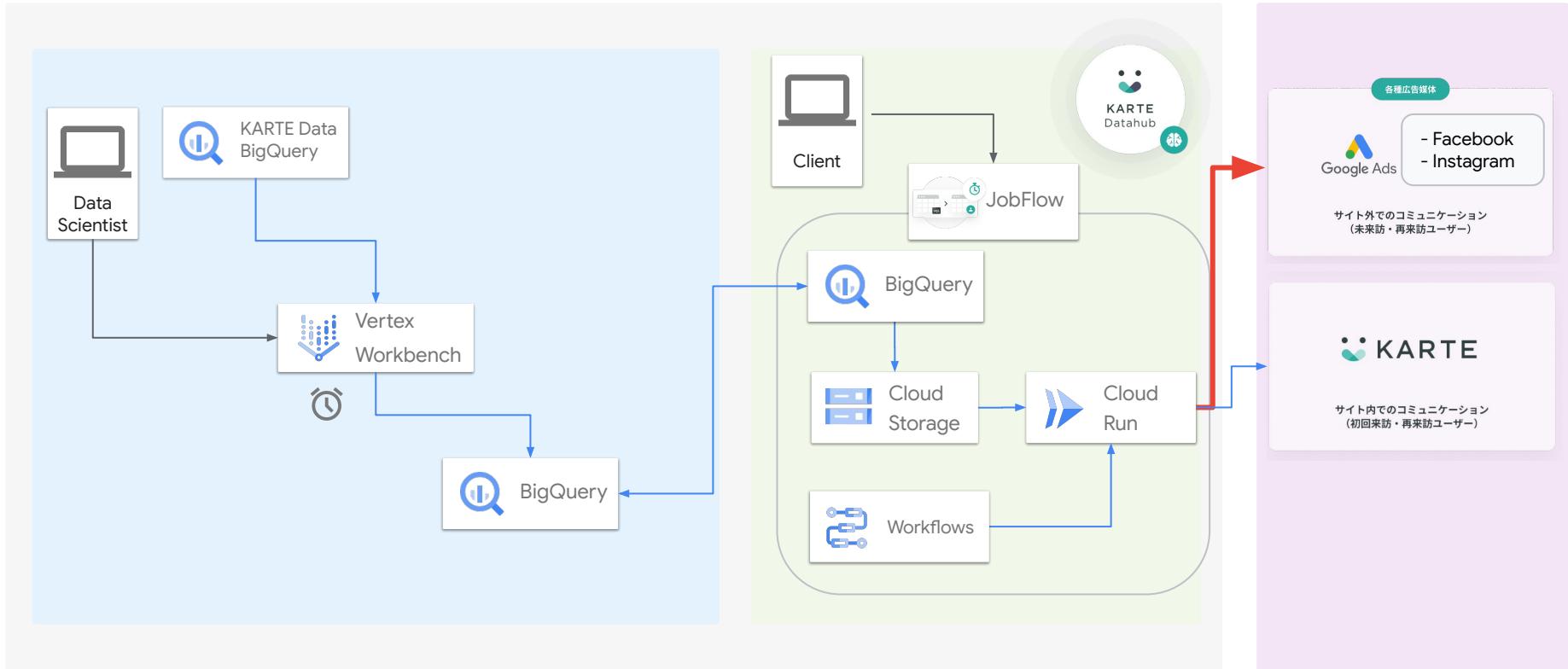
登録する

# External Table Link

KARTE Datahub のデータセットと連携します



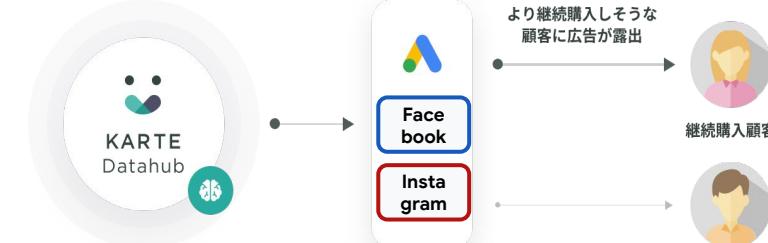
# “Light” MLOps Architecture



クエリ名 Adsコネクタ / fb広告連携用クエリ

エクスポート先 外部データ

接続先 Facebook Conversions API / CAPI(Default Pixel)



# External Table Link

JobFlow にて Facebook  
Conversions API に日次で連携する  
ジョブを構築します

# Data Scientist のワークフロー

## Notebook

Notebook で分析及び機械学習  
モデルを作成

## Schedule Execution

Notebook を定期実行する

## External Table Link

必要に応じて KARTE または  
外部のテーブルと連携する



モデル開発に集中可能！

エンジニア工数なし！



# 今後の展望

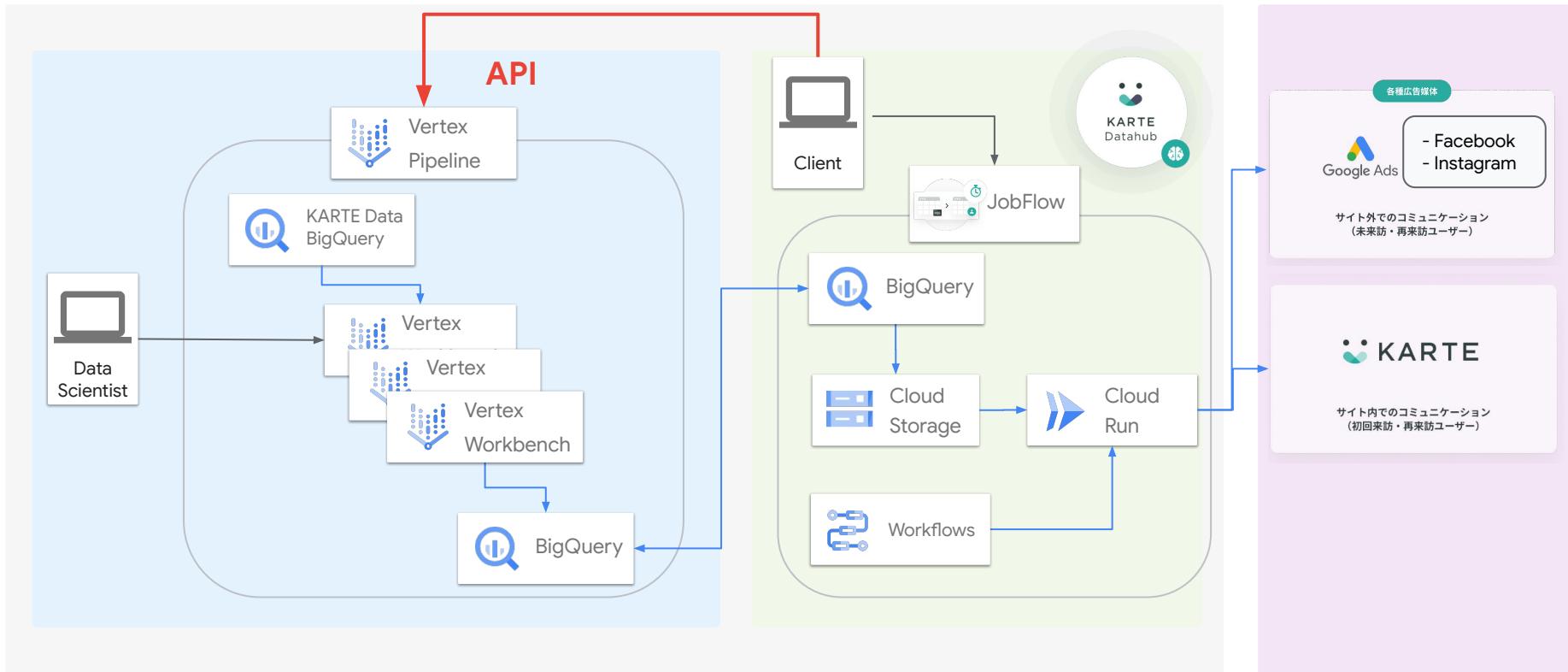
# Vertex Pipeline

execute notebook operator を利用することで、任意の notebook を pipeline 連携できるようになります

```
@kfp.dsl.pipeline
  name="notebook-as-a-step-sample",
  pipeline_root=PIPELINE_ROOT_PATH)
def pipeline(
    project: str,
    execution_id: str,
    input_notebook_file:str,
    output_notebook_folder:str,
    location:str,
    master_type:str,
    container_image_uri:str):

    execute_notebook_component = kfp.components.load_component_from_file(COMPONENT_YAML_
execute_notebook_op = execute_notebook_component(
    project=project,
    execution_id=execution_id,
    input_notebook_file=input_notebook_file,
    output_notebook_folder=output_notebook_folder,
    location=location,
    master_type=master_type,
    container_image_uri=container_image_uri,
    parameters=f'PROJECT_ID={project},EXECUTION_ID={execution_id}'
)
```

# “Light” MLOps Future Architecture





# まとめ

# “Light” MLOps まとめ

- 01 | エンジニアの工数を用いずに、データサイエンティストのみで MLOps がスムーズに行えるようになる
- 02 | データサイエンティストがこれまで以上にモデル開発に集中できるようになる
- 03 | Vertex Pipeline 等の Vertex AI 各種機能を用いることで、さらなる拡張が可能になる

# Thank you.

