This session will be recorded

ORACLE

Oracle Machine Learning

AskTOM Office Hours – Feature Highlight

Exporting and Importing Oracle Machine Learning (OML) Models

with Ethan Shmargad and Sherry LaMonica

Host: Mark Hornick

Product Management, Oracle Machine Learning

Migrating Oracle Machine Learning Models from Development to Production

Speakers: Ethan Shmargad and Sherry LaMonica

Even when machine learning models produce useful results, they only add value to an organization when their insights are regularly available to end users. Traditional solutions for migrating models across development, staging, and production databases offer challenges in model storage and management, causing delays and failures in model deployment. Join us for this Office Hours session to learn how Oracle Machine Learning streamlines the process of exporting and importing models across databases and platforms.

Poll #1: Using model export and import

What is your familiarity with in-database model export and import?

(select all that apply)

- I have exported/imported OML models from ODB to ODB, ADB to ADB, or between ODB and ADB
- I have exported in–database models using the API for import to OML Services
- I have used the OML Models UI or AutoML UI to deploy models to OML Services
- I’ve exported models as a custom backup
- I’m new to OML model export and import

Oracle Machine Learning

Exporting and Importing Oracle Machine Learning (OML) Models

Ethan Shmargad and Sherry LaMonica

Agenda

- Introduction
- Migration options for OML models
- Code examples & best practices
- Architecture scenarios
- Demonstration
- Q&A

Exporting and importing Oracle Machine Learning Models

Speakers: Ethan Shmargad and Sherry LaMonica

Traditional solutions for migrating models across development, staging, and production databases present challenges in model storage and management, often causing delays and failures in model deployment.

The focus of today's session is to show how Oracle Machine Learning streamlines the process of exporting and importing models across databases and platforms, including both Autonomous Database and Oracle Database on-premises.

Options for Exporting and Importing OML Models

Data Pump

expdp
impdp

Full database or schema export and import

DBMS_DATA_MINING package

EXPORT_MODEL
IMPORT_MODEL

PL/SQL procedures based on data pump for individual or sets of models

EXPORT_**SER**MODEL
IMPORT_**SER**MODEL

PL/SQL procedures for individual model serialization

Data Pump Utility

expdp/impdp command line utility

Traditional method to migrate database objects

Features:

- Exports/imports **all** models in a database or schema to/from a dump file set with model detail views retained
- Supports cross-database and schema model export and import for all OML in-database models

Use Cases

- Migrating one or more schemas
- Migrating databases

Data Pump

Export and import entire schema containing OML models

Export from source schema (OMLUSER)

```
# Data pump command run by the OML user to
# export all database objects from the current
# schema to dump file schema.dmp

$ expdp omluser/password
        dumpfile=schema.dmp
        directory=DATA_PUMP_DIR
```

The dump file set exported to the data pump directory contains all objects from the source schema, including the OML models. The number of files in the dump set is determined by the size of the objects being exported with a maximum file size of 2 GB. If the size of the exported object is greater than the maximum file size, additional files are created. The maximum file size can be specified in the optional *filesize* parameter. The dump file name is expanded to *file01.dmp*, even if there is only one file in the set. Any additional files are named sequentially as *file02.dmp*, *file03.dmp,* and so forth.

Import into target schema or database

```
# Data pump command run by the target OML user to
# import the dump file schema.dmp

$ impdp omluser2/password
                  dumpfile=schema01.dmp
                  directory=DATA_PUMP_DIR
                  remap_schema=OMLUSER:OMLUSER2
                  remap_tablespace=USERS:USERS2
```

The dump file set is copied to the target environment by the OML user and imported using impdp. The schema name and tablespace are remapped from source to target to match the target user's schema name and default tablespace.

DBMS_DATA_MINING package

export_model/import_model

PL/SQL procedures to export and import models based on data pump

Features

- Exports/imports **selected** models to/from a dump file set
  - Model detail views retained
  - Supports individual models, lists of models, or models filtered by mining function or algorithm
- Supports cross-database and schema model export and import for all OML in-database models

Use Cases

- For export/import of specific models
- Preserving model detail views in the target environment
- Target database is the same version or newer than the source database

DBMS_DATA_MINING package

export_model/import_model

Export model from source schema

```
-- EXPORT_MODEL command run by the OML user to
-- export model DT_MODEL from the current
-- schema to dump file set DT_MODEL.dmp

BEGIN DBMS_DATA_MINING.EXPORT_MODEL(
         filename =>     'DT_MODEL.dmp',
         directory =>    'DATA_PUMP_DIR',
         model_filter => 'name in(''DT_MODEL'')');
END;
```

The number of files in the dump file set is determined by the size of the models being exported, with a maximum file size of 50 MB. One can specify the file size in the optional *filesize* parameter. If the size of the exported model(s) is greater than the maximum file size, additional files are created. The dump file name is expanded to *file01.dmp*, even if there is only one file in the set. Any additional files are named sequentially as *file02.dmp*, *file03.dmp,* and so forth.

Import model into target schema or database

```
-- IMPORT_MODEL command run by the target OML
-- user to import the dump file set.


BEGIN DBMS_DATA_MINING.IMPORT_MODEL(
                   filename=>     'DT_MODEL01.dmp',
                   directory=>    'DATA_PUMP_DIR',
                   schema_remap=> 'OMLUSER:OMLUSER2',
                   tablespace_remap => 'USERS:USERS2');
END;
```

Dump file containing a single model DT_MODEL is copied to the target environment by the OML user and imported using IMPORT_MODEL. Multiple models also supported. The schema name and tablespace are remapped from source to target to match the target user's schema name and default tablespace.

DBMS_DATA_MINING package

export_sermodel/import_sermodel

Lightweight serialized model format

Features

- Exports/imports **individual** models to/from a serialized BLOB
  - Does not preserve model views, **only** metata required for scoring
  - Serialized models can be imported into another database or schema or deployed to OML Services
- Ease of deployment using most **robust** and **compact** representation
- Supports efficient transfer since footprint is up to 300 times smaller, transfer rate increased by up to 300 times.

Use Cases

- When model needed for scoring/inference, not model details
- Does **not** support models that don’t score (use prediction operators)
- Used for import to OML Services

DBMS_DATA_MINING package

export_sermodel/import_sermodel

Export model from source schema

```
-- EXPORT_SERMODEL command run by the OML user to
-- export model GLM_MODEL from the current
-- schema to serialized BLOB GLM_MODEL.mod

DECLARE
  MODNAME VARCHAR2(100) := 'GLM_MODEL';
  SER_MODNAME VARCHAR2(100);
  BLOB_MODEL BLOB;
BEGIN
  DBMS_LOB.CREATETEMPORARY(BLOB_MODEL, FALSE);
  SELECT CONCAT(MODNAME,'.mod') INTO SER_MODNAME from dual;
  DBMS_DATA_MINING.EXPORT_SERMODEL(BLOB_MODEL,'GLM_MODEL’);
  -- custom procedure to save the model to a file
  write_serialized_model(BLOB_MODEL,
                      'DATA_PUMP_DIR’,
                       SER_MODNAME);
  DBMS_LOB.FREETEMPORARY(BLOB_MODEL);
END;
```

Import model into target schema or database

```
-- IMPORT_SERMODEL command run by the target OML
-- user to import the serialized model

-- table to temporarily store the model
CREATE TABLE MODEL_IMPORT(MY_MODEL BLOB)

DECLARE
  BLOB_MODEL BLOB;
BEGIN
  INSERT INTO MODEL_IMPORT values
(BFILENAME('MYDIR','GLM_MODEL.mod'));
  commit;
  SELECT MY_MODEL INTO BLOB_MODEL FROM MODEL_IMPORT;
DBMS_DATA_MINING.IMPORT_SERMODEL(BLOB_MODEL, 'GLM_MODEL');
END;
```

Serialized model is written to a BLOB in the data pump directory. The model is copied to the target environment by the OML user and imported using IMPORT_SERMODEL.

Model Export and Import APIs

Deployment architectures and best practices

API	Description	Deployment Architecture	Best Practices
Data Pump expdp Impdp	Oracle Data Pump migrates entire databases or schemas containing OML models	• All-in-one • Split dev/prod	Migrating an entire database or schema
DBMS_DATA_MINING EXPORT_MODEL IMPORT_MODEL	PL/SQL procedures based on Data Pump	• All-in-one • Split dev/prod • Split workloads	Migrating selected models, when model details are needed, and when target database is the same version or newer than the source database
DBMS_DATA_MINING EXPORT_SERMODEL IMPORT_SERMODEL	PL/SQL procedures to serialize models. Designed as a lightweight approach to support scoring	• All-in-one • Split dev/prod • Split workloads • Real-time scoring	Created for scoring/inference Does not support models that don’t score

Top 5 Architectures for Enterprise Machine Learning with Oracle Database
https://bit.ly/42aooSz

Architectural Scenario

Separate source and target databases

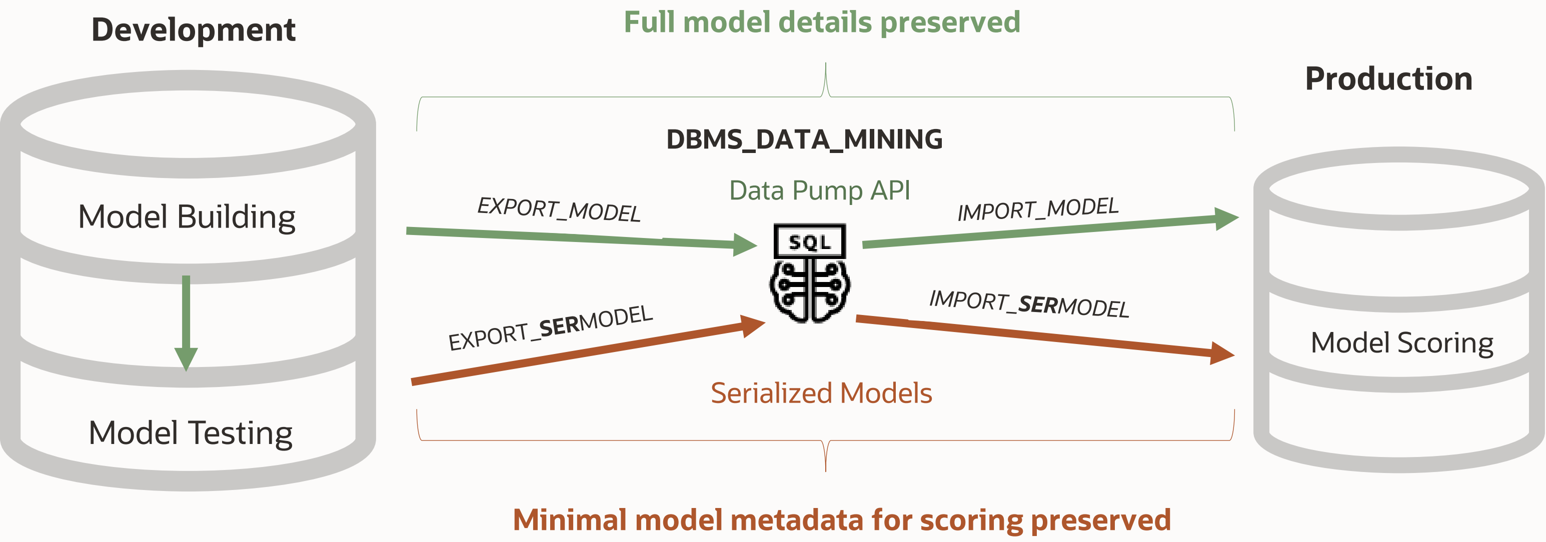
Typical split architecture between development and production deployment

export_model/import_model

- Specific models or a list of models, preserving full details and metadata
- Ideal for model analysis requiring complete model details

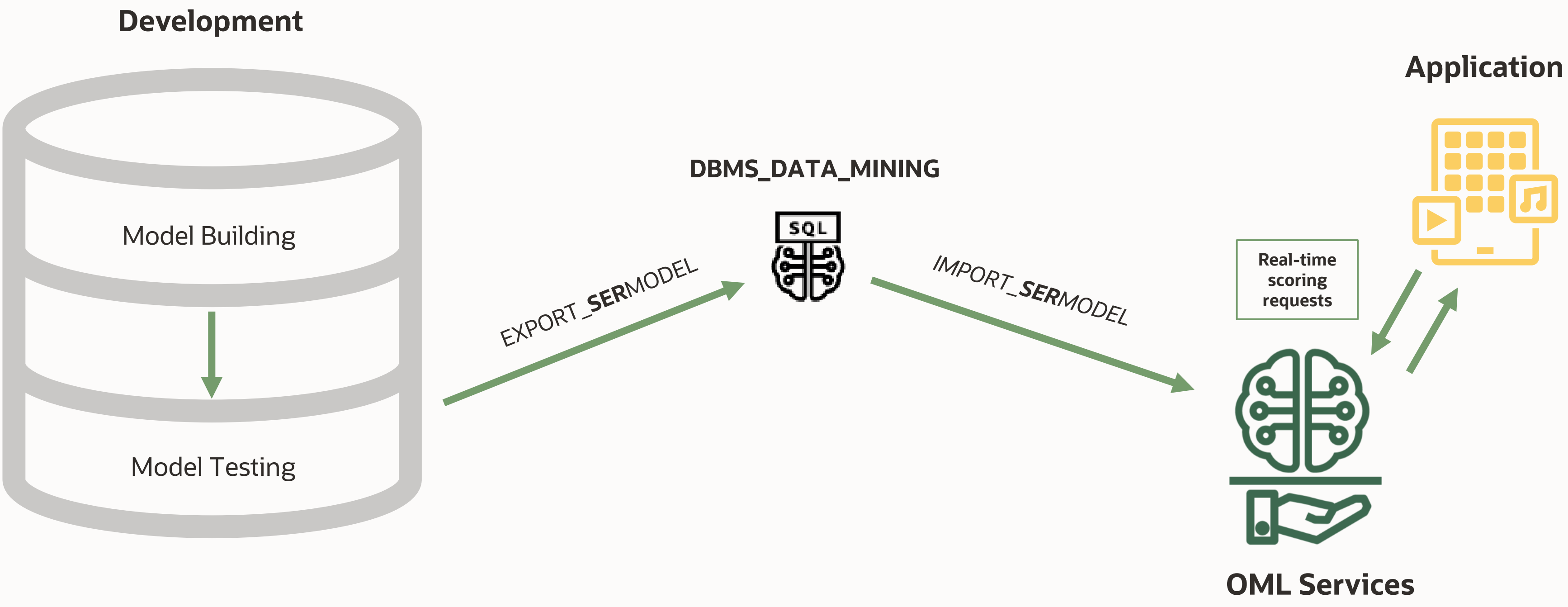
export_sermodel/import_sermodel

- Designed for scoring
- Individual models, preserving only metadata needed for scoring
- Facilitates lightweight model transfer within the split prod/dev architecture
- Scoring engine can be the database or OML Services



Deployment to OML Services

Serialized models deployed for real-time scoring Autonomous Database via REST



Demo

Poll #2: Session feedback

How has this session helped your understanding of OML model export and import?

- I was new to this topic, but now have a good understanding of how to apply this capability
- I'm more confident, but still have questions about how best to use this capability
- I feel confident to start using export and import in my next project
- I was already using this capability and this session reinforced or enhanced my understanding

For more information…

OML Webpage

https://oracle.com/machine-learning

Top 5 Deployment Architectures Blog

https://bit.ly/42aooSz

OML GitHub Repository

https://bit.ly/omlgithub

OML Office Hours

https://bit.ly/omlofficehours

OML Documentation

https://docs.oracle.com/en/database/oracle/machine-learning/index.html

Q & A

Thank you

ethan.shmargad@oracle.com

sherry.lamonica@oracle.com