Oracle Data Lakehouse

Robert Korošec, Oracle
Grega Dvoršak, Qubix
Žiga Vaupot, Qubix

2. junij 2022

Qubix
Enabling the Predictive Enterprise

Intro to the Data Lakehouse

Qubix

Typical Approach

Data Warehouse

- Solves the problem of analyzing transactional data
- Focus on curated data with known value that is well understood

Data Lake

- Handles data that is raw and un-curated. Unknown value or low value
- Open-source tools for processing, analysis and more

Neither approach alone can answer those questions

Qubix

Introducing the Data Lakehouse

Integrate the power of a **modern Cloud-based elastic SQL analytic platforms** with **modern data design of the Data Lake** to handle integrated analysis for all data

A lakehouse architecture can answer those questions

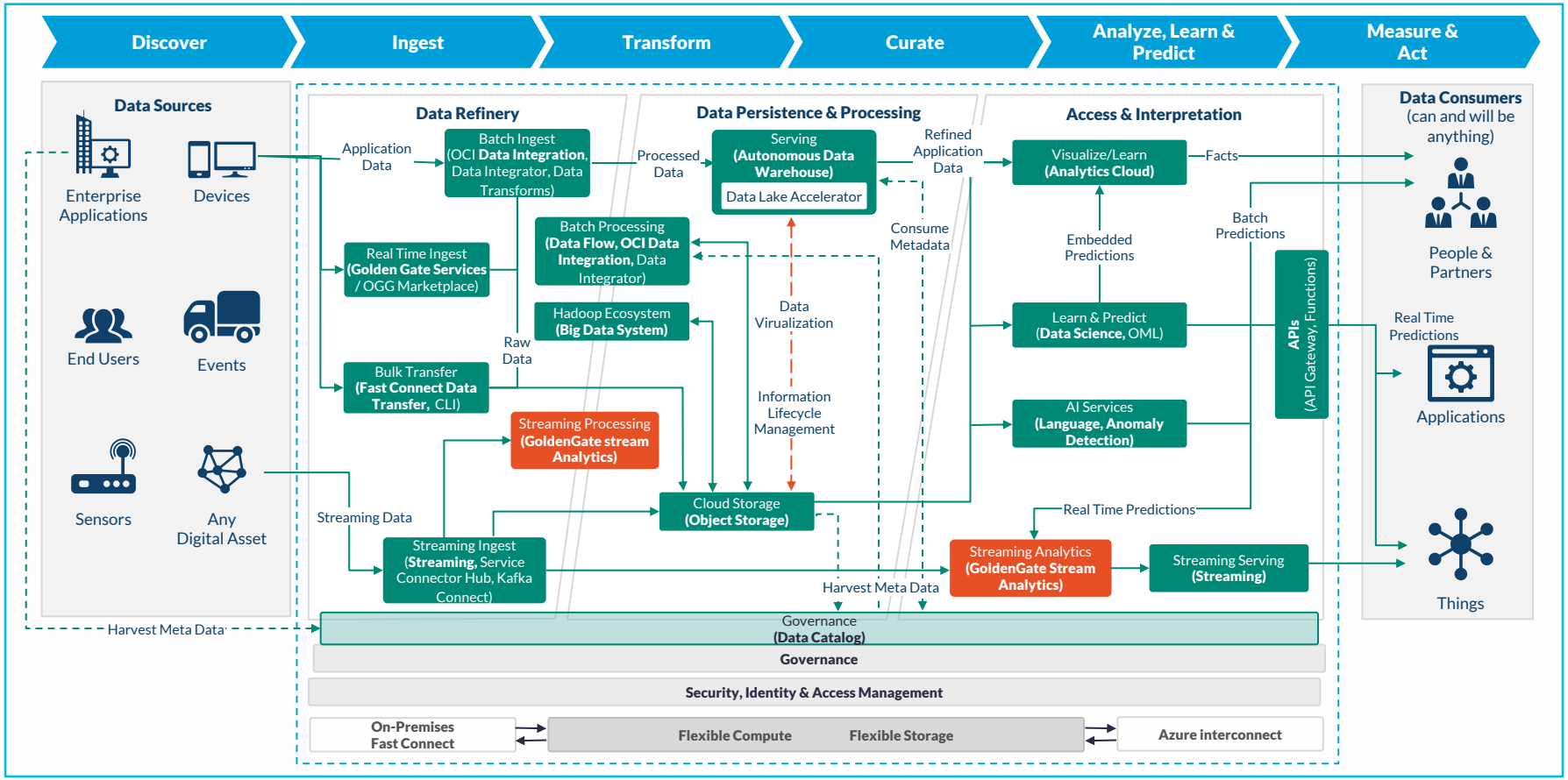
Qubix

Oracle Cloud Data Lakehouse – Reference Architecture

Built on OCI Backbone

Centered around:

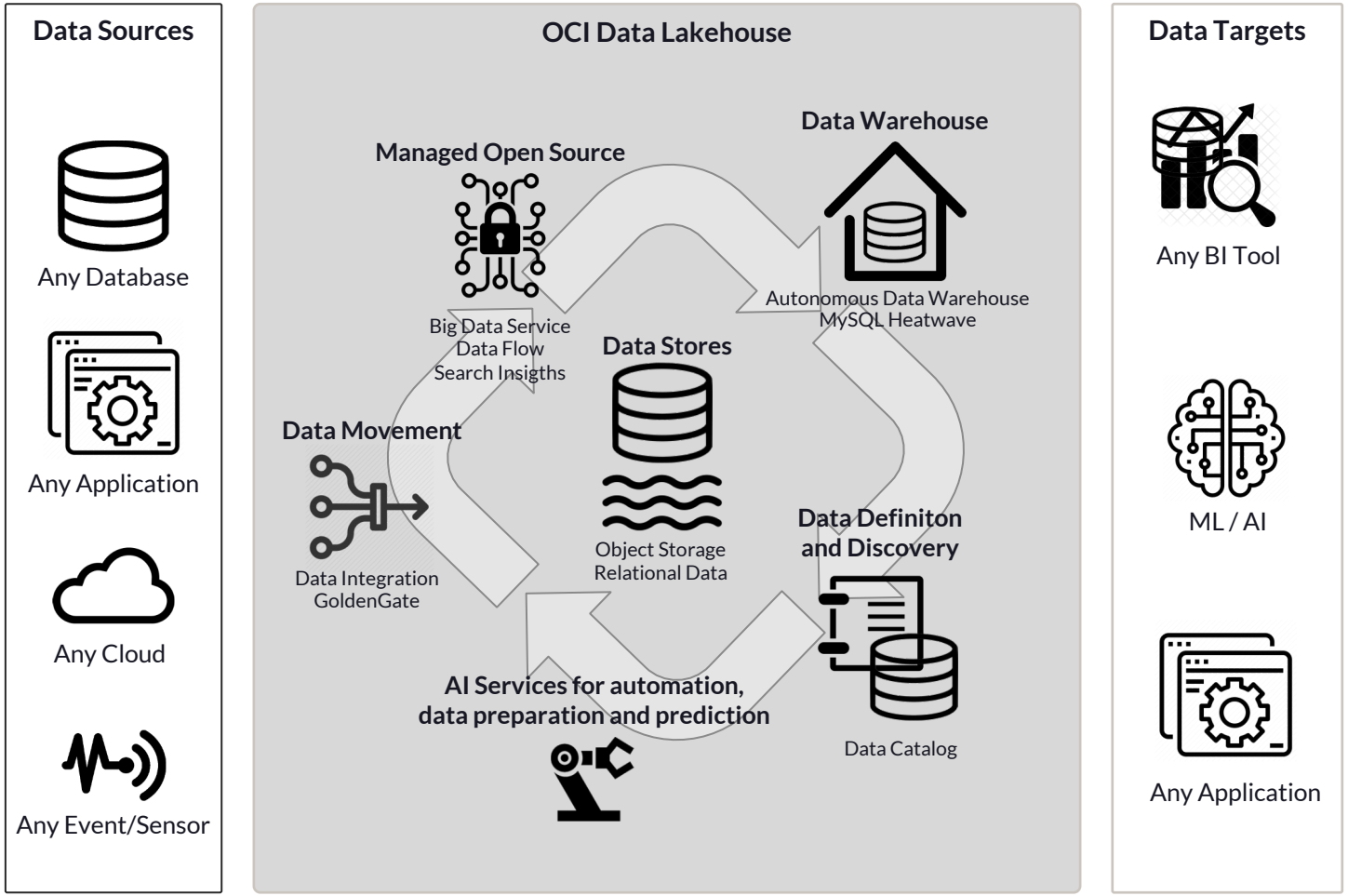
- Oracle Autonomous Data Warehouse
- Oracle Machine Learning
- Oracle Cloud Infrastructure Data Integration
- Oracle Cloud Infrastructure Data Flow
- Big Data Service
- Data Catalog
- Oracle Cloud Infrastructure Streaming



https://docs.oracle.com/en/solutions/oci-curated-analysis/index.htm

Qubix

Oracle Data Lakehouse

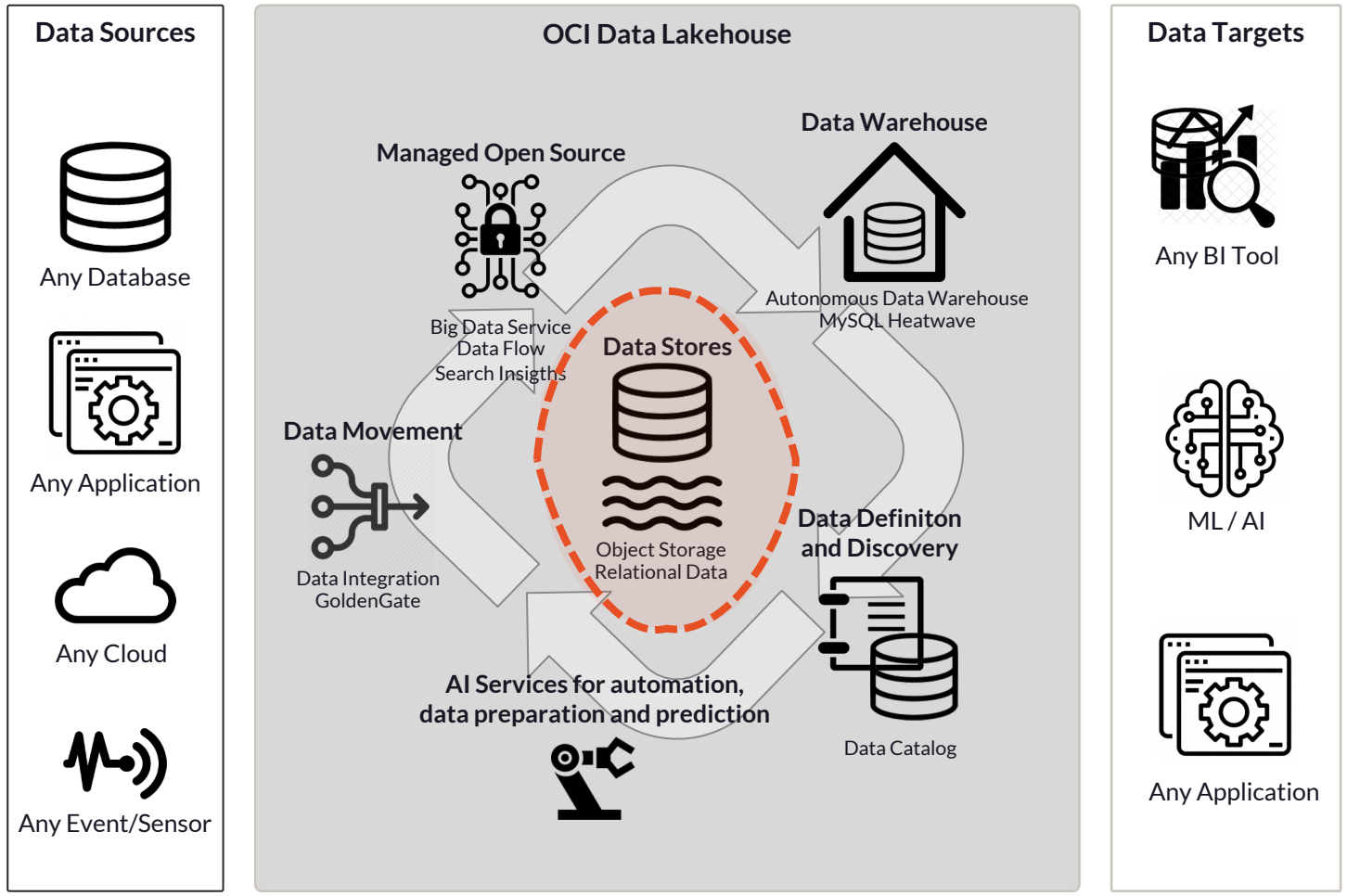


Qubix

Data Lakehouse
Object Storage

Qubix

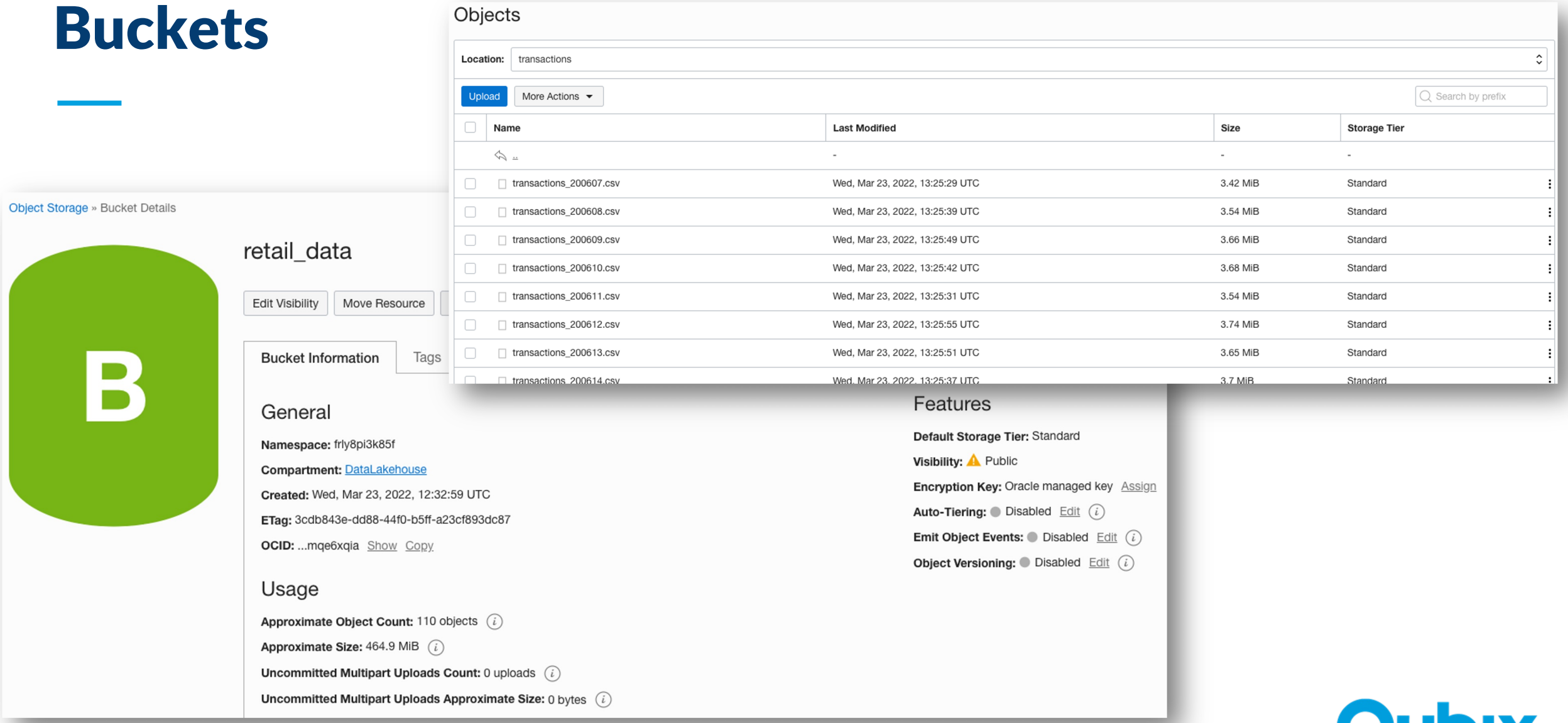
Data Stores: Object Storage



- Internet-scale, high-performance storage platform that offers reliable and cost-efficient data durability.
- Stores an unlimited amount of unstructured data of any content type, including analytic data and rich content, like images and videos.

Qubix

Buckets


Objects
Location: transactions
Upload
More Actions
Search by prefix
Name
Last Modified
Size
Storage Tier
transactions_200607.csv Wed, Mar 23, 2022, 13:25:29 UTC 3.42 MiB Standard
transactions_200608.csv Wed, Mar 23, 2022, 13:25:39 UTC 3.54 MiB Standard
transactions_200609.csv Wed, Mar 23, 2022, 13:25:49 UTC 3.66 MiB Standard
transactions_200610.csv Wed, Mar 23, 2022, 13:25:42 UTC 3.68 MiB Standard
transactions_200611.csv Wed, Mar 23, 2022, 13:25:31 UTC 3.54 MiB Standard
transactions_200612.csv Wed, Mar 23, 2022, 13:25:55 UTC 3.74 MiB Standard
transactions_200613.csv Wed, Mar 23, 2022, 13:25:51 UTC 3.65 MiB Standard
Object Storage » Bucket Details
B
retail_data
Edit Visibility
Move Resource
Bucket Information
Tags
General
Namespace: frly8pi3k85f
Compartment: DataLakehouse
Created: Wed, Mar 23, 2022, 12:32:59 UTC
ETag: 3cdb843e-dd88-44f0-b5ff-a23cf893dc87
OCID: ...mqe6xqia Show Copy
Usage
Approximate Object Count: 110 objects
Approximate Size: 464.9 MiB
Uncommitted Multipart Uploads Count: 0 uploads
Uncommitted Multipart Uploads Approximate Size: 0 bytes
Features
Default Storage Tier: Standard
Visibility: Public
Encryption Key: Oracle managed key Assign
Auto-Tiering: Disabled Edit
Emit Object Events: Disabled Edit
Object Versioning: Disabled Edit

Accessing Object Storage data using SQL

```
BEGIN
    DBMS_CLOUD.CREATE_EXTERNAL_TABLE(
        table_name => 'RETAIL_TRANSACTIONS',
        file_uri_list => 'https://objectstorage.eu-frankfurt-1.oraclecloud.com/p/CTuPS5zVfoZERhR7Y_fRDh64VXSwJfsHFAgidjRMj19HHOsWG3xyHZOTtB9a0vc_/n/frly8pi3k85f/b/retail_data/o/*.csv',
        format => json_object('type' value 'csv', 'skipheaders' value '1', 'delimiter' value ','),
        column_list => ' SHOP_WEEK VARCHAR2 (6) ,
                        SHOP_DATE VARCHAR2 (8) ,
                        SHOP_WEEKDAY NUMBER ,
                        SHOP_HOUR NUMBER ,
                        QUANTITY NUMBER ,
                        SPEND NUMBER ,
                        PROD_CODE VARCHAR2 (10) ,
                        PROD_CODE_10 VARCHAR2 (7) ,
                        PROD_CODE_20 VARCHAR2 (8) ,
                        PROD_CODE_30 VARCHAR2 (6) ,
                        PROD_CODE_40 VARCHAR2 (6) ,
                        CUST_CODE VARCHAR2 (14) ,
                        CUST_PRICE_SENSITIVITY VARCHAR2 (2) ,
                        CUST_LIFESTAGE VARCHAR2 (2) ,
                        BASKET_ID NUMBER ,
                        BASKET_SIZE VARCHAR2 (1) ,
                        BASKET_PRICE_SENSITIVITY VARCHAR2 (2) ,
                        BASKET_TYPE VARCHAR2 (20) ,
                        BASKET_DOMINANT_MISSION VARCHAR2 (20) ,
                        STORE_CODE VARCHAR2 (10) ,
                        STORE_FORMAT VARCHAR2 (2) ,
                        STORE_REGION VARCHAR2 (3) ');
END;
```

```
SELECT * FROM RETAIL_TRANSACTIONS
WHERE SHOP_WEEK = 200607;
```

Query Result | Script Output | DBMS Output | Explain Plan | Autotrace | SQL History | Data Loading

Download ▾ Execution time: 26.881 seconds

	shop_week	shop_date	shop_weekday	shop_hour	quantity	spend	prod_code	pro
1	200607	20060415	7	19	1	0.93	PRD0900033	CL
2	200607	20060413	5	20	1	1.03	PRD0900097	CL
3	200607	20060416	1	14	1	0.98	PRD0900121	CL
4	200607	20060415	7	19	1	3.07	PRD0900135	CL
5	200607	20060415	7	19	1	4.81	PRD0900220	CL
6	200607	20060412	4	19	1	0.28	PRD0900353	CL
7	200607	20060413	5	18	1	1.56	PRD0900547	CL
8	200607	20060413	5	20	3	0.84	PRD0900550	CL

Qubix

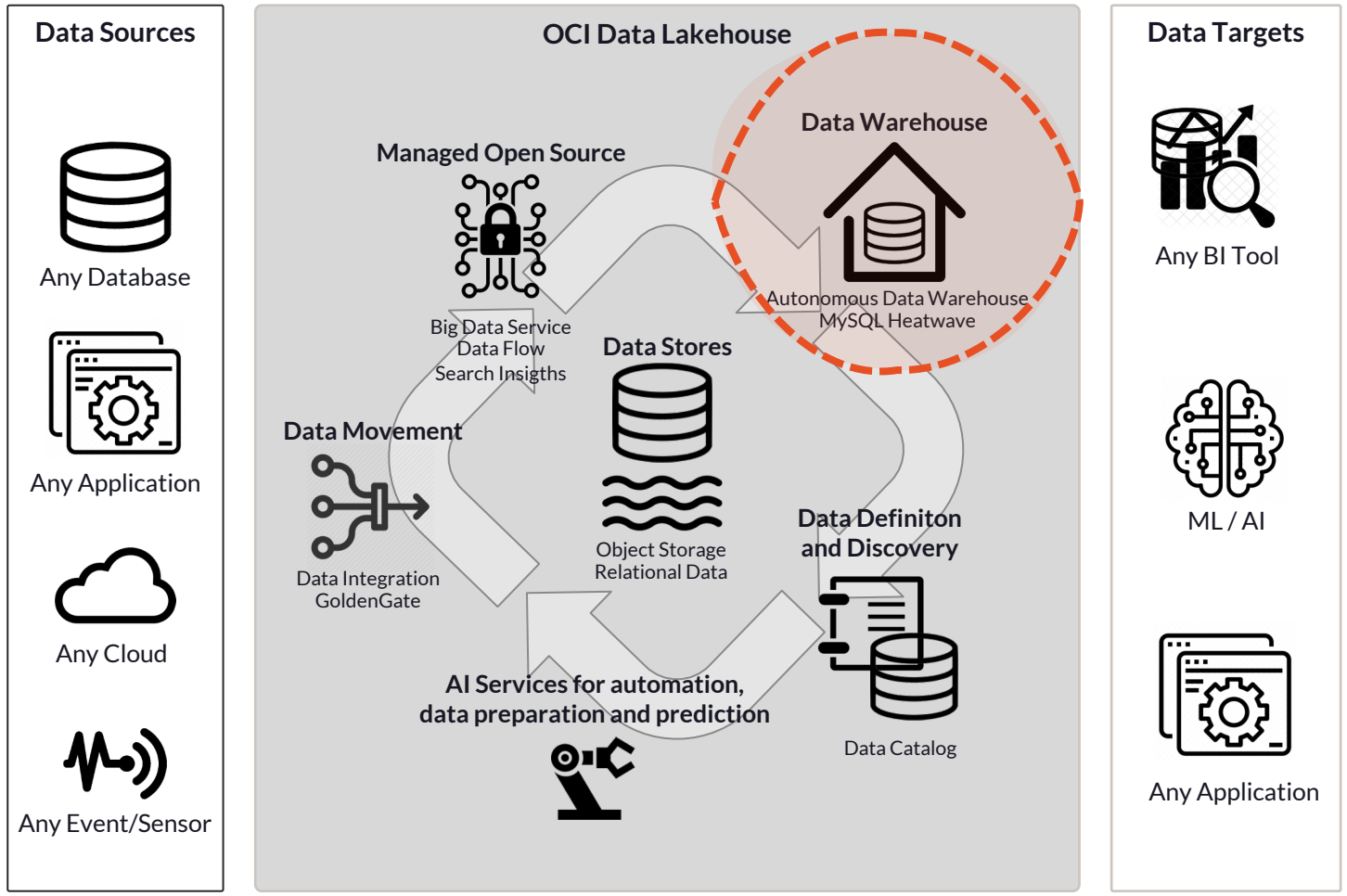
And finally, bring all into Oracle Analytics

Data Lakehouse

Data Warehouse

Qubix

Data Warehouse



Autonomous Data Warehouse

- Autonomous Database for Analytics and Data Warehousing provides an easy-to-use, fully autonomous database that scales elastically, delivers fast query performance and requires no database administration.
- Autonomous Database, is deployed on Exadata infrastructure by default

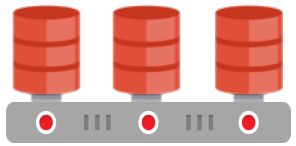
MySQL Heatwave

- HeatWave is a massively parallel, high performance, in-memory query accelerator that accelerates MySQL performance by orders of magnitude for analytics workloads, mixed workloads, and machine learning.

Qubix

Oracle Autonomous Database
Under the hood – A converged database

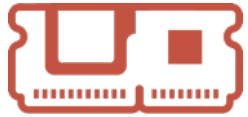
- Multitenant for Efficient, Agile Database Clouds
- AutoML for simple integrated Machine Learning
- In-Memory for Database Acceleration
- Native JSON for Document Data
- In-Memory Ingest for Fastest IoT
- Cloud SQL for integrating Object Store Data Lake
- Persistent Memory Store for Lowest Latency
- Spatial and Graph for Mapping and Social Networks


Multitenant


Machine Learning

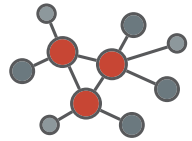

In-Memory Analytics


JSON


Persistent Memory


In-Memory IoT


Spatial

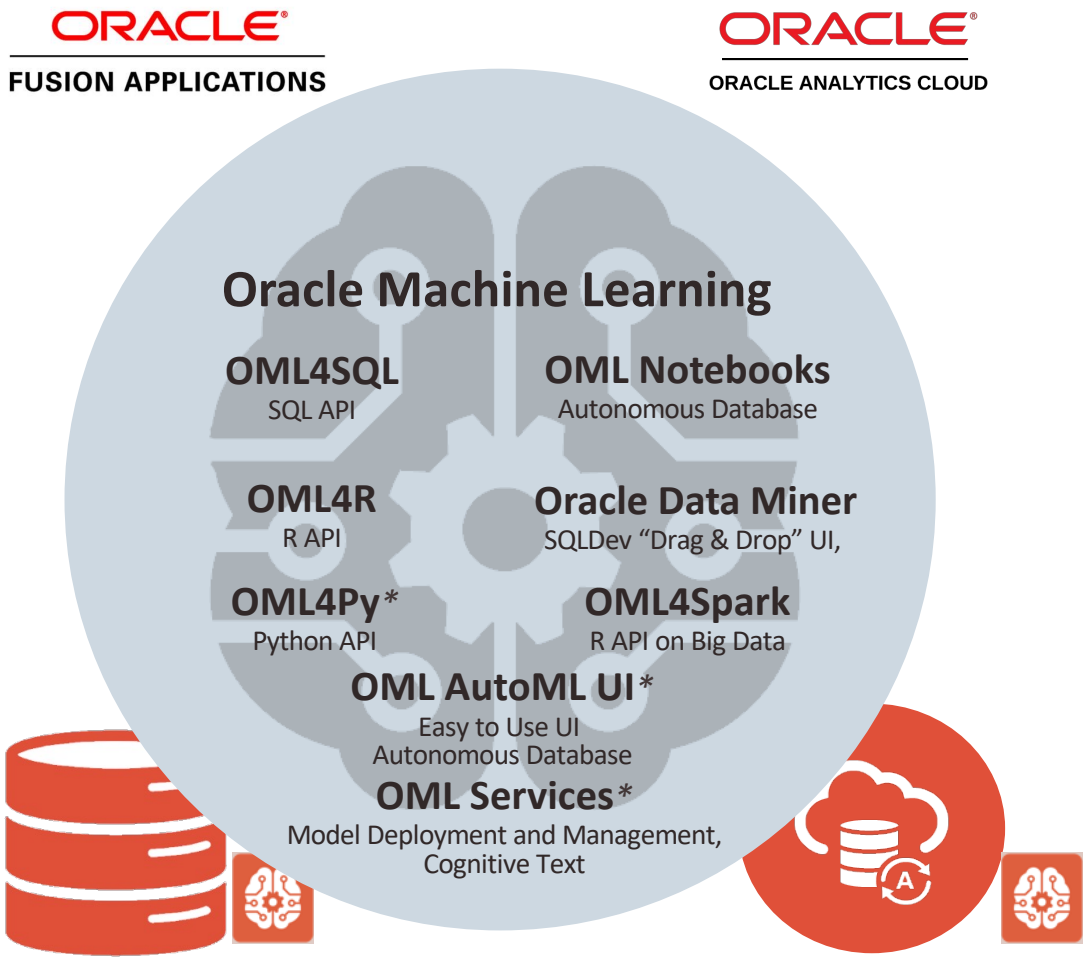

Graph


Cloud Integration

Qubix

Oracle Machine Learning

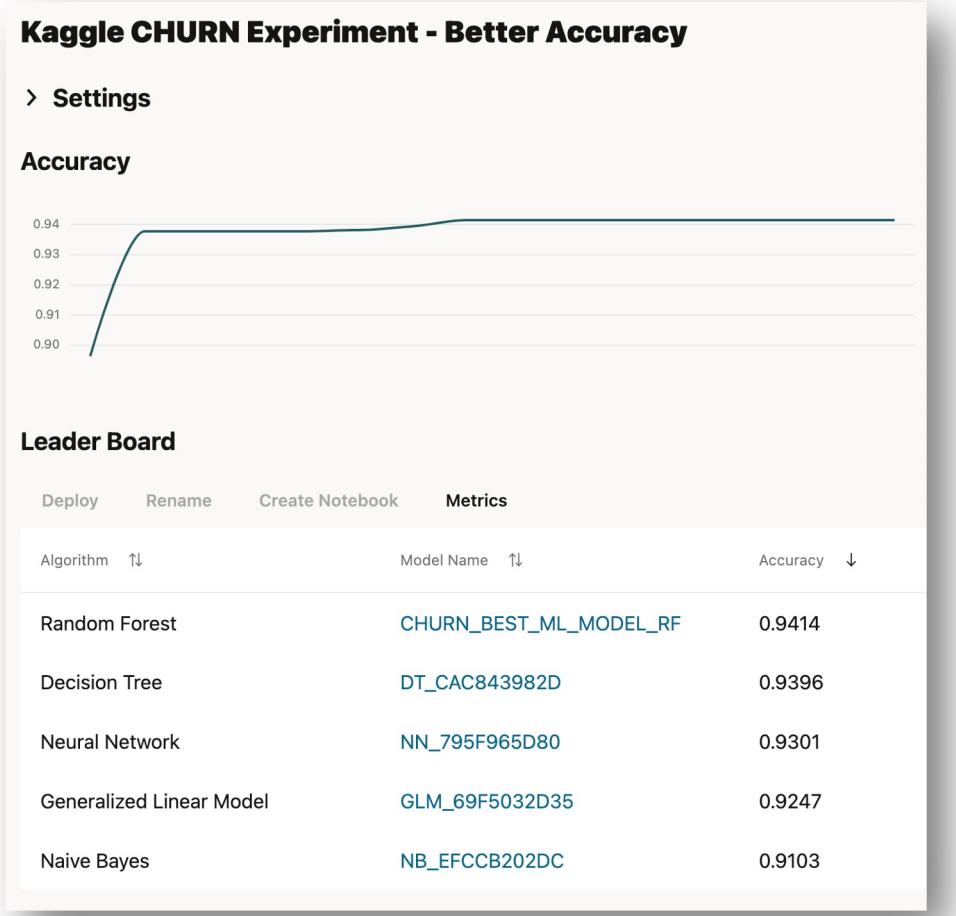
- Oracle Machine Learning extends Oracle Database(s) and enables users to build “AI” applications and analytical dashboards
- OML delivers powerful in-database machine learning algorithms, automated ML functionality via SQL APIs and integration with opensource Python* and R.



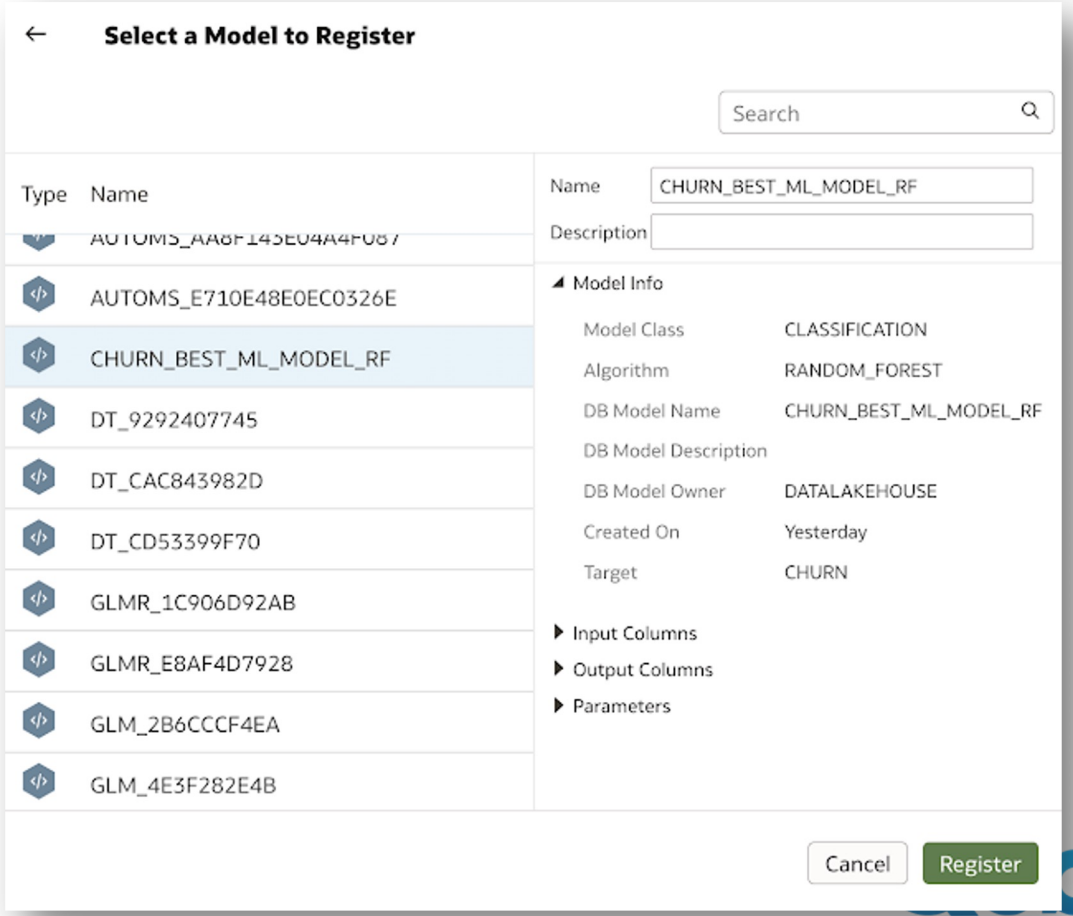
Qubix

OML AutoML UI

Train the best model using OML Auto UI

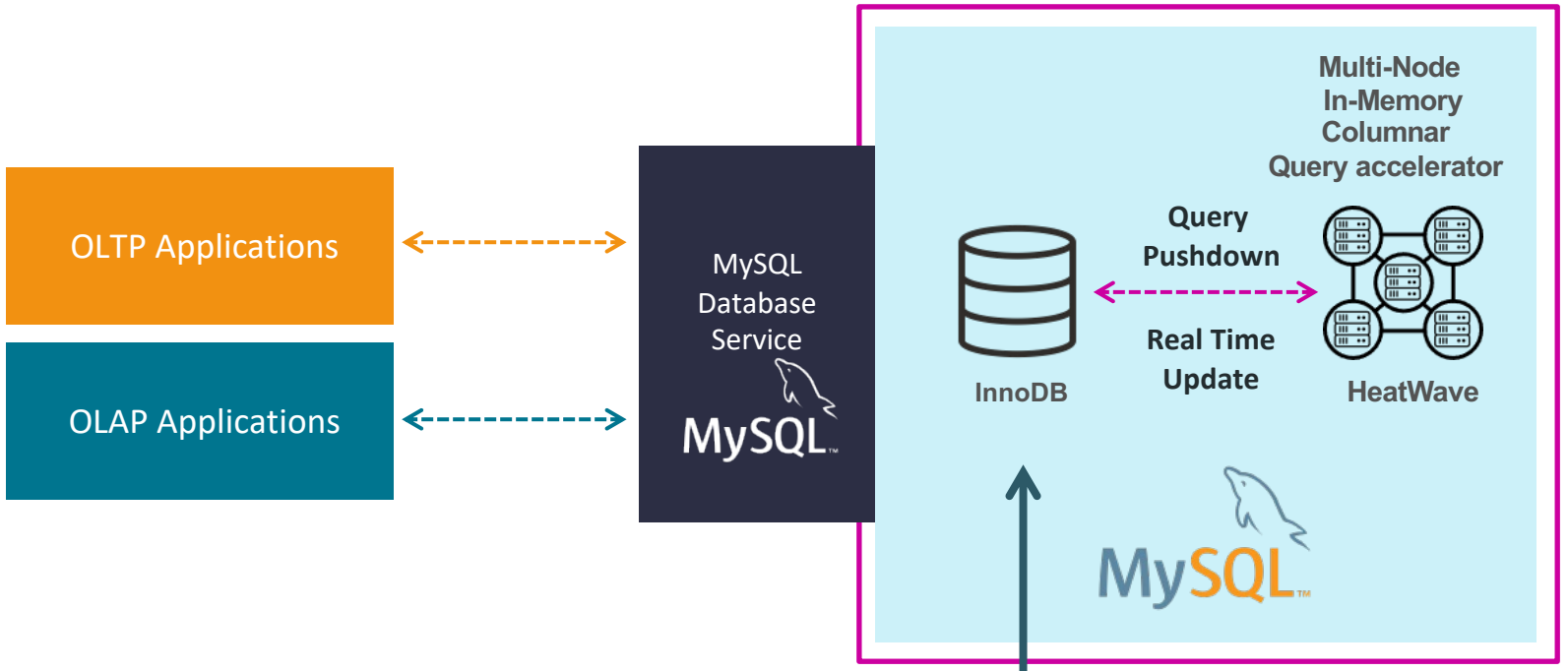


Deploy by registering the best model with Oracle Analytics



MySQL HeatWave

The only MySQL service with a native massively-scalable query accelerator



Single MySQL database for OLTP & analytics applications

All existing applications work without any changes

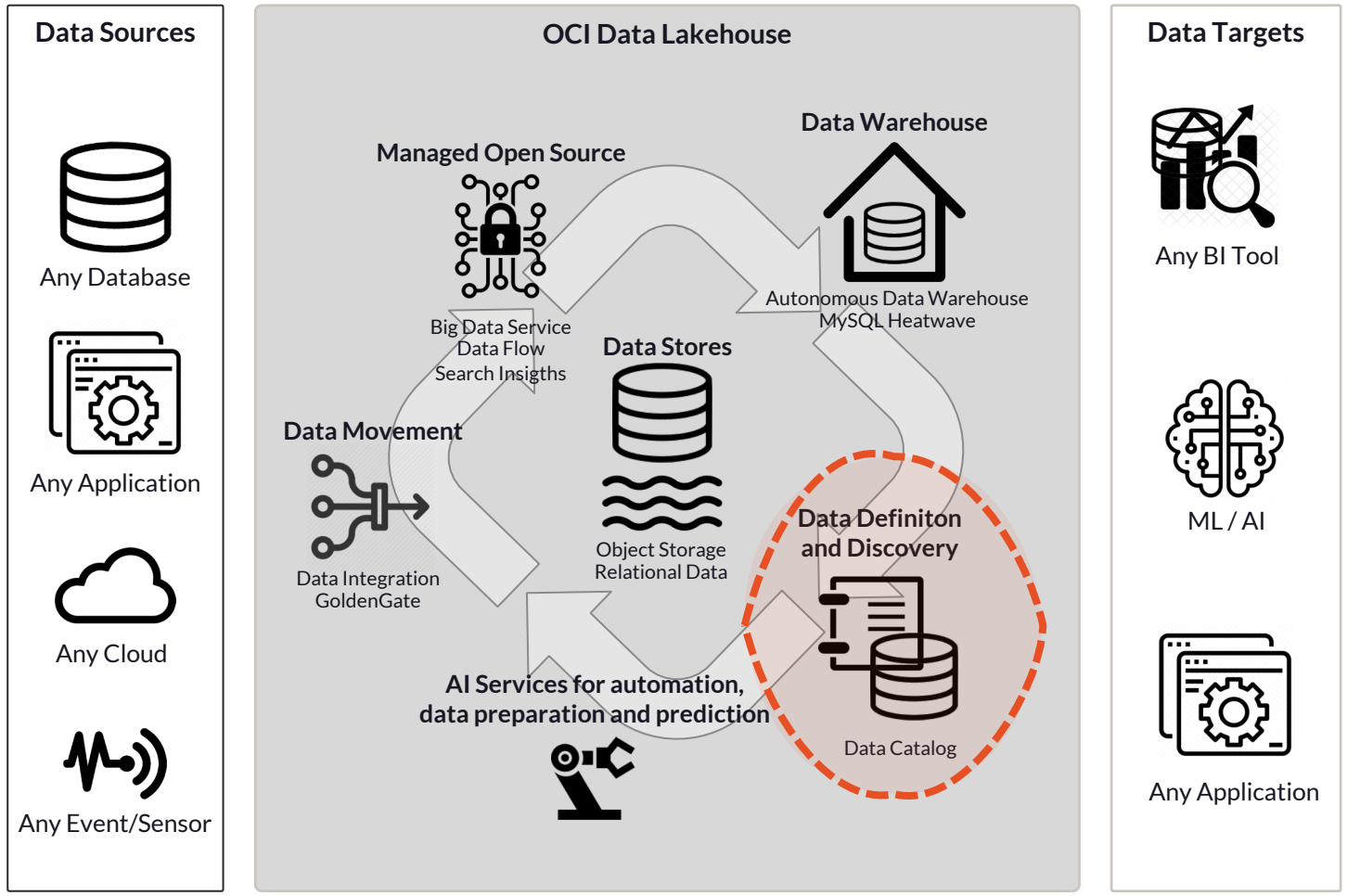
Extreme performance: 400x faster than MySQL, scales to thousands of cores

Qubix

Data Lakehouse
OCI Data Catalog

Qubix

Data Definition and Discovery: Data Catalog



- Metadata management service that helps data professionals discover data and support data governance.
- Designed specifically to work well with the Oracle ecosystem.
- It provides an inventory of assets, a business glossary, and a common metastore for data lakes.

Qubix

Harvest, Enrich, Search

Harvest metadata from a data source into your catalog. Let Data Catalog discover the available data sources, or you can identify them manually.
Learn more

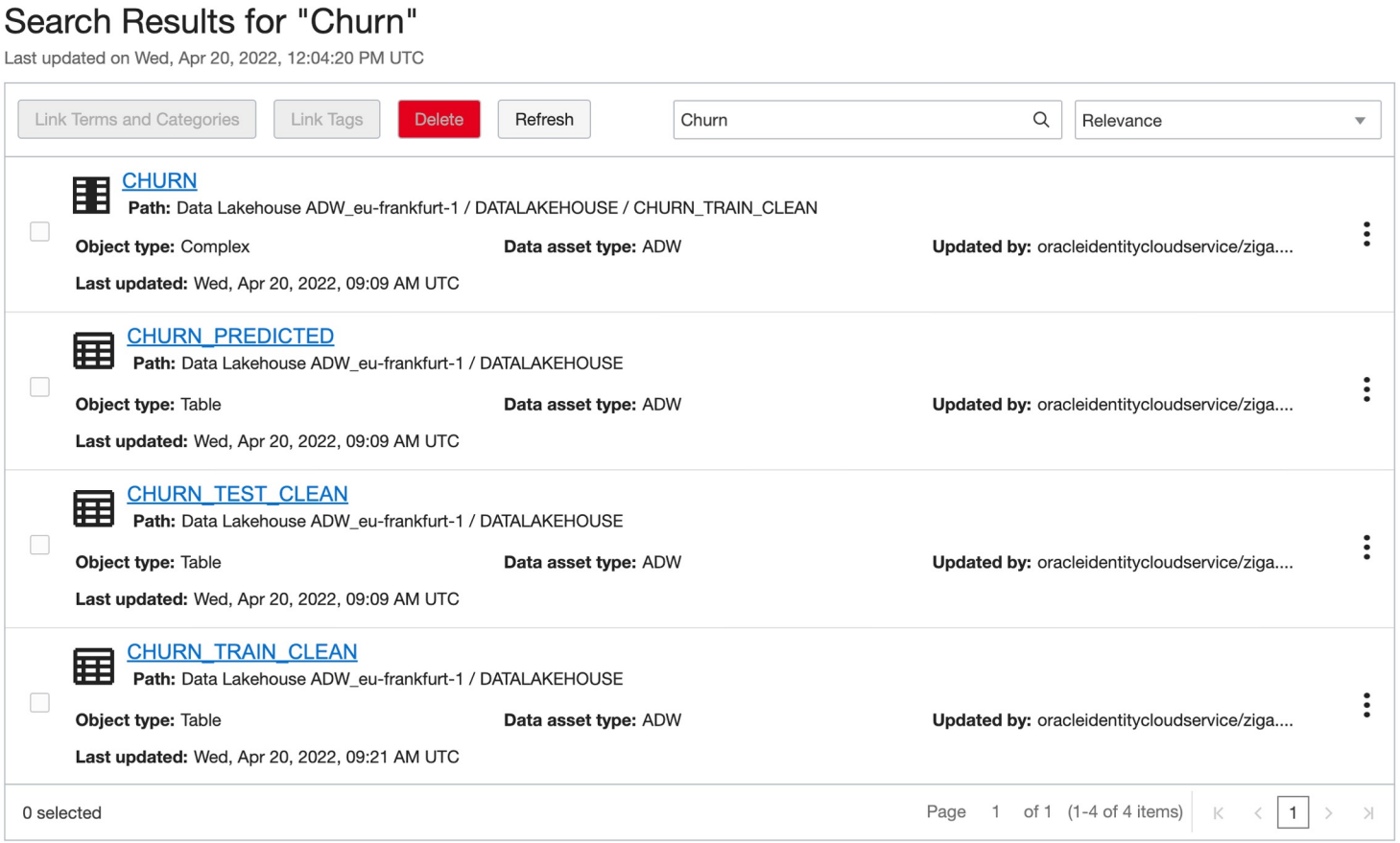
Add business context to the technical metadata using business glossary terms and categories, tags, and user defined properties.
Learn more

Find information about the data available to you by using technical or business names.
Learn more

Qubix

Data Catalog

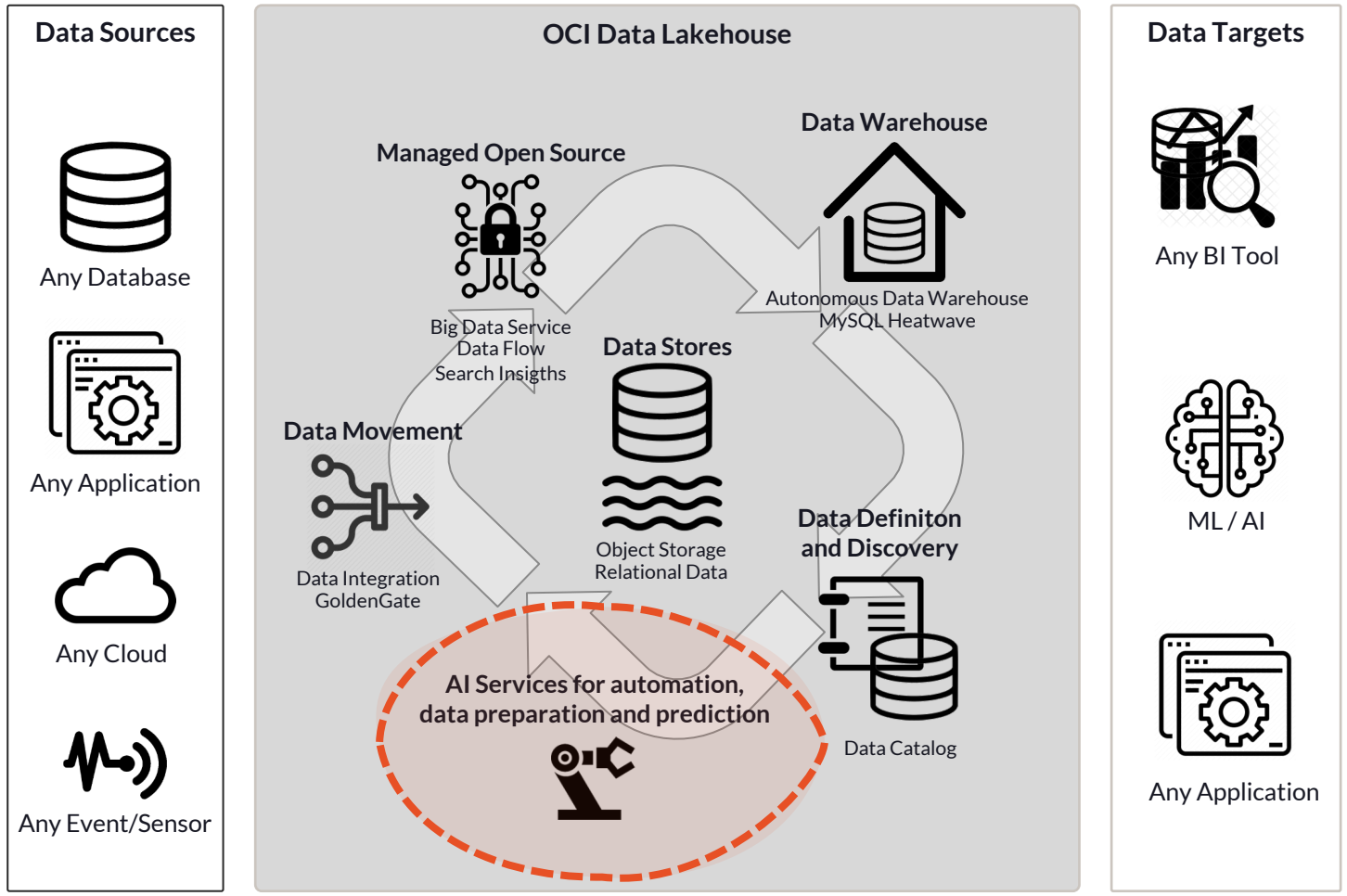
- Discover Data Sources
- Create Data Assets
- Harvesting
- Custom properties
- Glossary and Terms



Data Lakehouse
AI Services for automation, data preparation and prediction

Qubix

Data Science & AI Services



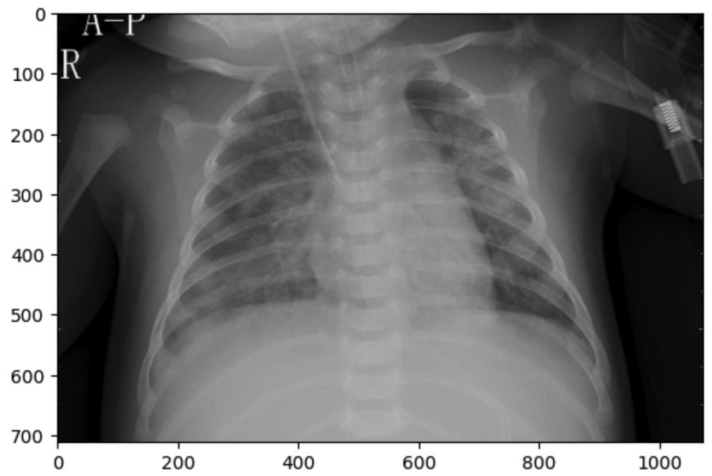
- Data Science Services
  - Data Science
  - Data Labeling
- AI services
  - Vision
  - Speech
  - Language
  - Anomaly Detection
  - Digital Assistant

Qubix

First, let‘s start with OCI Data Science

```
[8]: img = cv2.imread("kaggle/input/chest-xray-pneumonia/chest_xray/chest_xray/val/PNEUMONIA/person1946_bacteria_4875.jpeg")

[9]: plt.imshow(img)
     plt.show()
```




```
[46]: model = Sequential([
      #     data_augmentation,
      #     layers.Reshape((150,150,1)),
          layers.Conv2D(32, (3,3) ,strides = 1, input_shape=(150,150,1), padding='same', activation='relu'),
          layers.BatchNormalization(),
          layers.MaxPool2D((2,2) ,strides = 2 , padding = 'same'),
          layers.Conv2D(64, (3,3) ,strides = 1, padding = 'same', activation = 'relu'),
          layers.Dropout(0.1),
          layers.BatchNormalization(),
          layers.MaxPool2D((2,2) ,strides = 2 , padding = 'same'),
          layers.Conv2D(64, (3,3) ,strides = 1, padding = 'same', activation = 'relu'),
          layers.BatchNormalization(),
          layers.MaxPool2D(pool_size = (2,2) ,strides = 2 , padding = 'same'),
          layers.Conv2D(128, kernel_size = (3,3) ,strides = 1, padding = 'same', activation = 'relu'),
          layers.Dropout(0.2),
          layers.BatchNormalization(),
          layers.MaxPool2D(pool_size = (2,2) ,strides = 2 , padding = 'same'),
          layers.Conv2D(256, kernel_size = (3,3) ,strides = 1, padding = 'same', activation = 'relu'),
          layers.Dropout(0.2),
          layers.BatchNormalization(),
          layers.MaxPool2D(pool_size = (2,2) ,strides = 2 , padding = 'same'),
          layers.Flatten(),
          layers.Dense(units = 128,activation='relu'),
          layers.Dropout(0.2),
          layers.Dense(units = 1 , activation = 'sigmoid')
      ])

      METRICS = [
          keras.metrics.BinaryAccuracy(name='accuracy'),
          keras.metrics.Precision(name='precision'),
          keras.metrics.Recall(name='recall'),
          keras.metrics.AUC(name='auc'),
      ]

      model.compile(optimizer='rmsprop',
                    loss='binary_crossentropy',
                    metrics=METRICS)
```

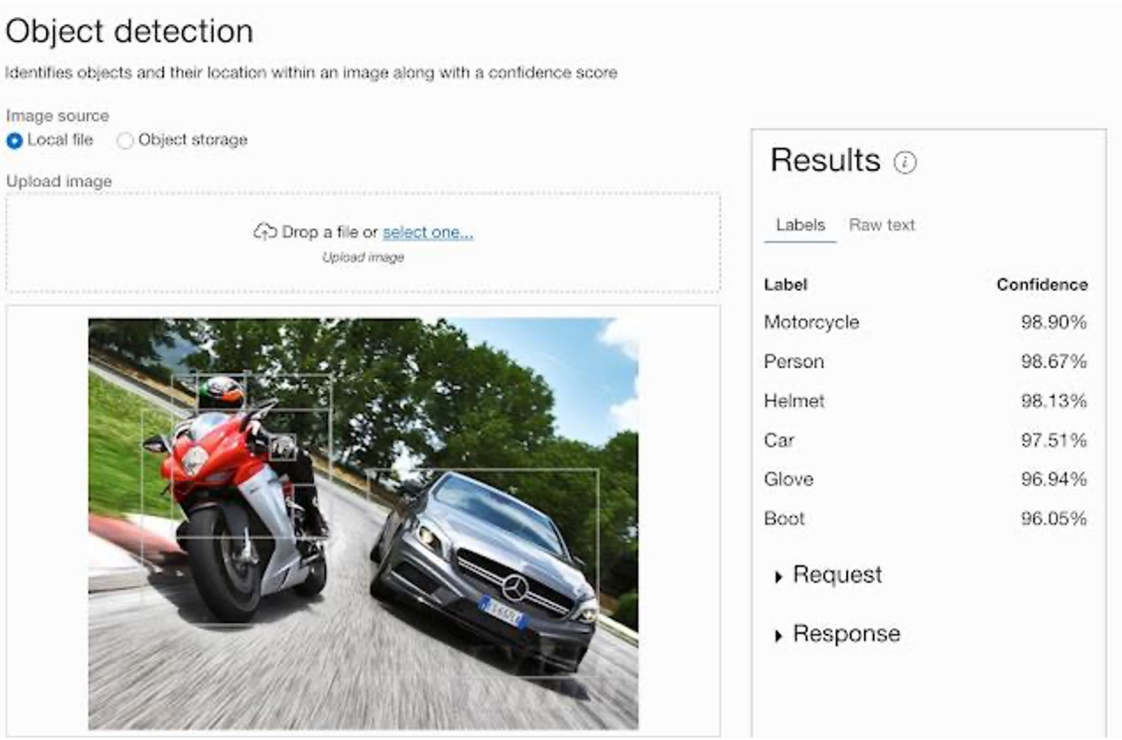
```
[73]: print("Testing set los: \t\t" + str(res_loss))
      print("Testing set accuracy: \t" + str(res_acc*100) + "%")
      print("Testing set precision: \t"  + str(res_prec*100) + "%")
      print("Testing set recall: \t"  + str(res_rec*100) + "%")
      print("Testing set AUC: \t\t"  + str(res_auc*100) + "%")
```

```
Testing set los:        0.24827460944652557
Testing set accuracy:   92.62820482254028%
Testing set precision:  91.54589176177979%
Testing set recall:     97.1794843673706%
Testing set AUC:        97.02881574630737%
```

OCI Data Science is a fully managed and serverless platform for data science teams to build, train, and manage machine learning models using Oracle Cloud Infrastructure.

Qubix

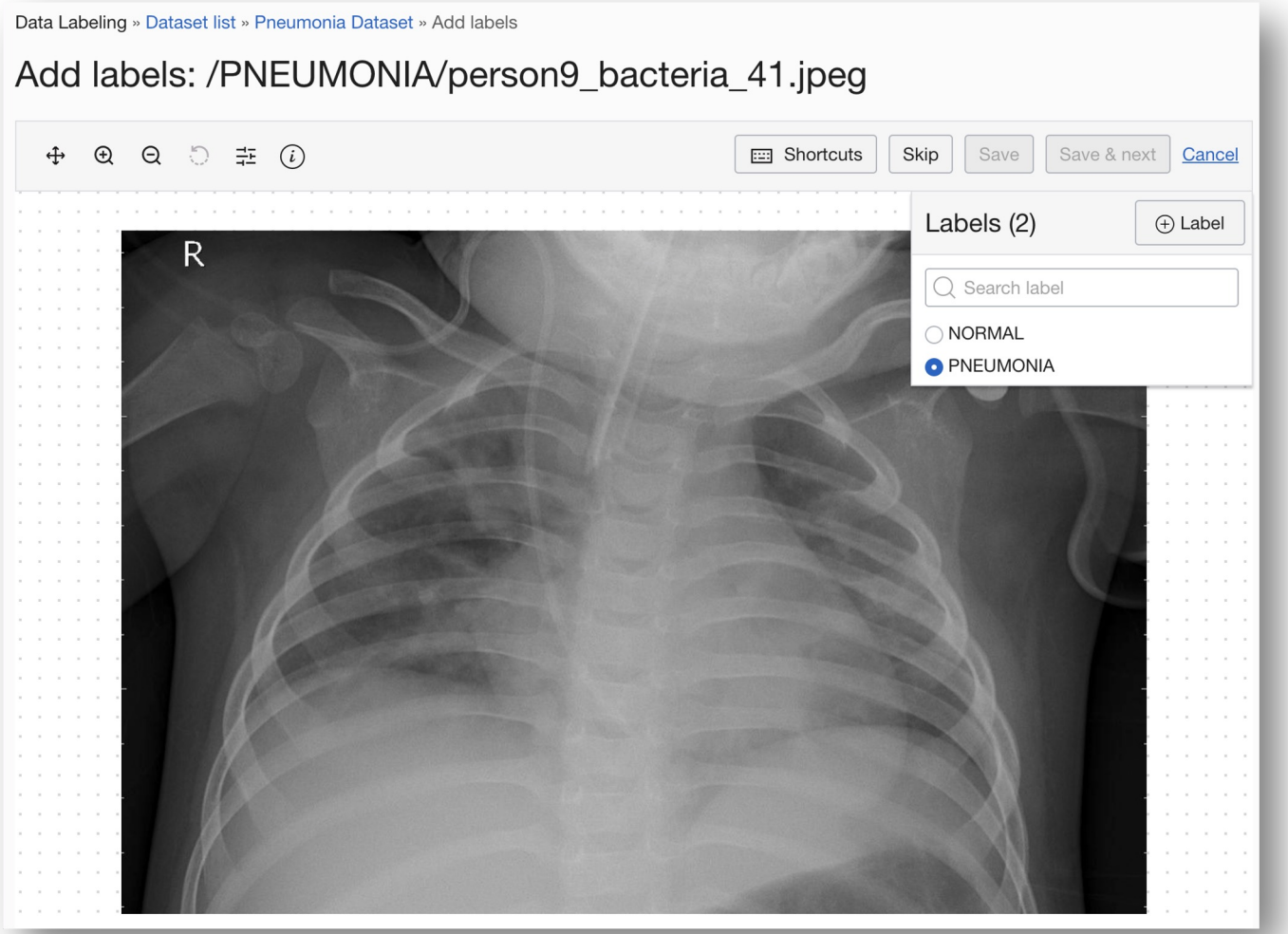
Vision AI Service



- OCI Vision is a serverless, multi-tenant service, accessible using the Console, or over REST APIs.
- Upload images to detect and classify objects in them.
- Process images in batch using asynchronous API endpoints

Qubix

But before anything else, data can be labeled using Data Labeling service



- Label images one by one manually or
- Run bulk labeling utility to label images all in one go

```
import dls_list_records
import dls_create_annotation
import sys
from config import *
import datetime
import labeling_schemes.first_letter as first_letter
import labeling_schemes.first_match as first_match

def main():
        num_records = list_records_limit
        count_batches=1
        count_records_total=0
        while num_records == list_records_limit: # if num_records < list_records_limit,
that would indicate the last loop i.e. batch
                names, ids, num_records = dls_list_records.main()
                count_records_in_batch=0
                for n in names:
                        if labeling_algorithm == "first_match":
                                first_match.main(name=n,
record_id=ids[count_records_in_batch])
                        elif labeling_algorithm == "first_letter":
                                first_letter.main(name=n,
record_id=ids[count_records_in_batch])
                        count_records_in_batch+=1
                        count_records_total+=1
                        print("current time: " + str(datetime.datetime.now()))
                        print("current batch #: " + str(count_batches))
                        print("# records labeled in current batch: " +
str(count_records_in_batch))
                count_batches+=1
        count_batches-=1
        print("-----")
        print("TOTALS:")
        print("current time: " + str(datetime.datetime.now()))
        print("# batches: " + str(count_batches))
        print("# records labeled: " + str(count_records_total))
        print()
main()
```

Qubix

… and then simply train and use the model

Pneumonia Image Classification Model

Edit | Move Resource | Delete

Model information | Tags

General information

Description:

Model type: IMAGE_CLASSIFICATION

OCID: ...uocrra Show Copy

Training data asset: ...lp3jxabotz34igu3yq Show Copy

Created by: oracleidentitycloudservice/ziga.vaupot@qubix.com

Created: Fri, May 20, 2022, 14:38:40 UTC

Max Training duration: Recommended mode (24 hours)

Training metrics

Precision: 0.9737

Recall: 0.9652

F1 score: 0.9694

Total images: 5060

Test images: 506

Trained duration in hours: 5.5525

Analyze

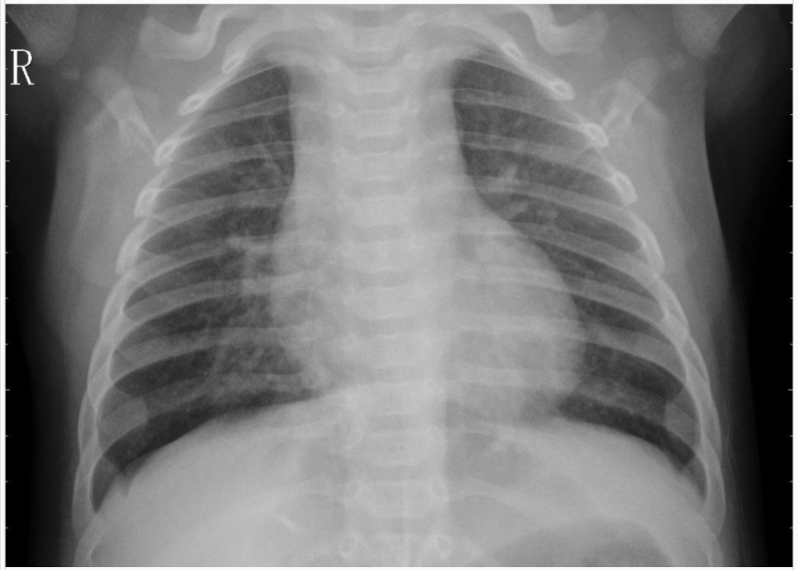
Analyze an image to test the newly trained model.

Image source

○ Local file ◉ Object storage

Enter Image URL

https://objectstorage.eu-frankfurt-1.oraclecloud.com/n/frly8pi3k85f/b/Pneumonia/o/trair Upload



Images

Selected: person1000_virus_1681.jpeg

Labels

Label	Confidence
PNEUMONIA	98.20%
NORMAL	1.40%

▾ Request

JSON Copy


```
{
  "compartmentId": "ocid1.compartment.oc1.
  "image": {
    "source": "OBJECT_STORAGE",
    "namespaceName": "frly8pi3k85f",
    "bucketName": "Pneumonia",
    "objectName": "train/PNEUMONIA/person1
  },
  "features": [
    {
      "modelId": "ocid1.aivisionmodel.oc1.
      "featureType": "IMAGE_CLASSIFICATION
      "maxResults": 5
    }
  ]
}
```


▾ Response

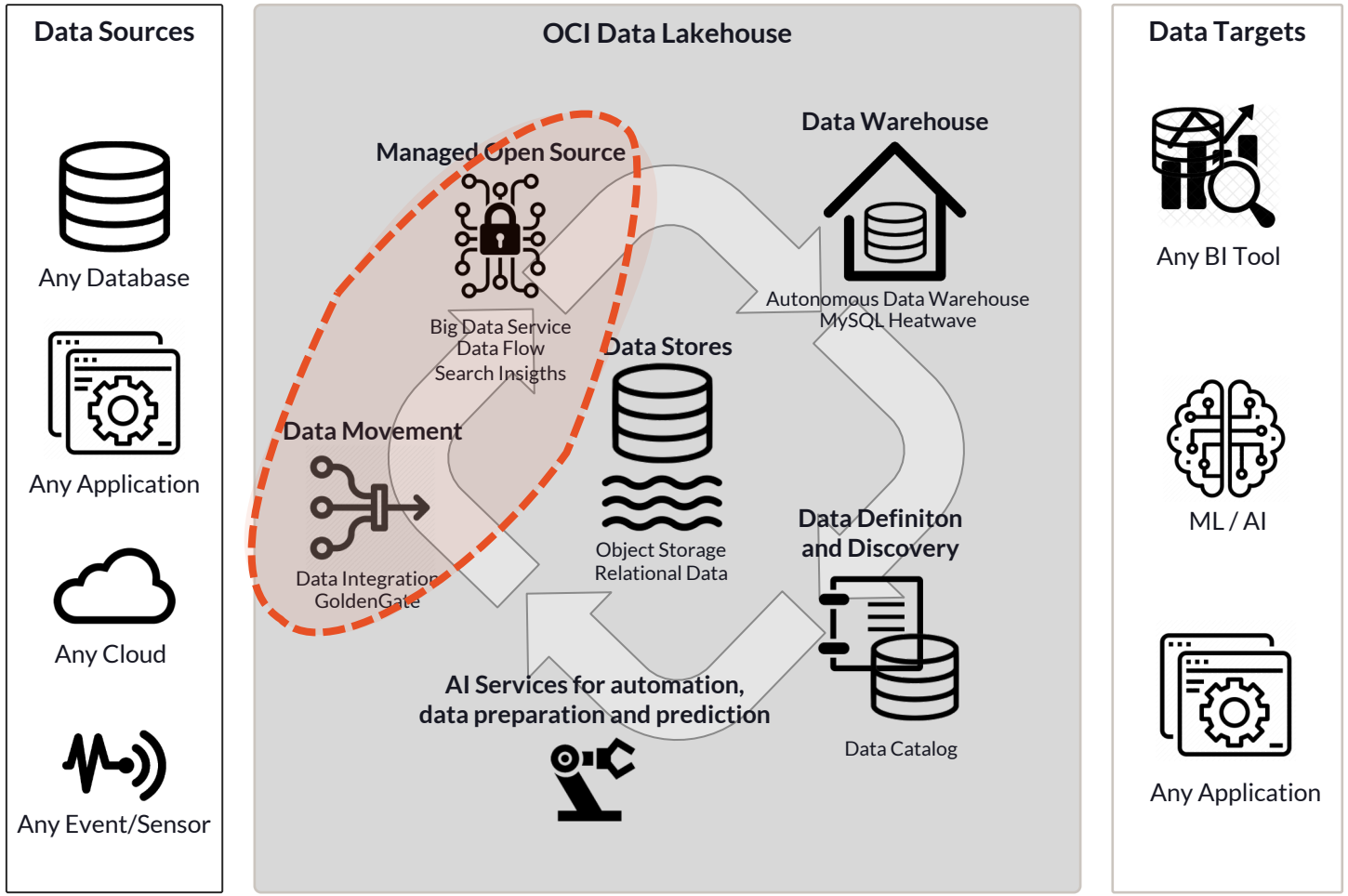
JSON Copy


```
{
  "imageObjects": null,
  "labels": [
    {
      "name": "PNEUMONIA",
      "confidence": 0.9820319
    },
    {
```

Data Lakehouse
Streaming

Qubix

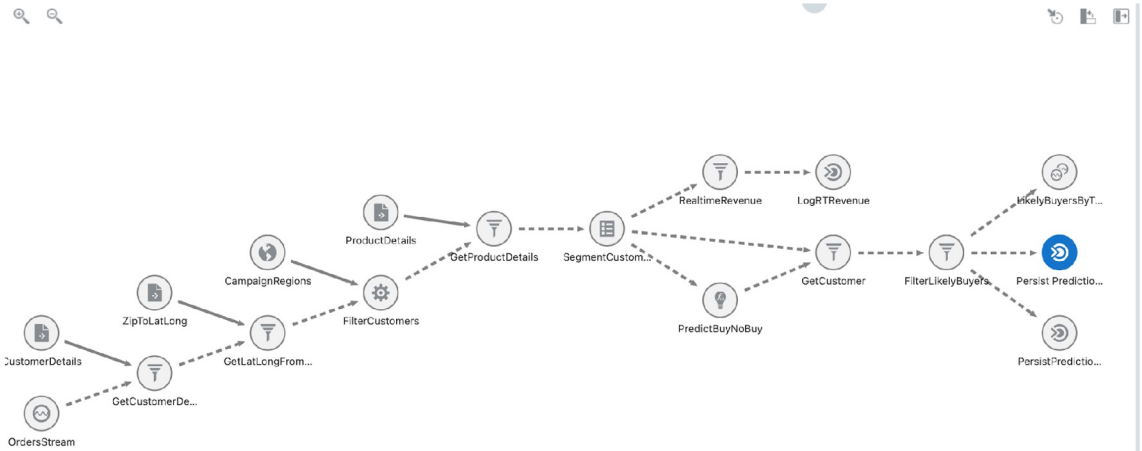
Streaming

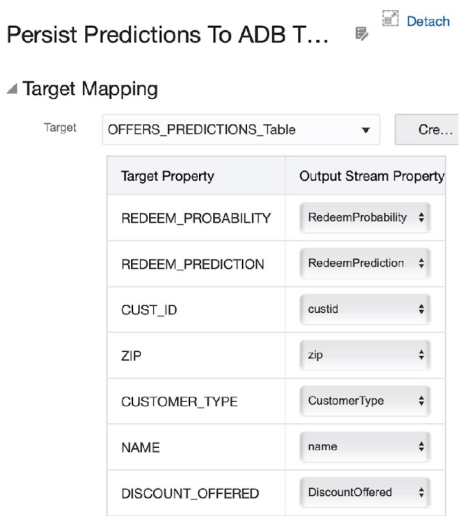


- **OCI Data Flow**
  - Fully managed serverless Apache Spark service that supports Spark Streaming.
  - Customers can now use OCI Data Flow to do cloud scale ETL on their continuously produced stream data.
- **GoldenGate Stream Analytics**
  - Creation of custom operational dashboards that provide real-time monitoring and analyses of event streams in an Apache Spark-based system.
  - Customers can identify events of interest in their Apache Spark-based system, execute queries against those event streams in real time and drive operational dashboards or raise alerts based on that analysis.

Qubix

Goldengate Stream Analytics





Persist Predictions To ADB Table - Live Output

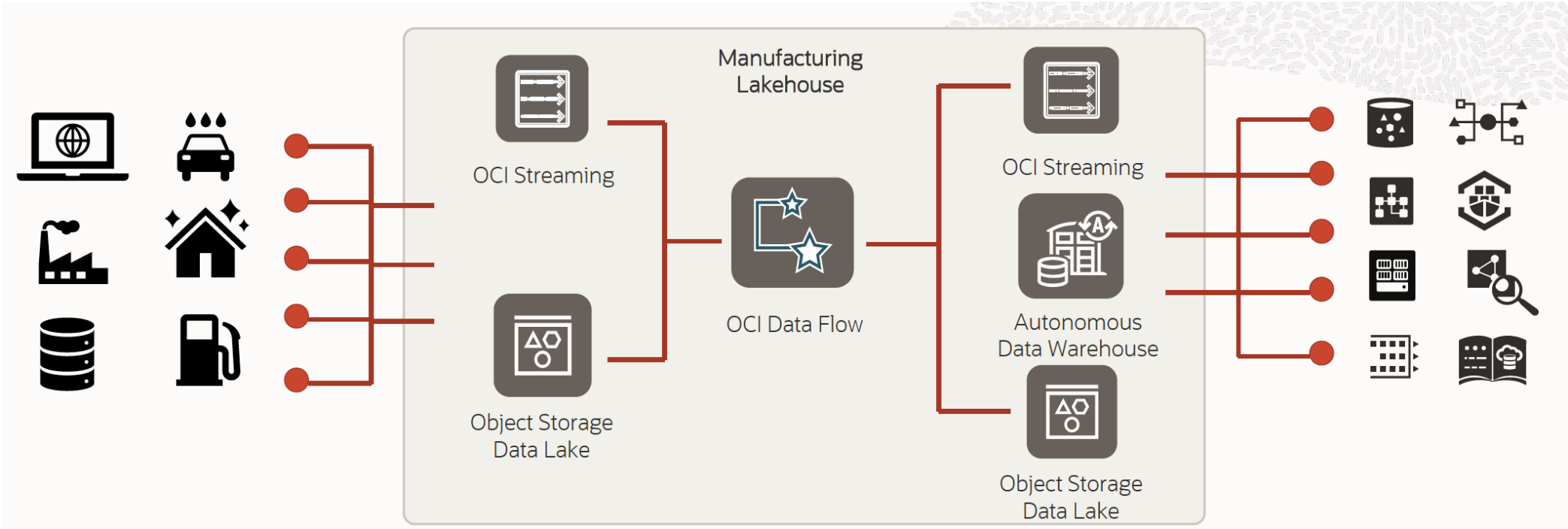
Columns | Detach

REDEEM_PROBABILITY	REDEEM_PREDICTION	CUST_ID	ZIP	CUSTOMER_TYPE	NAME	DISCOUNT_OFFERED
1	1	325	90017	Platinum	Sansa Rivers	25
0.900000000000000022204604925031308084726333618164062 5	1	812	90006	BRONZE	Jessica Waters	0
0.599999999999999977795539507496869191527366638183593 75	1	756	90012	BRONZE	Don Torino	0
1	1	883	90015	Platinum	Chris Zeus	25
1	1	59	90020	Platinum	Catelyn Bara	25
1	1	946	94583	Platinum	Petra Danza	25
1	1	179	90016	Platinum	Jorge Martell	25
0.599999999999999977795539507496869191527366638183593 75	1	748	90004	Platinum	Petra Frey	25

- Pipeline of stream data
- Data enrichment
- Data visualization
- Machine Learning
- Writing augmented data to targets

Qubix

Oracle Streaming & Data Flow



Demo

Machine Learning on Streaming Data with OCI Data Flow

Qubix

About the demo

- Use case: predicting remaining useful life of equipment
  - achieved with Data Flow (fully managed serverless Spark as a service)
  - streaming (continuous data streams)
- Other similar use cases:
  - real time fraud detection (financial services)
  - churn prediction (telecommunications)

Qubix

Application details

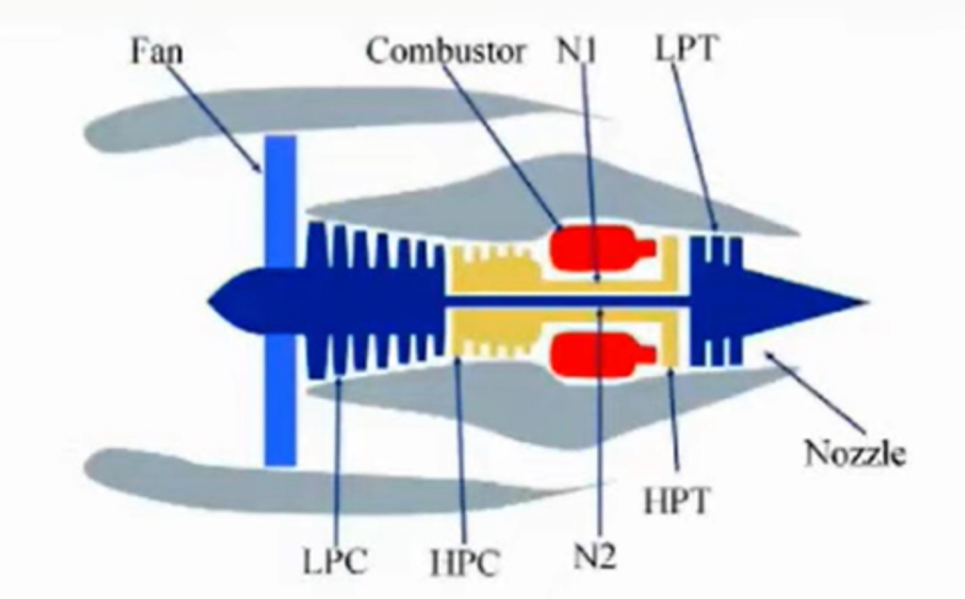
- Three applications used:
  - model trainer (constructs and trains a machine learning model)
  - simulator (generates sensor data, sends via stream)
  - predictor (uses simulated data from stream and trained model to predict remaining useful life)
- Predictions can be used in further streams or recorded into a database

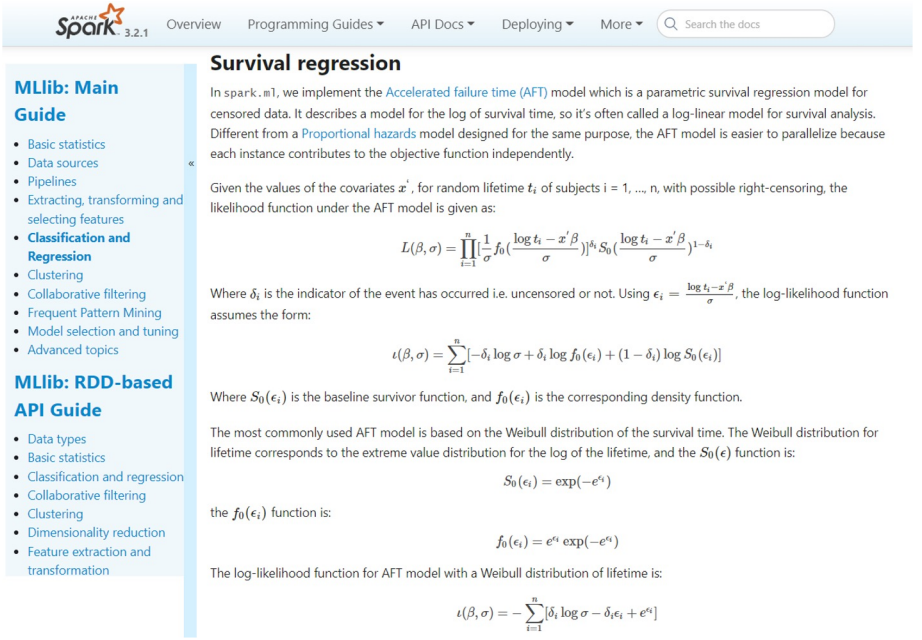
Color	RUL	Status, Action
Red	0 - 30 days	Degraded, perform maintenance
Yellow	30 - 60 days	Degrading, watch
Green	> 60 days	Healthy, no action needed

Qubix

Machine learning model and dataset

- ML model: Spark ML Survival Regression Model - Accelerated Failure Model (AFT)
  - available in the Spark ML library
- Dataset: Turbofan Engine Simulation Degradation Dataset
  - from NASA, contains data from a simulation of engine degradation of rocket parts





Spark 3.2.1 Overview Programming Guides API Docs Deploying More Search the docs

MLlib: Main Guide

- Basic statistics
- Data sources
- Pipelines
- Extracting, transforming and selecting features
- **Classification and Regression**
- Clustering
- Collaborative filtering
- Frequent Pattern Mining
- Model selection and tuning
- Advanced topics

MLlib: RDD-based API Guide

- Data types
- Basic statistics
- Classification and regression
- Collaborative filtering
- Clustering
- Dimensionality reduction
- Feature extraction and transformation

Survival regression

In `spark.ml`, we implement the Accelerated failure time (AFT) model which is a parametric survival regression model for censored data. It describes a model for the log of survival time, so it's often called a log-linear model for survival analysis. Different from a Proportional hazards model designed for the same purpose, the AFT model is easier to parallelize because each instance contributes to the objective function independently.

Given the values of the covariates $x^{'}$, for random lifetime t_i of subjects i = 1, ..., n, with possible right-censoring, the likelihood function under the AFT model is given as:

$$L(\beta,\sigma)=\prod_{i=1}^{n}[\frac{1}{\sigma}f_0(\frac{\log t_i-x^{'}\beta}{\sigma})]^{\delta_i}S_0(\frac{\log t_i-x^{'}\beta}{\sigma})^{1-\delta_i}$$

Where δ_i is the indicator of the event has occurred i.e. uncensored or not. Using $\epsilon_i=\frac{\log t_i-x^{'}\beta}{\sigma}$, the log-likelihood function assumes the form:

$$\iota(\beta,\sigma)=\sum_{i=1}^{n}[-\delta_i\log\sigma+\delta_i\log f_0(\epsilon_i)+(1-\delta_i)\log S_0(\epsilon_i)]$$

Where $S_0(\epsilon_i)$ is the baseline survivor function, and $f_0(\epsilon_i)$ is the corresponding density function.

The most commonly used AFT model is based on the Weibull distribution of the survival time. The Weibull distribution for lifetime corresponds to the extreme value distribution for the log of the lifetime, and the $S_0(\epsilon)$ function is:

$$S_0(\epsilon_i)=\exp(-e^{\epsilon_i})$$

the $f_0(\epsilon_i)$ function is:

$$f_0(\epsilon_i)=e^{\epsilon_i}\exp(-e^{\epsilon_i})$$

The log-likelihood function for AFT model with a Weibull distribution of lifetime is:

$$\iota(\beta,\sigma)=-\sum_{i=1}^{n}[\delta_i\log\sigma-\delta_i\epsilon_i+e^{\epsilon_i}]$$

Qubix

Enabling the Predictive Enterprise