



# Python Machine Learning in Oracle Autonomous ADB

Natwest Developer Session – HOL Details

July 2021

# Introduction



Interfaces for 3 popular data science languages: SQL, R, and Python

Code-free AutoML interface on Autonomous Database

Model Deployment and Management, Cognitive Text

Collaborative notebook environment based on Apache Zeppelin with Autonomous Database

SQL Developer extension to create, schedule, and deploy ML solutions through a drag-and-drop interface

ML for the big data environment from R with scalable algorithms

# Oracle Machine Learning

**OML4SQL**

**OML Notebooks**

**OML4R**

**Oracle Data Miner**

**OML4Py**

**OML4Spark**

**OML AutoML UI**

**OML Services**





# Oracle Machine Learning Notebooks

Autonomous Database as a Data Science Platform



## Collaborative UI

- Based on Apache Zeppelin
- Supports data scientists, data analysts, application developers, DBAs with SQL and Python
- Easy notebook sharing
- Permissions, versioning, and scheduling of notebooks

## Included with Autonomous Database

- Automatically provisioned and managed
- In-database algorithms and analytics functions
- Explore and prepare, build and evaluate models, score data, deploy solutions

The screenshot shows the Oracle Machine Learning Notebooks interface with three code cells and their results:

- Overloaded data visualization functions**:  
Code: 

```
%python  
import matplotlib.pyplot as plt  
plt.style.use('seaborn')  
plt.figure(figsize=[10,5])  
  
oml.graphics.boxplot(iris[:, :4], notch=True, showmeans = True,  
                     labels=iris.columns[:4])  
plt.title('Distribution of IRIS Attributes')  
plt.xlabel('cm')  
  
Text(72.625, 0.5, 'cm')
```

  
Result: A boxplot titled "Distribution of IRIS Attributes" showing the distribution of Sepal Length, Sepal Width, Petal Length, and Petal Width in centimeters.
- histogram**:  
Code: 

```
%python  
oml.graphics.hist(iris['SEPAL_LENGTH'], bins=10, color='red',  
                  linestyle='solid', edgecolor='black')  
  
plt.title('Sepal Length variation in IRIS data set')  
plt.xlabel('Sepal Length')  
plt.ylabel('# of iris instances')  
  
plt.show()
```

  
Result: A histogram titled "Sepal Length variation in IRIS data set" showing the frequency of Sepal Length values.
- Create derived variables**:  
Code: 

```
%python  
is_large_petal = (iris['PETAL_LENGTH'] > 5.0) & (iris['PETAL_WIDTH'] > 2.0)
```





# Oracle Machine Learning for Python

Supported in Oracle Autonomous Database with OML Notebooks

Use Oracle Database as HPC environment

- Explore, transform, and analyze data faster and at scale

Use in-database parallelized and distributed ML algorithms

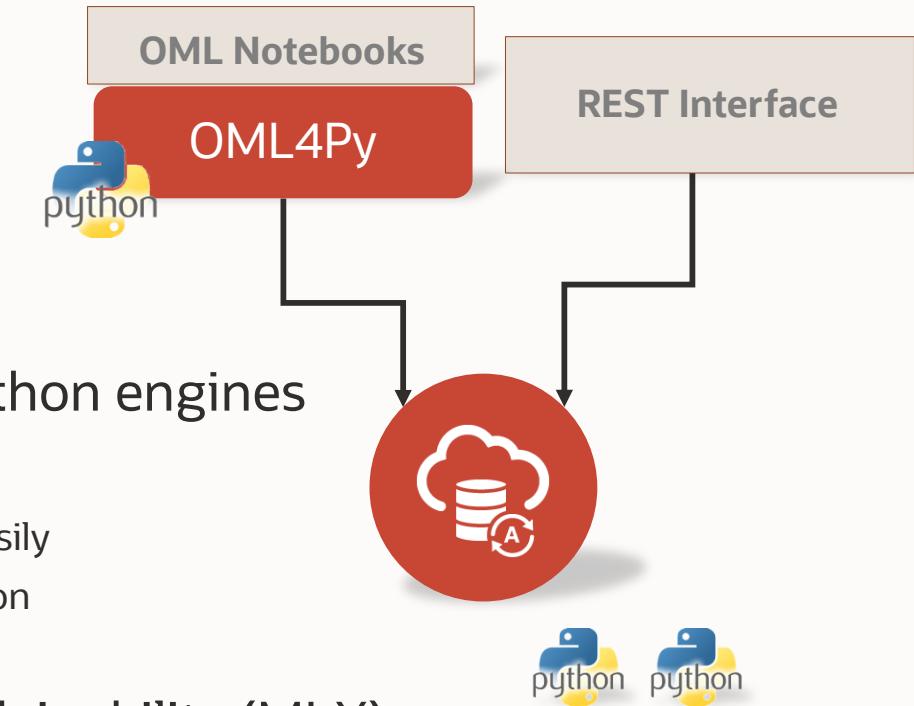
- Build more models on more data, and score large volume data – faster
- Use in-database algorithms from OML4SQL via well-integrated Python API
- Increased productivity from automatic data preparation, partitioned models, and integrated text mining capabilities

Run Python scripts in database-spawned and controlled Python engines and manage Python objects in-database

- Collaborate: hand-off data science products from data scientist to developers easily
- Run user-defined functions in data-parallel, task-parallel, and non-parallel fashion
- Return structured and image results in Python and REST API

New automated machine learning (AutoML) and model explainability (MLX)

- Enhance data scientist productivity and enable non-experts to use and benefit from machine learning
- Algorithm selection, feature selection, hyperparameter tuning, model selection
- Model-agnostic identification of important features that impact model predictions



# Labs overview



# **Lab high-level outline**

**Lab 1:** Getting started with OML4Py

**Lab 2:** Select and manipulate data using the Transparency Layer

**Lab 3:** Using in-database algorithms and models

**Lab 4:** Use datastores to store Python objects

**Lab 5:** Run user-defined functions using Embedded Python Execution

**Lab 6:** Use AutoML

**Lab 6:** Bonus Round - Use OML AutoML UI

# Lab 1: Getting started with OML4Py

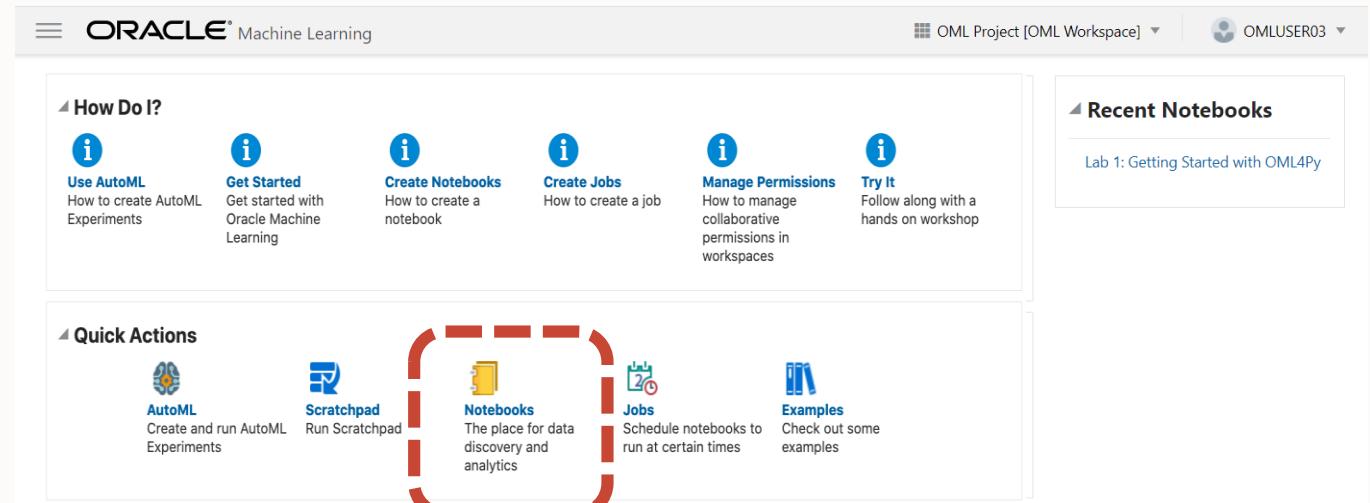


# Lab 1 Getting Started with OML4Py

Step 1: In your browser (Chrome or Firefox), go to the provided URL

Step 2: Log in using provided username and password

Step 3: Click “Notebooks”



Step 4: Click the notebook for  
"Lab 1: Getting Started with OML4Py"

The screenshot shows the 'Notebooks' list page. At the top, there's a toolbar with buttons for Edit, Create, Duplicate, Save as Template, Delete, Import, Version, and Search. Below the toolbar, there's a table with columns for Name, Comment, Last Update, Updated By, and Connection Group. Two rows are visible: 'Lab 1: Getting Started with OML4Py' (last updated 1/17/21 2:15 AM by OMLUSER03) and 'Lab 2: Data Selection and Manipulation using the...' (last updated 1/17/21 12:18 AM by OMLUSER03). The 'Name' column for both rows is highlighted with a red dashed box.

Name	Comment	Last Update	Updated By	Connection Group
Lab 1: Getting Started with OML4Py		1/17/21 2:15 AM	OMLUSER03	Global
Lab 2: Data Selection and Manipulation using the...		1/17/21 12:18 AM	OMLUSER03	Global

# Lab 1: Getting Started with OML4Py

The screenshot displays the Oracle Machine Learning interface with several toolbars and status elements:

- Top Toolbar:** Includes "Show/Hide Output" and "Clear Notebook" buttons.
- Second Toolbar:** Includes "Run All", "Show/Hide Code", "Clear Output", "Export Notebook", and "Search Code" buttons.
- Third Toolbar:** Includes icons for Run, Stop, Refresh, Notebook, Cell, and Cell Type.
- User Interface Elements:**
  - "Connected Users": Shows 1 user connected.
  - "List Shortcuts": Shows "OMLUSER03" connected.
  - "Interpreter Bindings": Shows "OML Project [OML Workspace]" and "OMLUSER03" status.
  - "Paragraph Status": Shows "FINISHED".
  - "Run Hide ParagraphOutput": Buttons for running and hiding paragraph output.
  - "More Features": A dropdown menu with options like "Show Editor" and "Width".
- Bottom Status Bar:** Shows "FINISHED" and a timestamp: "20210117-001841\_797132823".
- Right Panel:** A detailed list of keyboard shortcuts for various actions.

# Lab 2: Select and manipulate data using the Transparency Layer



# Transparency Layer

In-database performance – indexes, query optimization, parallelism, partitioning

Leverages proxy objects for database data: *oml.DataFrame*

```
# Create table from Pandas DataFrame data
DATA = oml.create(data, table = 'BOSTON')

# Get proxy object to DB table boston
DATA = oml.sync(table = 'BOSTON')
```

Uses familiar Python syntax to manipulate database data  
Overloads Python functions translating functionality to SQL

```
DATA.shape
DATA.head()
DATA.describe()
DATA.std()
DATA.skew()
```

```
TRAIN, TEST =
    DATA.split()
TRAIN.shape
TEST.shape
```

# Data transfer-related functions

**oml.create(x, table[, oranumber, dbtypes, ... ])**

- Creates a table in Oracle Database from a Pandas DataFrame returning a proxy object

**oml.push(x[, oranumber, dbtypes])**

- Pushes data to Oracle Database creating a temporary table returning a proxy object

**oml.sync(schema=None, regex\_match=False, table=None, view=None, query=None)**

- Creates a DataFrame proxy object in Python that represents an Oracle Database table

**oml.drop([table, view])**

- Drops the named database table or view

**oml.dir()**

- Returns the names of OML objects in the workspace

**oml.cursor()**

- Returns a cx\_Oracle cursor object of the current OML database connection



# In-database scalable aggregation

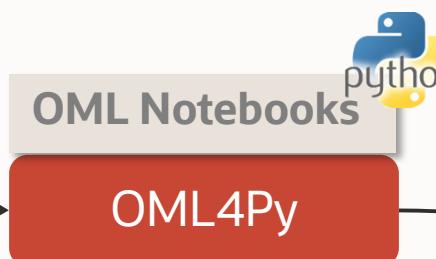
Example using the crosstab function

```
ONTIME_S = oml.sync(table="ONTIME_S")
res = ONTIME_S.crosstab('DEST')
type(res)
res.head()
```

Source data is a DataFrame, ONTIME\_S, which is an Oracle Database table

crosstab() function overloaded to accept OML DataFrame objects and transparently generates SQL for scalable processing in Oracle Database

Returns an 'oml.core.frame.DataFrame' object



```
>>> ONTIME_S = oml.sync(table="ONTIME_S")
>>> res=ONTIME_S.crosstab('DEST')
>>> type(res)
<class 'oml.core.frame.DataFrame'>
>>> res.head()
   DEST  count
0  ABE    237
1  ABI     34
2  ABQ   1357
3  ABY     10
4  ACK      3
```

```
select DEST, count(*)
from ONTIME_S
group by DEST
```

# Functions on OML DataFrame executed in-database

KFold  
append  
columns  
concat  
corr  
count  
create\_view  
crosstab  
cumsum  
describe  
drop  
drop\_duplicates

rename  
round  
select\_types  
shape  
skew  
sort\_values  
split  
std  
sum  
t\_dot  
tail  
types

dropna  
head  
kurtosis  
materialize  
max  
mean  
median  
merge  
min  
nunique  
pivot\_table  
pull

# Lab 2: Select and manipulate data using the Transparency Layer

**Goal:** Become familiar with creating and using DataFrame proxy objects for exploring and transforming database tables and views

Step 1: Import libraries and create OML DataFrame proxy object

Step 2: Select table columns

Step 3: Select table rows

Step 4: Use Pandas DataFrame objects + TIY

Step 5: Use the split and KFold functions + TIY

Step 6: Use the crosstab and pivot\_table functions

Step 7: Use oml.boxplot and oml.hist

Step 8: Create a persistent database table



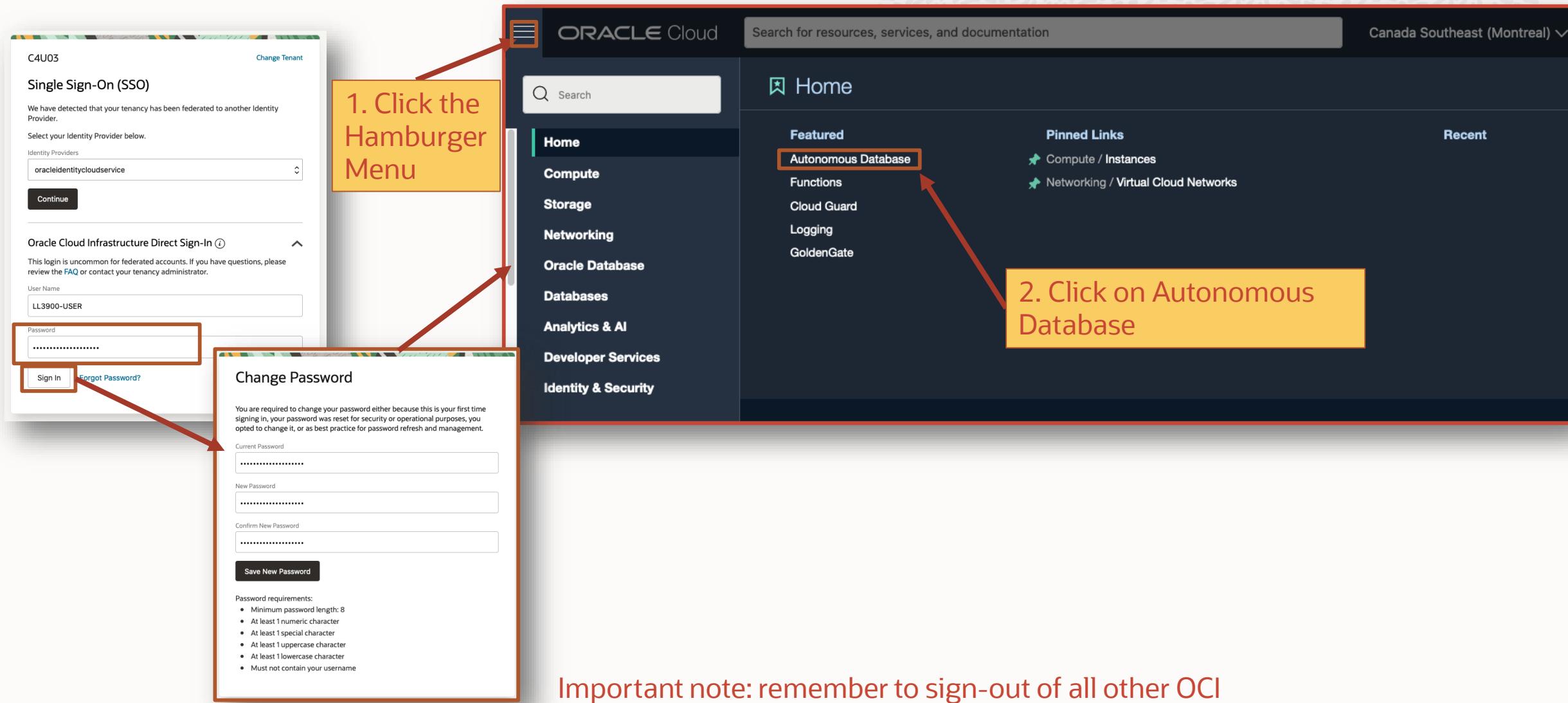
# Quick check on the environment readiness

Contact us via "**Chat to the Panelists**" if your environment is not ready yet, or if you have problems

Have you received the email indicating that your LiveLabs environment is ready to use?



**In the Single Sign-On screen, enter the Password given in the Workshop screen for your user **LL#####-USER**. You will need to change the Password (you can use the same), then click on the Hamburger Menu on the Top left of OCI, then Autonomous Database.**



**Important note: remember to sign-out of all other OCI Account Tabs you might have open**

# Select the specific ADW Compartment indicated in the Workshop Main screen, and then click on the Autonomous Database

The screenshot shows the Oracle Cloud Autonomous Database interface. On the left, there's a sidebar with 'Autonomous Database' selected. Below it, a 'List Scope' section has 'Compartment' selected, showing a tree view of compartments under 'c4u03 (root)'. The 'LL3900-COMPARTMENT' node is highlighted. A red box encloses the main content area titled 'Autonomous Databases in LL3900-COMPARTMENT'. Inside, a table lists one database: 'ADW3900' (Available, No Dedicated, 1 OCPUs, 1 TB). A red box highlights the 'ADW3900' row. A large red arrow points from the 'LL3900-COMPARTMENT' node in the sidebar to the 'ADW3900' row in the table.

**Autonomous Databases in LL3900-COMPARTMENT**

Display Name	State	Dedicated	OCPUs	Storage (TB)
ADW3900	Available	No	1	1

**Autonomous Databases in LL3900-COMPARTMENT**

Display Name	State	Dedicated	OCPUs	Storage (TB)
<a href="#">ADW3900</a>	Available	No	1	1

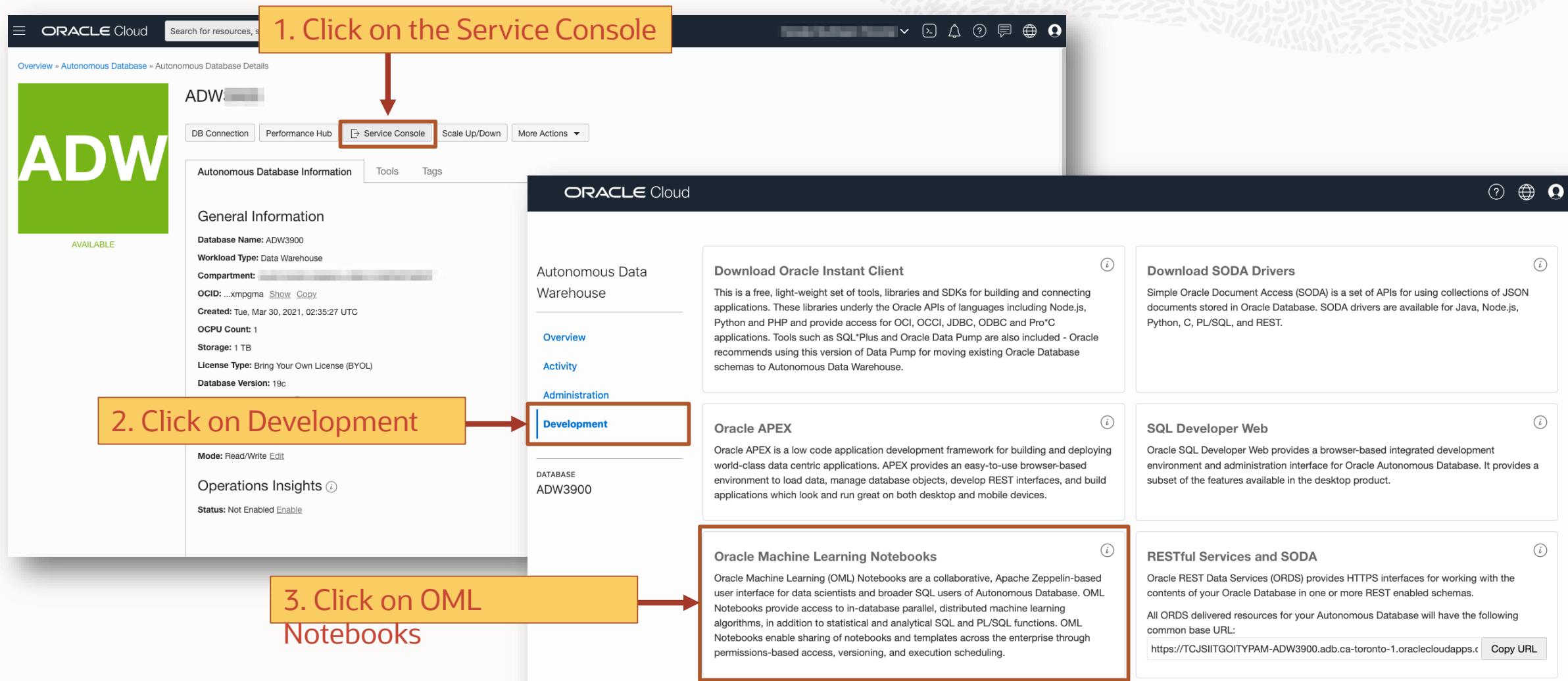
**List Scope**

**Compartment**

1. Click and select the Compartment associated with your Account

2. Click on the Autonomous Database Name assigned to you

**Once inside the Autonomous Database, click on "Service Console".  
In there, select "Development" and then click on "Oracle Machine Learning Notebooks"**



In the Sign-In screen of OML Notebooks, use **OMLUSER** and the Password **AAbbcc123456**. Once you are in, click on the OML Notebooks link

Sign in with  
**OMLUSER**  
and the Password  
**AAbbcc123456**

ORACLE Cloud Infrastructure

SIGN IN

Database name:  
ADW3900

Sign in with your Oracle Machine Learning Database User credentials

USERNAME  
**OMLUSER**

PASSWORD  
.....

Sign In

ORACLE Machine Learning

OMLUSER Project [OMLUSER Works...]

OMLUSER

How Do I?

- Use AutoML
- Get Started
- Create Notebooks
- Create Jobs
- Manage Permissions
- Try It

Recent Notebooks

Nothing to Display

Quick Actions

- AutoML
- Scratchpad
- Notebooks**
- Jobs
- Examples

Recent Activities

No items to display.

Click on OML Notebooks

# Back in the LiveLabs Main page (where all links and passwords are), scroll down to the bottom of the page and click "Open the workshop instructions in a new tab"

The screenshot shows the Oracle LiveLabs main page. At the top, there's a navigation bar with 'LiveLabs' and a search bar. Below it, a banner for the 'Launch Oracle Machine Learning for Python on Autonomous Database Workshop' is displayed. The banner includes a 'Workshop Details' section with various login credentials and a 'Launch Console' button. A red box highlights the 'Let's Get Started - Log in to Oracle Cloud' link. Another red box highlights the 'Click here to open the next part of the workshop' link. A third red box highlights the 'Open the workshop instructions in a new tab' link at the bottom of the page.

**New TAB with the Instructions**

The right side of the screenshot shows a new tab with the title 'Get Started with OML4Py on Autonomous Database'. The page contains an introduction, objectives, and a list of steps. The 'Introduction' section says: 'This lab walks you through the steps to create an OML notebook and connect to the Python interpreter.' The 'Objectives' section lists: 'Create an OML user', 'Access OML Notebooks', 'Create an OML Notebook', 'Familiarize with the OML Notebook toolbar', 'Familiarize with the OML Notebook interpreters', 'Connect to the Python interpreter', 'Verify Connection to the Autonomous Database', and 'View help files'. The 'Step 1: Create an OML User' section is expanded, showing sub-steps like 'Step 1: Create an OML User', 'Step 2: Access Oracle Machine Learning Notebooks', 'Step 3: (Optional) Download and View the Notebook File', etc. Other sections like 'Step 2', 'Step 3', 'Step 4', 'Step 5', 'Step 6', 'Step 7', 'Learn More', and 'Acknowledgements' are also visible.

# Let's import the Notebooks for the Labs



Option 1: Download All Labs Notebooks in a single ZIP file from:

<http://oracle.com/a/tech/docs/otn-batch1/oml4py-hol-notebooks.zip>

Option 2: At each Lab, under the Optional Section, you can download a copy of each of the Notebooks to import them into your own. Another way is to copy and paste each paragraph on a new Notebook

The screenshot shows the Oracle Machine Learning for Python on Autonomous Database documentation. The main content area is titled "Get Started with OML4Py on Autonomous Database". It includes sections for "Introduction", "Objectives", and a list of "What You'll Learn". Below this, there are four steps: Step 1: Create an OML User, Step 2: Access Oracle Machine Learning Notebooks, Step 3: (Optional) Download and View the Notebook File, and Step 4: Create an OML Notebook. The "Step 3: (Optional) Download and View the Notebook File" section is highlighted with a red box and contains the following text:

To download the notebook version of this lab (without screenshots), click [here](#).

To view the downloaded notebook file:

1. In the Notebooks page, click **Import** and upload the notebook file.
2. After the notebook is successfully imported, click the notebook to view it.

**Note:** You can now follow the instructions in the notebook for performing the steps below.

A callout box with a red border and arrow points from the "here" link in the Step 3 section of the left screenshot to the "here" link in the "Step 3: (Optional) Download and View the Notebook File" section of this screenshot. The text in the callout box is:

To download the notebook version of this lab (without screenshots), click [here](#).

# Import the Notebooks into your OML Notebooks Session

It should say 6 out of 6 notebooks imported successfully

The screenshot shows the Oracle Machine Learning interface for managing notebooks. A red box highlights the 'Import' button in the top navigation bar. A red arrow points from a file selection dialog on the left to the 'Import' button. Another red arrow points from the 'Open' button in the dialog to the 'Import' button. A yellow callout box contains the text: 'You can import multiple by selecting the 6 Notebook JSON files on your machine'. The main interface displays a list of imported notebooks with the message '6 out of 6 notebooks imported successfully'.

**Notebooks**

Name Comment Last Update Updated By Connection Group

No data to display.

Page 1 (0 of 0 items) K < 1 > K

**Import**

OMLUSER Project [OMLUSER Works...]

OMLUSER

Search...

6 out of 6 notebooks imported successfully

**Notebooks**

Name Com... Last Update Updated By Connection Group

Lab 1: Get Started with OML4Py on Autonomous Database (LiveLabs)_1	3/30/21 3:41 PM	OMLUSER	Global
Lab 5: Run user-defined functions using Embedded Python Execution	3/30/21 3:41 PM	OMLUSER	Global
Lab 2: Select and manipulate data using the Transparency Layer	3/30/21 3:41 PM	OMLUSER	Global
Lab 3: Use in-database algorithms and models	3/30/21 3:41 PM	OMLUSER	Global
Lab 4: Use Datastores to store Python objects	3/30/21 3:41 PM	OMLUSER	Global
Lab 6: Use AutoML	3/30/21 3:41 PM	OMLUSER	Global

Search...

OML Project [OML Workspace]

OMLUSER

Cancel Open

INSYNC21

6 items 6 documents - 1.1 MB

Information

Created Mar 26 - 30, 2021 Modified Mar 26 - 30, 2021

lab1\_get\_st\_livelabs.json  
lab2\_select\_te\_data.json  
lab3\_in-db.algo.json  
lab4\_datastores.json  
lab5\_embed\_python.json  
lab6\_automl.json

You can import multiple by selecting the 6 Notebook JSON files on your machine

Open

*Work on lab 2  
10 Minutes...*

# Lab 3: Using in-database algorithms and models



# OML4Py 1.0

Machine Learning in-database algorithms



## Classification

- Decision Tree
- Naïve Bayes
- Generalized Linear Model
- Support Vector Machine
- Random Forest
- Neural Network

## Regression

- Generalized Linear Model
- Neural Network
- Support Vector Machine

## Clustering

- Expectation Maximization
- Hierarchical k-Means

## Attribute Importance

- Minimum Description Length

## Anomaly Detection

- 1 Class Support Vector Machine

## Association Rules

- Apriori – Association Rules

## Feature Extraction

- Singular Value Decomposition
- Explicit Semantic Analysis
- Principal Component Analysis via SVD

*Supports automatic data preparation, partitioned model ensembles, integrated text mining*

# Scalable in-database algorithms

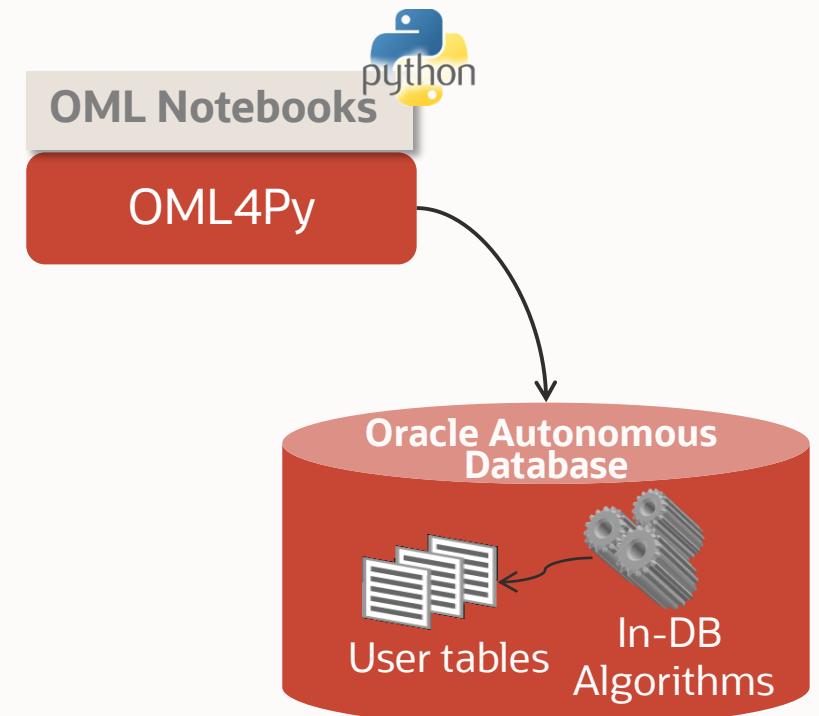
Example using Support Vector Machine for anomaly detection

```
from oml import svm

# create proxy object
ONTIME = oml.sync(table='ONTIME')

# define model object
settings = {'svms_outlier_rate' : 0.01}
svm_mod = svm('anomaly_detection',
              svms_kernel_function =
                  'dbms_data_mining.svms_linear',
              **settings)

# build anomaly detection model
svm_mod = svm_mod.fit(x=ONTIME, y=None)
# view model object
svm_mod
```



# Partitioned Models

Automating a typical machine learning task

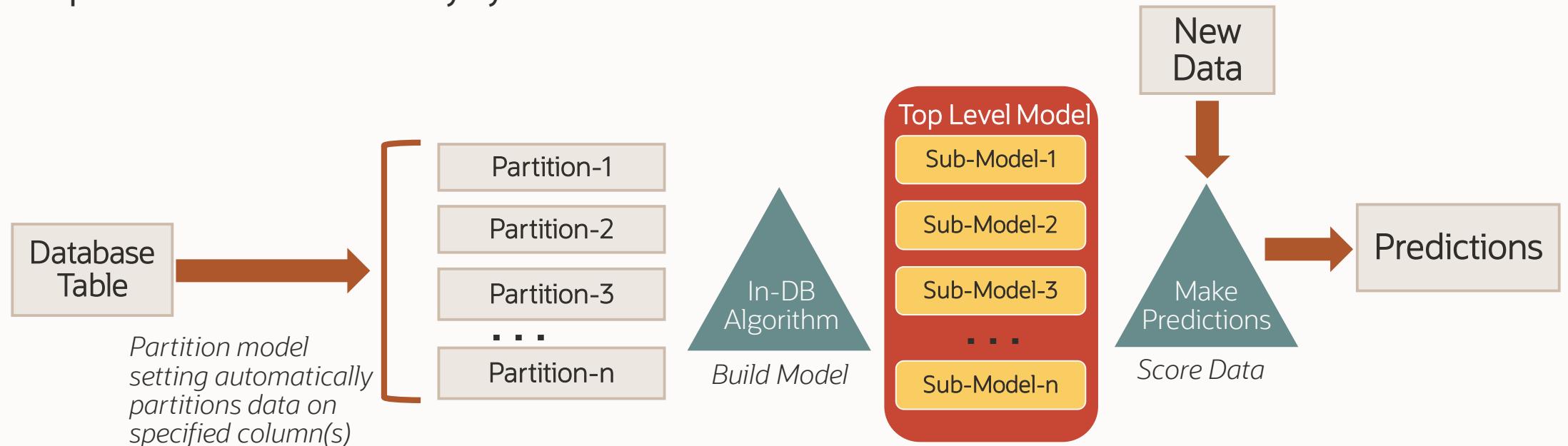


Builds ensemble model with multiple sub-models, one for each data partition

- Potentially achieve better accuracy through multiple targeted models
- Sub-models automatically managed and used as one model

Simplified scoring using top-level model only

- Proper sub-model chosen by system based on row of data to be scored



# Lab 3: Using in-database algorithms and models

**Goal:** Learn how to build and score using in-database machine learning algorithms, as well as some of the other model-related functionality

Step 1: Import libraries

Step 2: Regression using GLM

Step 3: Clustering using KMeans

Step 4: Partitioned Models

Step 5: Rank attribute importance using Model Explainability



*Work on lab 3  
10 Minutes...*

# Lab 4: Use datastores to store Python objects



# Datastore for Python object persistence

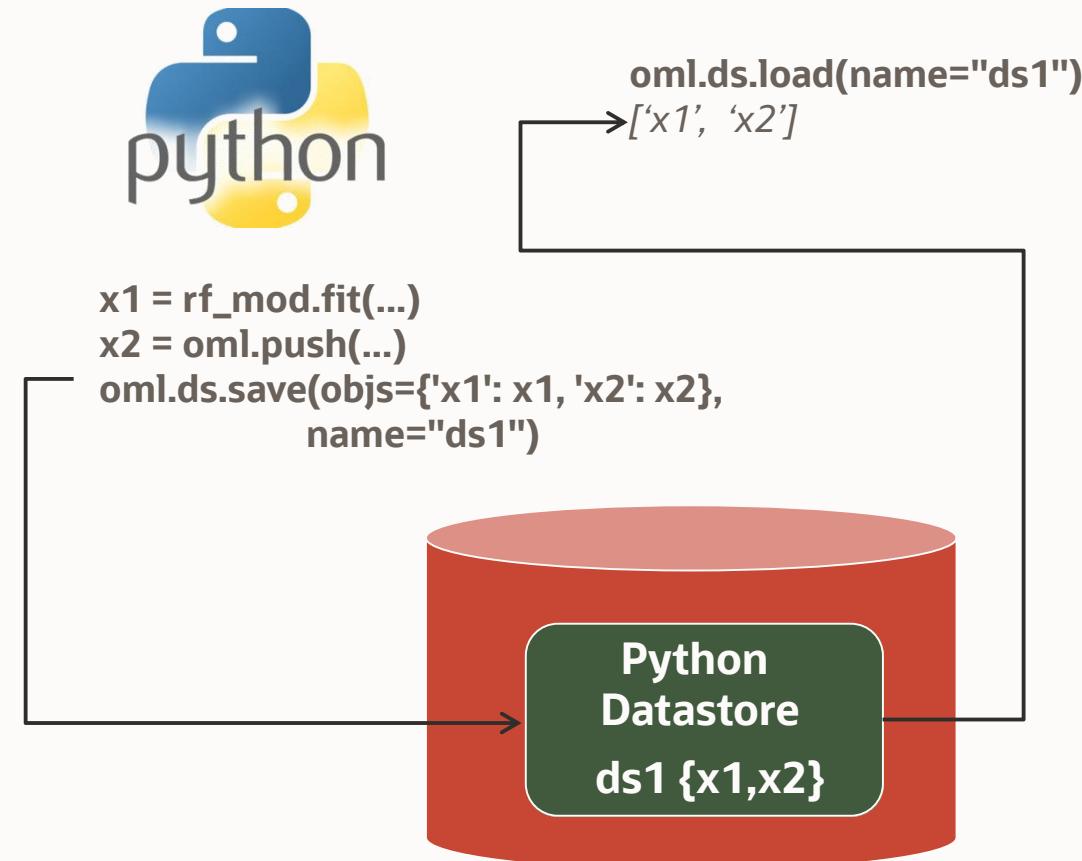


`oml.ds.save()` and `oml.ds.load()`

Provide database storage to save/restore  
Python and OML4Py objects across  
Python sessions

## Use cases

- Preserve OML4Py objects across Python sessions
- Passing arguments to Python functions with embedded Python execution, especially when non-scalar for REST invocation, such as native Python ML models



# Datastore functions for storing Python objects



**oml.ds.save**(objs, name[, description, . . . ])

- Saves Python objects to a datastore in the user's schema

**oml.ds.dir**([name, regex\_match, dstype])

- Lists existing datastores available to the current session user

**oml.ds.describe**(name[, owner])

- Describes the contents of the named datastore available to the current session user

**oml.ds.load**(name[, objs, owner, to\_globals])

- Loads Python objects from a datastore in the user's schema

**oml.ds.delete**(name[, objs, regex\_match])

- Deletes one or more datastores from the user's schema
- Deletes specific objects within a named datastore

**oml.grant**(name[, typ, user])

- Grants read privilege for a Python datastore

**oml.revoke**(name[, typ, user])

- Revokes read privilege for a Python datastore

# Lab 4: Use datastores to store Python objects

**Goal:** Learn how to store and manage Python objects, both native and from OML, in the database using Datastore

Step 1: Import libraries supporting OML4Py

Step 2: Create Pandas DataFrames and load them into Autonomous Database

Step 3: Save Python objects to datastore

Step 4: Save model objects in a datastore

Step 5: Load datastore objects into memory

Step 6: View datastores and other details

Step 7: View contents of a datastore

Step 8: Manage datastore privileges

Step 9: Delete datastore content



# Lab 4: Create OMLUSER2 for managing privileges to OML4Py Data Store



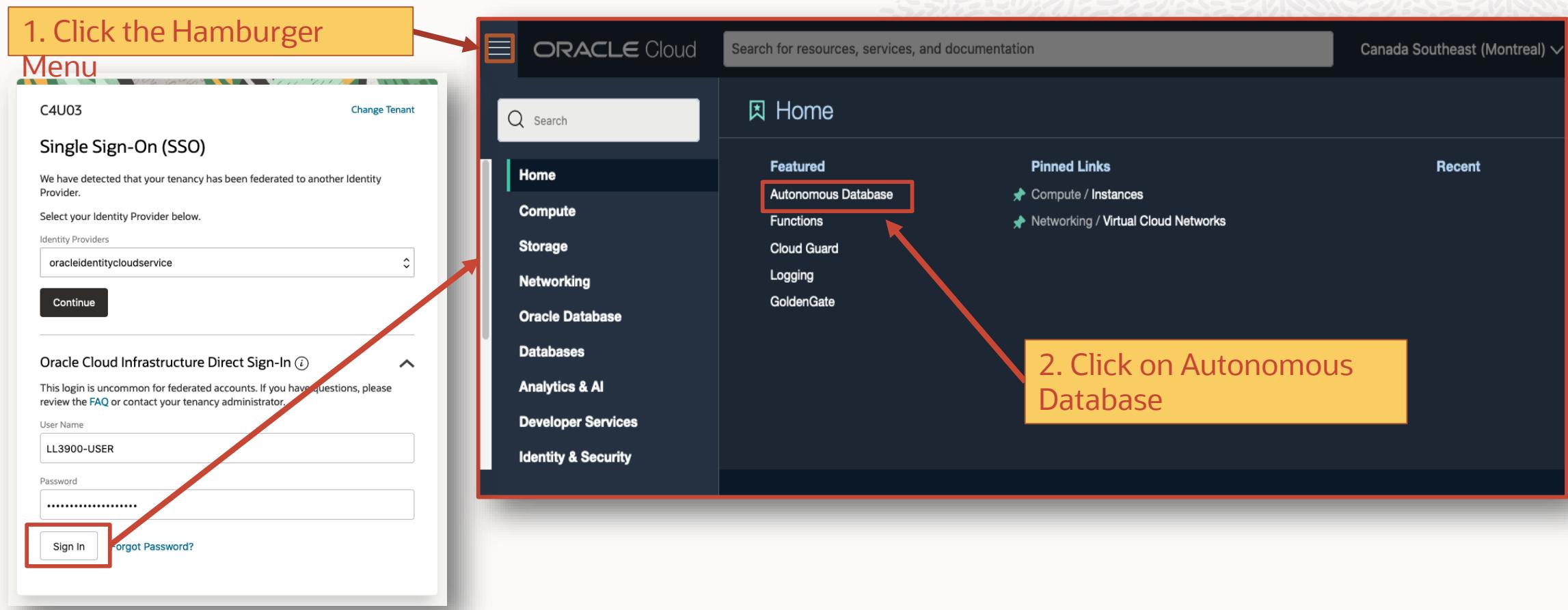
# If you don't have the Autonomous Database already open, we need to get there:

In the "My Reservations" window at: <https://apexapps.oracle.com/pls/apex/dbpm/r/livelabs/my-reservations>  
Click on "Launch Workshop", then "Launch Console"

The screenshot shows the Oracle LiveLabs interface. At the top, there's a navigation bar with 'LiveLabs', a search bar, 'Feedback', and user profile information. Below it, a sub-navigation bar includes 'Home', 'Available Workshops', 'My Reservations', 'Have an Event Code?', and 'FAQ'. The main content area features a green header for a workshop titled 'Launch Oracle Machine Learning for Python on Autonomous Database Workshop'. The header includes a search bar, a 'Feedback' link, and a 'Rate this workshop?' button. Below the header, a section titled 'Workshop Details' (with a plus sign icon) lists various parameters: Login URL, Tenancy name, Region, User name, Initial password, Compartment, Compartment OCID, Autonomous DB Admin Password, and Database Name. Each parameter has a 'Copy value' button next to it. A large red box highlights the 'Launch Console' button, which is located to the right of the 'Autonomous DB Admin Password' row. At the bottom of the main content area, there are 'Reset Workshop Password' and 'Extend Workshop Reservation' buttons. The background of the main content area has a green circular pattern.

This screenshot shows the 'My Reservations' page from the Oracle LiveLabs interface. It features a header with 'LiveLabs', a search bar, and a navigation bar with 'Home', 'Available Workshops', and 'My Reservations'. The main content area is titled 'My Reservations' and contains a note stating 'All your current workshop reservations are listed below.' Below this, there's a note about email notifications for status updates. At the bottom, there's a section for a specific reservation: 'Oracle Machine Learning for Python on Autonomous Database' (status: Available), with a 'Launch Workshop' button highlighted by a red arrow. Other buttons in this section include 'Workshop Details' and a trash can icon. The background of the main content area has a light gray grid pattern.

In the Single Sign-On screen, enter the Password given in the Workshop screen for your user **LL#####-USER** (or whatever password you changed it to, if you did change it). Click on the Hamburger Menu on the Top left of OCI, then Autonomous Database.



# Select the specific ADW Compartment indicated in the Workshop Main screen, and then click on the Autonomous Database

The screenshot shows the Oracle Cloud Autonomous Database interface. On the left, a sidebar lists options like Autonomous Database, Dedicated Infrastructure, Autonomous Container Database, and Autonomous Exadata Infrastructure. Below this is a 'List Scope' section with a 'Compartment' dropdown. The 'Search compartments' field has 'c4u03 (root)' selected, and under it, 'Livelabs' is expanded, showing compartments such as LL3298-COMPARTMENT, LL3872-COMPARTMENT, LL3892-COMPARTMENT, LL3899-COMPARTMENT, LL3900-COMPARTMENT, LL6082-COMPARTMENT, and LL6127-COMPARTMENT. The 'LL3900-COMPARTMENT' item is highlighted with a red box and an arrow pointing to it from the first callout. To the right of the sidebar is a main content area titled 'Autonomous Databases in LL3900-COMPARTMENT'. It features a 'Create Autonomous Database' button and a table with columns: Display Name, State, Dedicated, OCPUs, and Storage (TB). One row in the table is highlighted with a red box and an arrow pointing to it from the second callout. This row contains the 'ADW3900' display name, which is also highlighted with a blue box and an arrow pointing to it from the third callout.

Autonomous Databases in LL3900-COMPARTMENT

Display Name	State	Dedicated	OCPUs	Storage (TB)
ADW3900	Available	No	1	1

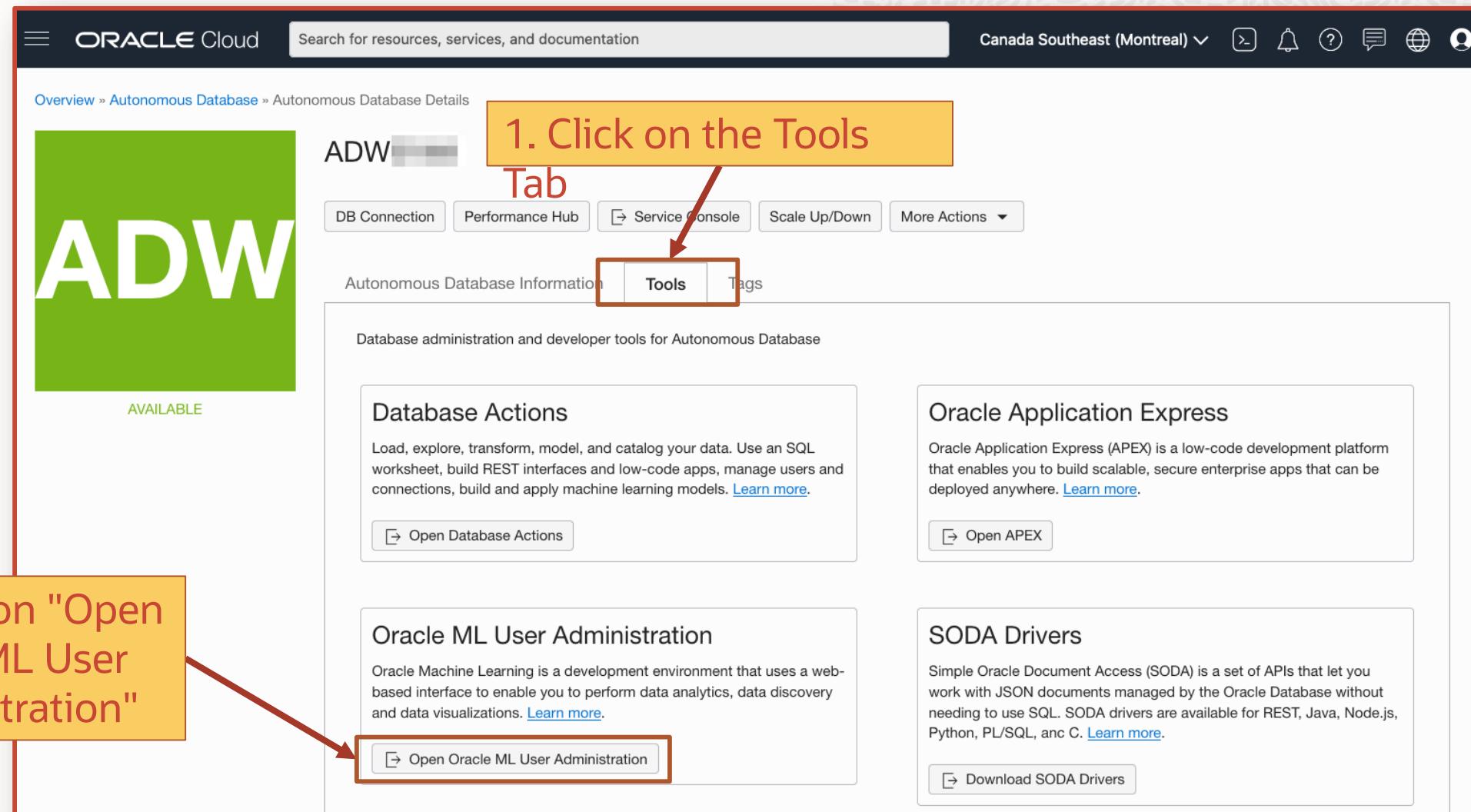
Autonomous Databases in LL3900-COMPARTMENT

Display Name	State	Dedicated	OCPUs	Storage (TB)
ADW3900	Available	No	1	1

1. Click and select the Compartment associated with your Account

2. Click on the Autonomous Database Name assigned to you

Once inside the Autonomous Database, click on the "Tools" tab.  
In there, click on the "Open Oracle ML User Administration" under the Oracle ML Tile



You will be asked to Login, this time as the Administrator of the Oracle Database  
Make sure to use the user **ADMIN** and the **Initial Password** indicated in the Console

The image shows two side-by-side screenshots. On the left, the 'Launch Oracle Machine Learning for Python Workshop' page from Oracle LiveLabs displays various connection details. A red box highlights the 'Initial password' field, which is obscured by a redacted string. To its right is a 'Copy Password' button. A large orange callout box with the text 'Copy the INITIAL PASSWORD from here' has arrows pointing from it to both the redacted password field and the 'Copy Password' button. On the right, the 'ORACLE Cloud Infrastructure' sign-in page shows the 'Database name:' field filled with 'ADW5144'. Below it, the 'USERNAME' field contains 'ADMIN' and the 'PASSWORD' field contains a redacted string. A 'Sign In' button is at the bottom.

LiveLabs

Search Workshops...

Home Available Workshops My Reservations Have an Event Code? FAQ

## Launch Oracle Machine Learning for Python Workshop

Python is one of the favorite languages used in AI and ML applications. Learn how the Oracle Database highlighting the broad set of algorithms and high performance execution.

Time Remaining: 3h 9m 5s

**Workshop Details** (click + to view login details for the workshop)

Login URL	: https://cloud.oracle.com/?region=ca-toronto-1&telemetry=false&t
Tenancy name	: C4U03
Region	: Canada Southeast (Toronto)
User name	: LL3900-USER
Initial password	: [REDACTED] <input type="button" value="Copy Password"/>
Compartment	: LL3900-COMPARTMENT
Compartment OCID	: ocid1.compartment.oc1.aaaaaaaa4g7x6rbvqne3
Autonomous DB Admin Password	: [REDACTED] <input type="button" value="Copy value"/>
Database Name	: ADW3900 <input type="button" value="Copy value"/>

**ORACLE® Cloud Infrastructure**

SIGN IN

Database name:  
**ADW5144**

Sign in with your Oracle Machine Learning Database Administrator credentials

USERNAME  
**ADMIN**

PASSWORD  
.....

Sign In

Copy the INITIAL PASSWORD from here

You will see the users ADMIN and OMLUSER, but we want to create a new one named OMLUSER2. We can give it just a random e-mail, and use the same Initial Password

The screenshot shows two windows of the Oracle Machine Learning User Administration application.

**Left Window (Users List):**

- Header: ORACLE® Machine Learning User Administration
- Section: Users
- Buttons: + Create, Delete, Show All Users
- Data:

User Name	Full Name	Role
ADMIN		System Administrator
OMLUSER		Developer

A red box highlights the "+ Create" button with the instruction "Click on the "+Create". A yellow box contains the following instructions:

- Fill it in with:
- OMLUSER2
- Any e-mail address
- Uncheck Generate Pass
- Paste the Initial Password twice

**Right Window (Create User):**

- Header: ORACLE® Machine Learning User Administration
- Title: Create User
- Buttons: Create, Cancel
- Form fields:

* Username	OMLUSER2
First Name	[empty]
Last Name	[empty]
* Email Address	any.email@anyserver.com
<input type="checkbox"/> Generate password and email account details to user. User will be required to reset the password on first sign in.	
* Password	[redacted]
* Confirm Password	[redacted]

Dotted arrows point from the yellow box on the left to the corresponding fields in the right window: "OMLUSER2" to "Username", "any.email@anyserver.com" to "Email Address", the checkbox to "Generate password and email account details to user.", and the password fields to "Password" and "Confirm Password".

*Work on lab 4*  
*10 Minutes...*

# Lab 5: Run user-defined functions using Embedded Python Execution



# Embedded Python Execution – features on Autonomous Database

- Database environment controls and manages spawning of Python engines
- Return results as Oracle tables and views, accessible via DataFrame proxy objects
- Return image and structured content from user-defined functions
- Use provided third-party packages in user-defined functions
- Run user-defined Python functions using data- and task-parallelism
- Store and manage user-defined Python functions in database script repository
- Invoke user-defined Python functions using REST for application integration



# Embedded Python execution functions for invoking user-defined Python functions



## **oml.do\_eval(func[, func\_owner, graphics])**

Runs the user-defined Python function using a Python engine spawned and controlled by the database environment

## **oml.table\_apply(data, func[, func\_owner, ... ])**

Runs the user-defined Python function with data pulled from a database table or view using a Python engine spawned and controlled by the database environment

## **oml.row\_apply(data, func[, func\_owner, ... ])**

Partitions database data into chunks of rows and runs the user-defined Python function on each chunk using Python engines spawned and controlled by the database environment

## **oml.group\_apply(data, index, func[, ... ])**

Partitions database data by the column(s) specified in index and runs the user-defined Python function on each partition using Python engines spawned and controlled by the database environment

## **oml.index\_apply(times, func[, func\_owner, ... ])**

Runs the user-defined Python function multiple times, passing the run index as first argument, using Python engines spawned and controlled by the database environment

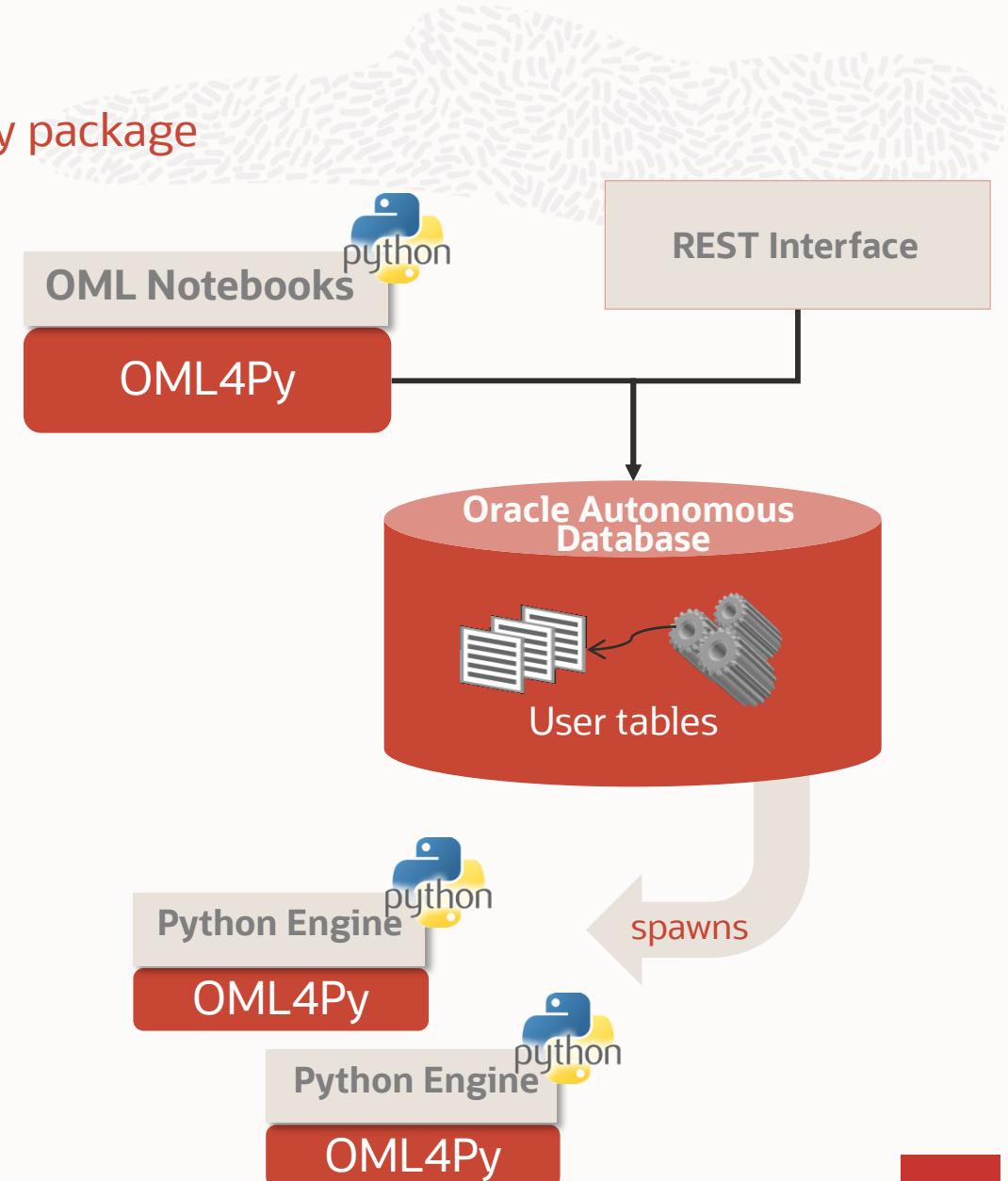
# Embedded Python Execution

Example of parallel partitioned data flow using third party package

```
# user-defined function using sklearn
def build_lm(dat):
    from sklearn import linear_model
    lm = linear_model.LinearRegression()
    X = dat[['PETAL_WIDTH']]
    y = dat[['PETAL_LENGTH']]
    lm.fit(X, y)
    return lm

# select column(s) for partitioning data
index = oml.DataFrame(iris['SPECIES'])
# invoke function in parallel on IRIS table
mods = oml.groupby_parallel(iris, index,
                            func=build_lm,
                            parallel=2)

mods.pull().items()
```



# OML4Py script repository



Use the script repository to...

- Create and store user-defined Python functions as scripts in Oracle Database
- Grant or revoke the read privilege to a script
- List available scripts
- Load a script function into the Python environment
- Drop a script from the script repository

# Script repository functions for managing Python scripts



## **oml.script.create**(name, func[, is\_global, ... ])

- Creates a Python script, which contains a single function definition, in the Oracle Database Python script repository

## **oml.script.dir**([name, regex\_match, sctype])

- Lists the scripts present in the Oracle Database Python script repository

## **oml.script.load**(name[, owner])

- Loads the named script from the Oracle Database Python script repository as a callable object

## **oml.script.drop**(name[, is\_global, silent])

- Drops the named script from the Oracle Database Python script repository

## **oml.grant**(name[, typ, user])

- Grants read privilege for a Python script (or datastore)

## **oml.revoke**(name[, typ, user])

- Revokes read privilege for a Python script (or datastore)

# Lab 5: Run user-defined functions using Embedded Python Execution

**Goal:** Learn how to run user-defined Python functions in database-controlled Python engines and work with the Python script repository

Step 1: Import the OML4Py library

Step 2: Build and score a Linear Model

Step 3: Build the model using embedded Python execution

Step 4: Build one model per Species using the group\_apply function

Step 5: Invoke a function N times

Step 6: Return multiple images from embedded Python execution

Step 7: Using the Python script repository

Step 8: Create scripts in repository

Step 9: Store a function as a global function and invoke with table\_apply

Step 10: Drop scripts from repository

*Work on lab 5*  
*10 Minutes...*

# Lab 6: Use AutoML



# OML4Py AutoML objectives

Alleviating pain points

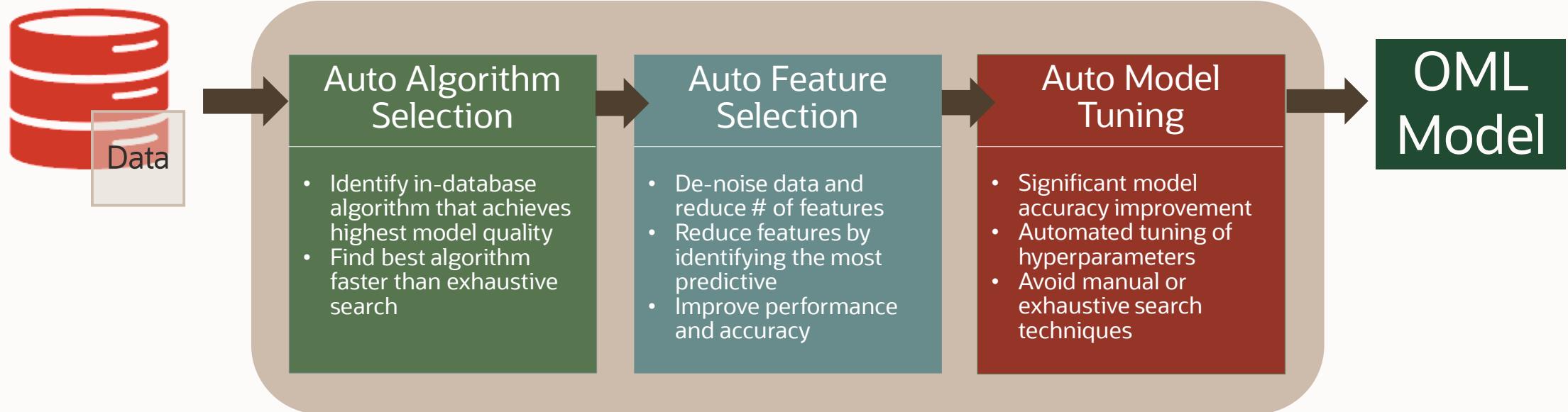
- Eliminate repetitive tasks of model building / evaluation to increase user productivity
- **Apply ML to the ML process** to reduce algorithm and hyperparameters search space and reduce compute time and cost
- Enable non-expert users to leverage machine learning





# AutoML – *new* with OML4Py

Increase data scientist productivity – reduce overall compute time



Enables non-expert users to leverage Machine Learning

# Lab 6: Use AutoML

**Goal:** Become familiar with the AutoML workflow and related functions

Step 1: Import libraries supporting OML4Py

Step 2: Automated Algorithm Selection

Step 3: Automated Feature Selection

Step 4: Automated Model Tuning

Step 5: Automated Model Selection

*Note: Some AutoML function invocations can take a few minutes to complete.  
A lot of going on behind the scenes. Please be patient. ☺*

*Work on lab 6*  
*10 Minutes...*

# Lab 6: Bonus Round - Use OML AutoML UI



If you are not yet at the OML Notebooks screen, Sign-In again with **OMLUSER** and the Password **AAbbcc123456**. Once you are in, click on the **AutoML** link

The screenshot shows the Oracle Cloud Infrastructure sign-in page and the Oracle Machine Learning dashboard.

**Sign-in Page:** The sign-in page has a blue header with the ORACLE Cloud Infrastructure logo and a cloud icon. It displays the database name "ADW3900". The sign-in form asks for "USERNAME" (OMLUSER) and "PASSWORD" (AAbbcc123456), with a "Sign In" button. A yellow callout box with red arrows points to the "USERNAME" field, the "PASSWORD" field, and the "Sign In" button. The text inside the callout box reads: "Sign in with OMLUSER and the Password AAbbcc123456".

**Dashboard:** After signing in, the user is directed to the Oracle Machine Learning dashboard. The top navigation bar shows the project name "OMLUSER Project [OMLUSER Works...]" and the user "OMLUSER". The dashboard features several sections:

- How Do I?** A row of six informational cards:
  - Use AutoML**: How to create AutoML Experiments.
  - Get Started**: Get started with Oracle Machine Learning.
  - Create Notebooks**: How to create a notebook.
  - Create Jobs**: How to create a job.
  - Manage Permissions**: How to manage collaborative permissions in workspaces.
  - Try It**: Follow along with a hands on workshop.
- Quick Actions**: A row of four action cards:
  - AutoML**: Create and run AutoML Experiments. This card is highlighted with a red border and a red arrow points from the "AutoML" callout on the sign-in page to this card.
  - Scratchpad**: Run Scratchpad.
  - Notebooks**: The place for data discovery and analytics.
  - Jobs**: Schedule notebooks to run at certain times.
- Recent Activities**: A section stating "No items to display."

A yellow callout box with a red arrow points to the "AutoML" card in the "Quick Actions" section, with the text "Click on AutoML" inside.

# In the AutoML Experiments screen, click "Create" to create a new one

The screenshot shows the Oracle Machine Learning interface. In the top left, it says "ORACLE Machine Learning". In the top right, there's a dropdown for "OML Project [OML Workspace]" and a user icon for "OMLUSER". Below this, the title "AutoML Experiments" is displayed. A toolbar below the title has several buttons: "+ Create" (highlighted with a red box and arrow), "Edit", "Delete", "Duplicate", "Start", and "Stop". To the right of the toolbar is a search bar with a magnifying glass icon. The main area shows a table with columns "Name", "Comment", "Created On", and "Created By". A message "No data to display." is shown. At the bottom, there's a page navigation bar with "Page 1 (0 of 0 items)" and some navigation icons. A large yellow callout box with red text appears over the "+ Create" button, containing three steps: 1. Click on +Create, 2. Give the Experiment a name, and 3. Click on the Loupe to search for a Data Source. To the right of this callout, a modal window titled "Create Experiment" is open. It has fields for "Name \*" (containing "First AutoML Experiment"), "Comments" (an empty text area), "Data Source \*" (a dropdown menu with a search icon highlighted with a red box and arrow), and "Prediction Type \*" (a dropdown menu). The "Data Source" field has a placeholder "Search..." and a small search icon.

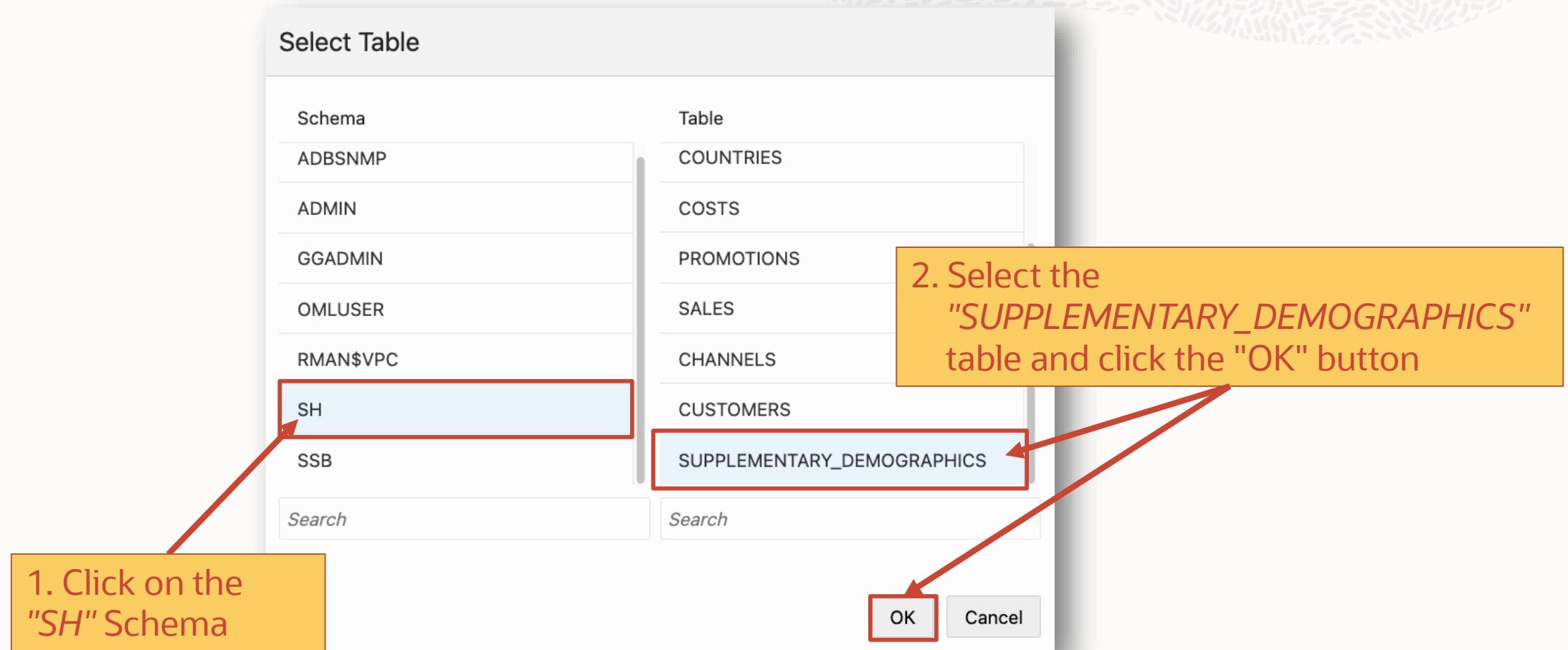
1. Click on +Create

2. Give the Experiment a name

3. Click on the Loupe to search for a Data Source

In the menu that appears, select:

Schema = *SH* & Table = *SUPPLEMENTARY\_DEMOGRAPHICS*



# In the Create Experiments screen, enter the *Predict*, *Case ID*, and adjust performance settings

The screenshot shows the 'Create Experiment' dialog box. Key fields include:

- Name \***: First AutoML Experiment
- Data Source \***: SH.SUPPLEMENTARY\_DEMOGRAPHICS
- Prediction Type \***: Classification
- Predict \***: AFFINITY\_CARD
- Case ID**: CUST\_ID
- Additional Settings**:
  - Maximum Top Models \*: 2
  - Maximum Run Duration (Hours) \*: 8
  - Database Service Level \*: Medium
  - Model Metric \*: Balanced Accuracy

Red arrows point from the numbered callouts to the corresponding fields in the dialog.

1. For the "Predict" attribute, select AFFINITY\_CARD from the pull-down menu
2. For the "Case ID" attribute, select CUST\_ID from the pull-down menu
3. Expand the "Additional Settings" and reduce the Maximum Top Models to "2" to save time in this exercise.
4. Change the "Database Service Level" to Medium to get increased parallelism

# We are ready to start the Experiment

Click on the Start -> Faster Results

The screenshot shows the Oracle Machine Learning interface for creating an experiment. The top navigation bar includes the Oracle logo, the project name "OMLUSER Project [OMLUSER Works...]", and the user "OMLUSER". The main title is "Create Experiment". The form fields include "Name \*" (First AutoML Experiment), "Comments", "Data Source \*" (SH.SUPPLEMENTARY\_DEMOGRAPHICS), "Prediction Type \*" (Classification), and "Additional Settings" with options for "Maximum Top Models" (set to 2), "Maximum Run Duration (Hours)" (set to 8), and "Database Service Level" (set to Medium). A modal window titled "Start" is open, showing three options: "Start", "Save", and "Cancel". The "Start" button is highlighted with a red box and an arrow labeled "1. Click on Start". A dropdown menu from the "Start" button contains "Faster Results" and "Better Accuracy", with "Faster Results" also highlighted with a red box and an arrow labeled "2. Click on Faster Results to make sure we have a quick result for this Session". In the background, a progress bar indicates the experiment is "Starting Experiment". Three yellow callout boxes with arrows point to the "Start" button, the "Faster Results" option, and the progress bar.

1. Click on Start

2. Click on Faster Results to make sure we have a quick result for this Session

3. Experiment will start

# Click on the three dots (...) to see the Progress

The screenshot shows the Oracle Machine Learning interface. At the top, there's a navigation bar with the Oracle logo, the text "Machine Learning", and a dropdown for "OMLUSER Project [OMLUSER Works...]" and "OMLUSER". Below the navigation, the main content area is titled "First AutoML Experiment" and displays a chart titled "Balanced Accuracy" with a single data point at 0.500. Below the chart is a "Leader Board" table.

A yellow callout box with red text "Click on the three dots (...) to open the Progress Report" points to a red-bordered "Progress" modal window. The "Progress" window lists several tasks:

- Algorithm Selection: Completed
- Adaptive Sampling: Completed
- Feature Selection: Running
- Model Tuning: Queued
- Naive Bayes: Queued
- Generalized Linear Model: Queued
- Feature Prediction Impact: Queued

At the top right of the "Progress" window, there's a "Stop" button next to a "Running" status indicator.

# Completed vs Running Steps

The screenshot shows the Oracle Machine Learning interface. At the top, there's a navigation bar with the Oracle logo, the project name "OMLUSER Project [OMLUSER Works...]", and a user profile "OMLUSER". Below the navigation bar, the main content area is titled "First AutoML Experiment". It includes sections for "Experiment Settings" (with an "Edit" button), "Balanced Accuracy" (showing a value of 0.500), and a "Leader Board" table.

A modal window titled "Progress" is open, listing the steps of the experiment:

Step	Status
Algorithm Selection	Completed
Adaptive Sampling	Completed
Feature Selection	Running
Model Tuning	Queued
Naive Bayes	Queued
Generalized Linear Model	Queued
Feature Prediction Impact	Queued

A red box highlights the "Completed" status for the first two steps, and a red arrow points from a text callout to the "Running" status for the Feature Selection step.

Checkboxes are shown for completed steps, while running steps show an animation

# Top algorithms selected by AutoML

The screenshot shows the Oracle Machine Learning interface. At the top, it says "OMLUSER Project [OMLUSER Works...]" and "OMLUSER". Below that, it says "First AutoML Experiment". On the left, there's a "Balanced Accuracy" chart and a "Leader Board" table. A red box highlights the "Algorithm" column in the Leader Board table, and a red arrow points from this box to a yellow callout box containing the text: "Only the Top algorithms identified in the \"Algorithm Selection\" phase are shown in the Leader Board". To the right of the Leader Board is a "Progress" dialog box. A red arrow points from the yellow callout box towards the "Progress" dialog. The "Progress" dialog lists several tasks: "Algorithm Selection" (Completed), "Adaptive Sampling" (Completed), "Feature Selection" (Running), "Model Tuning" (Queued), "Naive Bayes" (Queued), "Generalized Linear Model" (Queued), and "Feature Prediction Impact" (Queued). The "Feature Selection" task is currently running.

Only the Top algorithms identified in the "Algorithm Selection" phase are shown in the Leader Board

Algorithm	Model Name	Balanced Accuracy
Naive Bayes	nb_5894c519c9	0.500
Generalized Linear Model	glm_15baeb3f99	0.496

# A completed Experiment looks like this

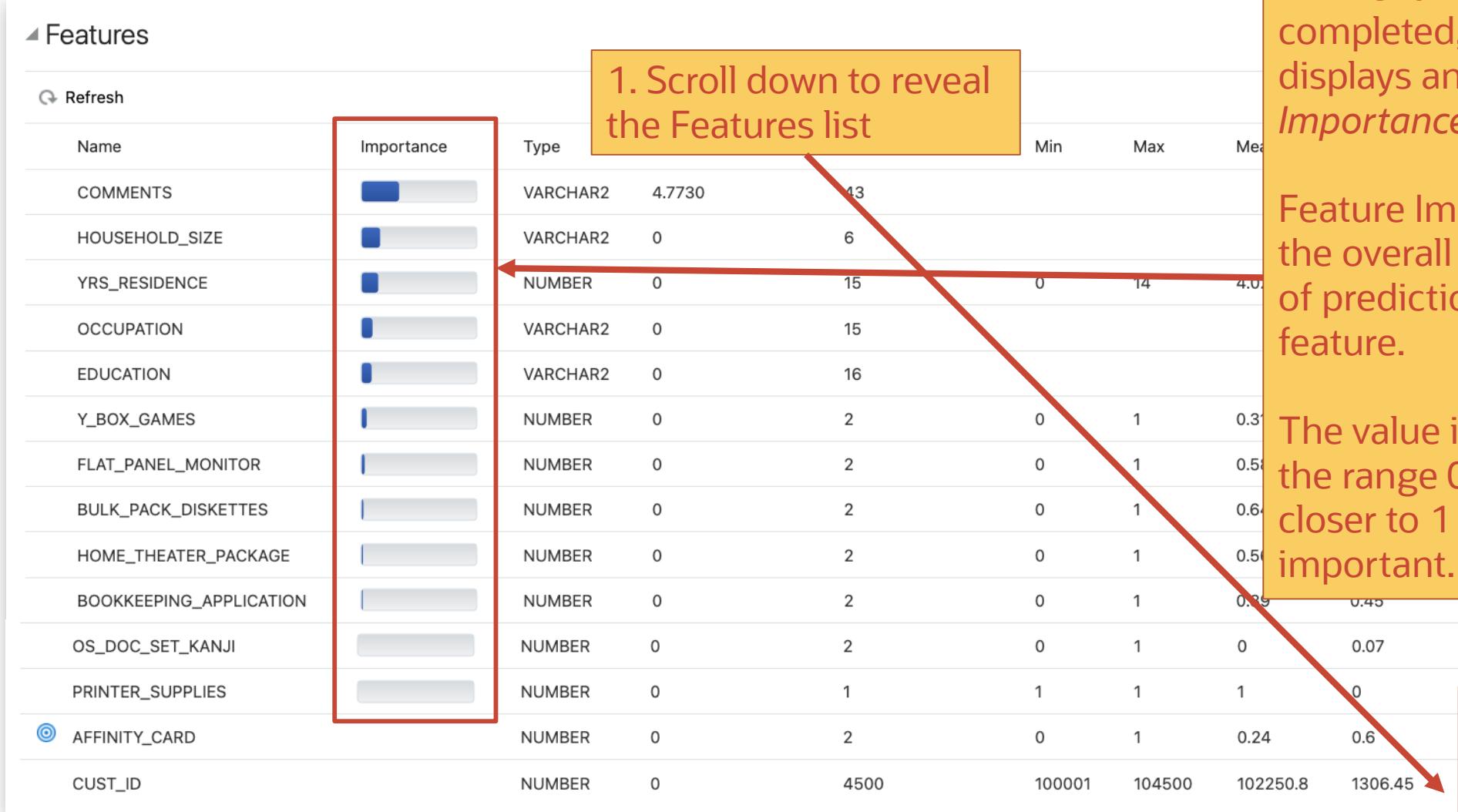
The screenshot shows the Oracle Machine Learning interface with a completed experiment. The top navigation bar includes the Oracle logo, 'Machine Learning', 'OMLUSER Project [OMLUSER Works...]', and 'OMLUSER'. A red callout box on the left says: 'The Experiment finishes when you see the "Completed" message at the top' with an arrow pointing to the 'Completed' status indicator. Another red callout box on the right says: 'Checkboxes are shown for all completed steps' with an arrow pointing to the 'Run Summary' section. The main content area displays the 'First AutoML Experiment' with a 'Balanced Accuracy' chart (ranging from 0.50 to 0.80) showing a sharp rise from ~0.50 to ~0.85. Below the chart is a 'Leader Board' table:

Deploy	Create Notebook	Metrics
Algorithm	Model Name	Balanced Accuracy
Naive Bayes	<a href="#">nb_5894c519c9</a>	0.8868
Generalized Linear Model	<a href="#">glm_15baeb3f99</a>	0.8697

**Run Summary**

Algorithm Selection	Completed
Adaptive Sampling	Completed
Feature Selection	Completed
Model Tuning	Completed
Naive Bayes	Completed
Generalized Linear Model	Completed
Feature Prediction Impact	Completed

# Let's check the global Feature Importance



# Let's open the Model Detail to inspect the Model

The screenshot shows the Oracle Machine Learning interface. At the top, it displays "ORACLE® Machine Learning", the project name "OMLUSER Project [OMLUSER Works...]", and the user "OMLUSER". Below this, the title "First AutoML Experiment" is shown, along with "Experiment Settings" and an "Edit" button. A "Completed" status indicator with a green checkmark and a "Start" button are also present.

A chart titled "Balanced Accuracy" is displayed, showing a curve that rises sharply from approximately 0.55 to 0.85. A yellow callout box with a red border contains the text "Click on the Naïve Bayes model name in blue". A red arrow points from this callout to the "Model Name" column in the "Leader Board" table, specifically highlighting the entry for "Naive Bayes".

The "Leader Board" table has columns for "Algorithm", "Model Name", and "Balanced Accuracy". The data is as follows:

Algorithm	Model Name	Balanced Accuracy
Naive Bayes	<a href="#">nb_5894c519c9</a>	0.8868
Generalized Linear Model	<a href="#">glm_15baeb3f99</a>	0.8697

To the right of the main interface, a modal window titled "Run Summary" is open, listing the completed steps of the experiment:

- Algorithm Selection (Completed)
- Adaptive Sampling (Completed)
- Feature Selection (Completed)
- Model Tuning (Completed)
- Naive Bayes (Completed)
- Generalized Linear Model (Completed)
- Feature Prediction Impact (Completed)

# Review the Model Detail – Prediction Impacts

The screenshot shows the Oracle Machine Learning interface. In the center, a modal window titled "Model Detail - nb\_5894c519c9" is displayed. The window has two tabs: "Prediction Impacts" (which is selected) and "Confusion Matrix". The "Prediction Impacts" tab contains a table with columns "Name" and "Prediction Impact". The table lists several attributes with their corresponding impact values represented by horizontal bars. A red box highlights this table. To the right of the modal, a large orange callout box provides information about the feature:

A new window with the Model Details show the Prediction Impacts of the Attributes.

It uses OML's Machine Learning Explainability module to provide model-agnostic functionality to identify the important features that impact a trained model's predictions.

Name	Prediction Impact
COMMENTS	~0.70
HOUSEHOLD_SIZE	~0.55
OCCUPATION	~0.45
YRS_RESIDENCE	~0.40
Y_BOX_GAMES	~0.40
HOME_THEATER_PACKAGE	~0.40
EDUCATION	~0.35
BOOKKEEPING_APPLICATION	~0.35
BULK_PACK_DISKETTES	~0.35

# Review the Model Detail – Confusion Matrix

The screenshot shows the Oracle Machine Learning interface. At the top, it says "OMLUSER Project [OMLUSER Works...]" and "OMLUSER". Below that, there's a "Completed" status indicator and a "Start" button. On the left, there's a link "[-> Experiments](#)". The main title is "First AutoML Experiment". On the left, there are links for "Experiment Settings" and "Balanced Accuracy". A graph for "Balanced Accuracy" is shown, starting at 0.50 and rising to approximately 0.85. Below the graph is a "Leader Board" table.

**Model Detail - nb\_5894c519c9**

**Prediction Impacts**      **Confusion Matrix**

	Predicted: 0	Predicted: 1
Actual: 0	598	63
Actual: 1	32	212

A red box highlights the "Confusion Matrix" tab, and a red arrow points from this box to a yellow callout box containing the following text:

The Confusion Matrix shows an evaluation of the Model on the Validation Data selected by AutoML at the end of the Process

Algorithm	Model Name	Balanced Accuracy
Naive Bayes	<a href="#">nb_5894c519c9</a>	0.8868
Generalized Linear Model	<a href="#">glm_15baeb3f99</a>	0.8697

# **What have we accomplished...**

**Lab 1:** Getting started with OML4Py

**Lab 2:** Select and manipulate data using the Transparency Layer

**Lab 3:** Using in-database algorithms and models

**Lab 4:** Use datastores to store Python objects

**Lab 5:** Run user-defined functions using Embedded Python Execution

**Lab 6:** Use AutoML

**Lab 6:** Bonus Round - Use OML AutoML UI

# Where to go from here?

# Helpful Links

## ORACLE MACHINE LEARNING ON O.COM

<https://www.oracle.com/machine-learning>



## OML TUTORIALS

**OML LiveLab:** [https://apexapps.oracle.com/pls/apex/dbpm/r/livelabs/view-workshop?p180\\_id=560](https://apexapps.oracle.com/pls/apex/dbpm/r/livelabs/view-workshop?p180_id=560)

**OML4Py LiveLab:** <https://apexapps.oracle.com/pls/apex/dbpm/r/livelabs/view-workshop?wid=786>

**Interactive tour:** <https://docs.oracle.com/en/cloud/paas/autonomous-database/oml-tour>

Two cards from the Oracle website. The top card is titled "Machine Learning on Autonomous Database Workshop" and describes exploring machine learning notebooks and prediction in applications on Autonomous Database. The bottom card is titled "Oracle Machine Learning for Python on Autonomous Database Workshop" and describes Python support in Autonomous Database, highlighting algorithms and performance. Both cards mention a 2-hour workshop length.

## OML OFFICE HOURS

<https://asktom.oracle.com/pls/apex/asktom.search?office=6801#sessions>



## ORACLE ANALYTICS CLOUD

<https://www.oracle.com/solutions/business-analytics/data-visualization/examples.html>



## OML4PY

[OML4Py \(2m video\)](#)

[OML4Py Introduction \(17m video\)](#)

[OML4Py Technical Brief](#)

[OML4Py User's Guide](#)

[Blog: Introducing OML4Py](#)

[GitHub Repository with Python notebooks](#)

## ORACLE AUTOML UI

[Oracle Machine Learning AutoML UI \(2m video\)](#)

[Oracle Machine Learning Demonstration \(6m video\)](#)

[OML AutoML UI Technical Brief](#)

[Blog: Introducing Oracle Machine Learning AutoML UI](#)

## OML SERVICES

[Oracle Machine Learning Services \(2m video\)](#)

[OML Services Technical Brief](#)

[Oracle Machine Learning Services Documentation](#)

[Blog: Introducing Oracle Machine Learning Services](#)

[GitHub Repository with OML Services examples](#)

# Oracle Machine Learning AskTom Office Hours sessions

<https://asktom.oracle.com/pls/apex/asktom.search?office=6801#sessions>



Sessions every Tuesday at 8AM Pacific Time

June 15 2021	OML usage highlight: Machine Learning Recommendations for Maintenance and Repair	
June 8 2021	Weekly Office Hours: OML on Autonomous Database - Ask & Learn	
June 1 2021	Hands-On Lab using Oracle Machine Learning for Python on Autonomous Database	
May 18 2021	Hands-On Lab using Oracle Machine Learning Services on Autonomous Database	
May 11 2021	OML usage highlight: Oracle Process Automation with Real-time OML Services scoring	

Machine Learning 101 sessions

## Previous Sessions

April 20 2021	OML usage highlight: Oracle Stream Analytics with Real-time OML Services scoring	
April 13 2021	OML usage highlight: Making Oracle Digital Assistant smarter with OML Services	
March 30 2021	OML feature highlight: OML AutoML UI for Automated Model Building	
March 23 2021	Weekly Office Hours: OML on Autonomous Database - Ask & Learn	
March 11 2021	OML feature highlight: OML Services on Autonomous for Model Deployment	
March 2 2021	Weekly Office Hours: OML on Autonomous Database - Ask & Learn	
February 23 2021	Hands-On Lab using Oracle Machine Learning for Python on Autonomous Database	
February 18 2021	Machine Learning 102 - Feature Extraction	
December 17 2020	Machine Learning 101: Feature Extraction	
November 5 2020	Machine Learning 102: Clustering	
October 28 2020	Oracle Machine Learning for R: An Introduction	
September 29 2020	Machine Learning 101: Clustering	
August 4 2020	Machine Learning 102: Regression	
July 21 2020	Machine Learning 101: Regression	
June 9 2020	Machine Learning 102: Classification	
May 19 2020	Machine Learning 101: Classification	
March 11 2020	Oracle Machine Learning Notebooks - Deep Dive	
February 18 2020	Oracle Machine Learning for Spark	
January 8 2020	Oracle Machine Learning for Python, with Demos	
December 11 2019	Oracle's Comprehensive Machine Learning Platform	

# Thank you !

---



**ORACLE**

