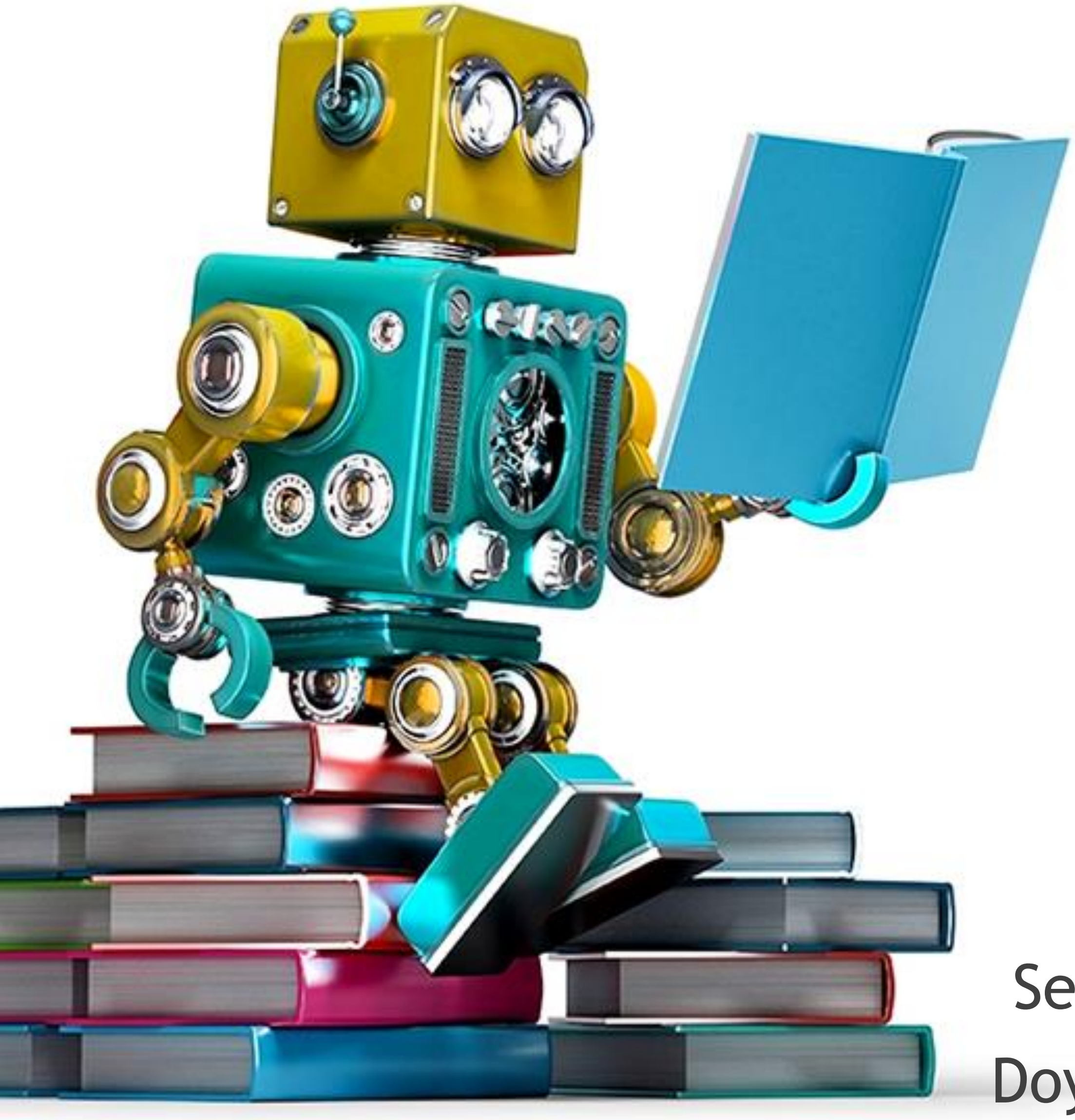


Augumented Intelligence through ML

Rajan Chandru
Senior Consultant- EBS
Doyen Systems Pvt. Ltd.

Prakash Ramamurthy
Competency Head - EBS
Doyen Systems Pvt. Ltd.

DOYENSYS

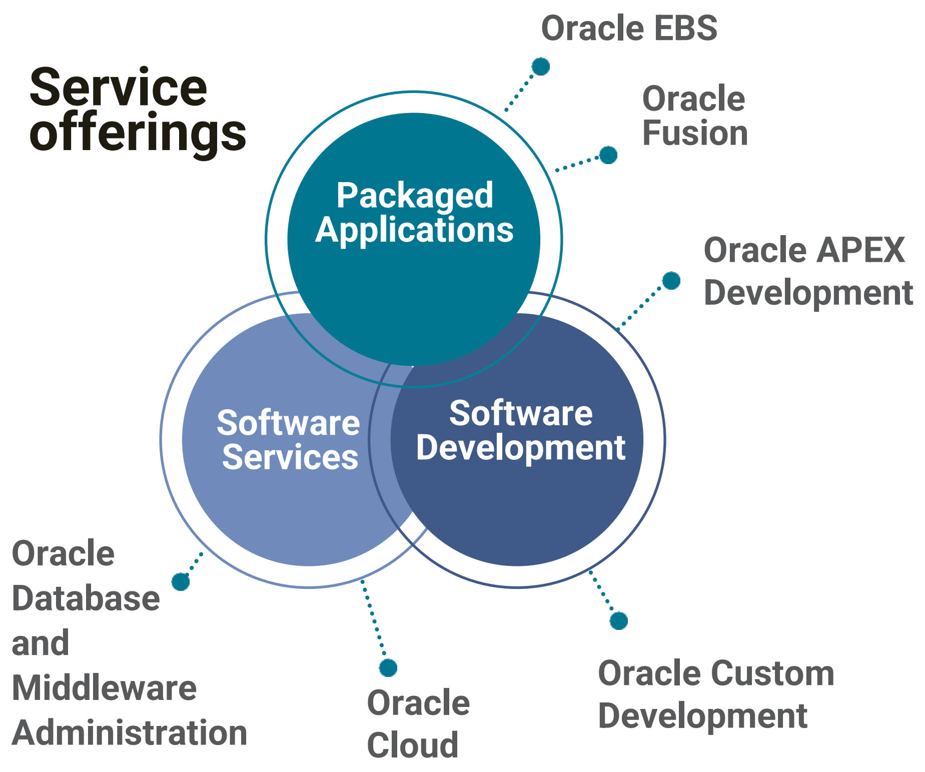
Servicing Customers for 13 Years

Team strength	Certified Professionals
250+	91% Oracle Certified MBA . PMI

30+ Delighted Customers

US . India . Canada

Service offerings



ORACLE PartnerNetwork ORACLE GoldPartner

Great Place To Work. Certified APR 2019-MAR 2020 INDIA

amazon

DATA INTENSITY

weight watchers

1·800 flowers.com

The Hackett Group World-Class Defined and Enabled

Wellpartner

HONDA

Panasonic ideas for life

Oracle Technology Books Authored by Doyens

Presenter(s) Info

Prakash Ramamurthy

Competency Head - EBS, Doyen Systems Pvt Ltd

- 24+ years of overall IT experience involving a spectrum of responsibilities - Program management, Technical Leadership , Technical Development, Business processes understanding
- Associated with Oracle & related Technologies for 20+ Years

Presenter(s) Info

DOYENSYS

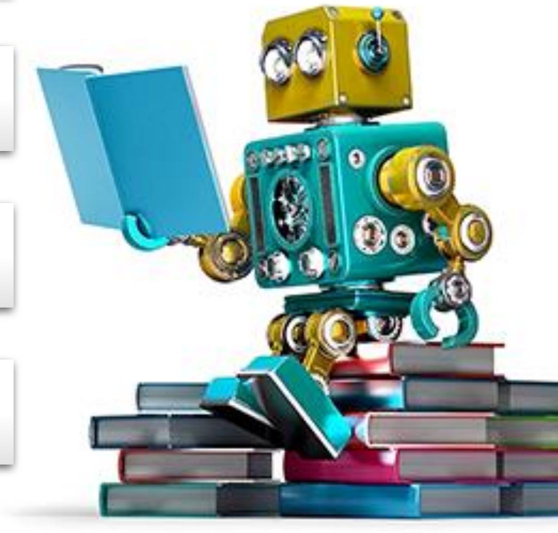
Rajan Chandru

Sr Consultant, Doyen Systems Pvt Ltd

- 8+ years of overall IT experience involving in APEX application development implementation and support
- Associated with Oracle APEX and other oracle related technologies for 8+ Years, with expertise in mutual fund domain .

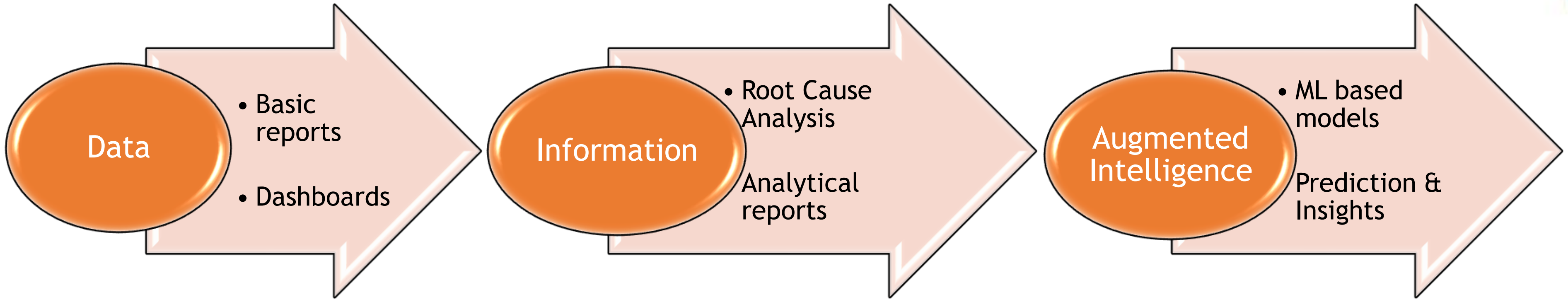
Agenda

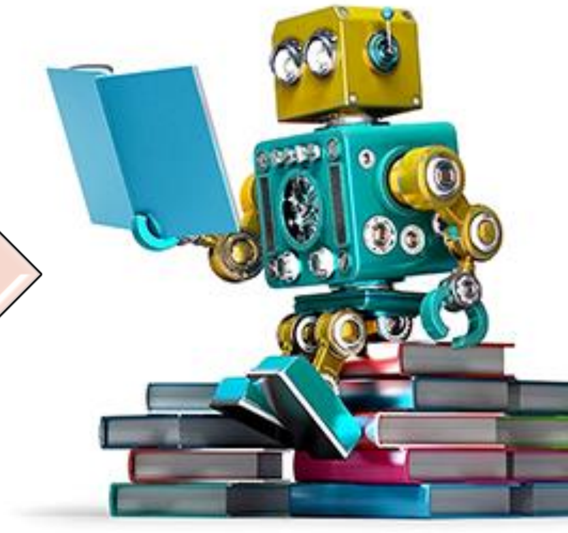
- Why ML?
- How ML can augment EBS?
- ML – Tools / Options
- ML Model Development Process
- Deployment options
- Use Case Demo
- Way forward



Why ML?

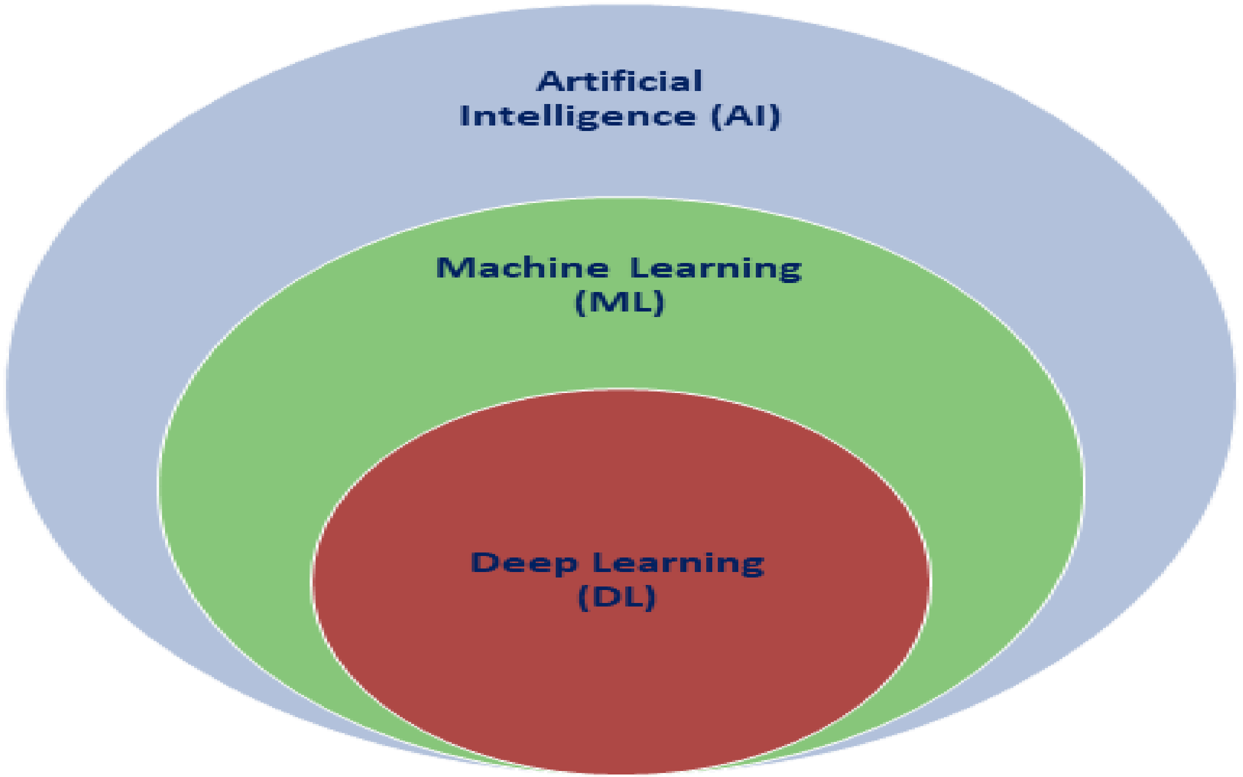
- ❑ In Today's world, Machine Learning is leveraged extensively , for instance
  - ❑ Online Shopping recommendations, Customer Service, Social Media , Sales Promotions etc.
- ❑ Enabling Systems to make educated guesses rather than keeping things Manual
- ❑ Oracle recent Product releases strengthen this fact further:
  - ❑ Autonomous Database
  - ❑ Security Tier : Anamoly Detection
  - ❑ Oracle Adaptive Intelligent (AI) Apps for Manufacturing
  - ❑ Configure, Price, Quote (CPQ)

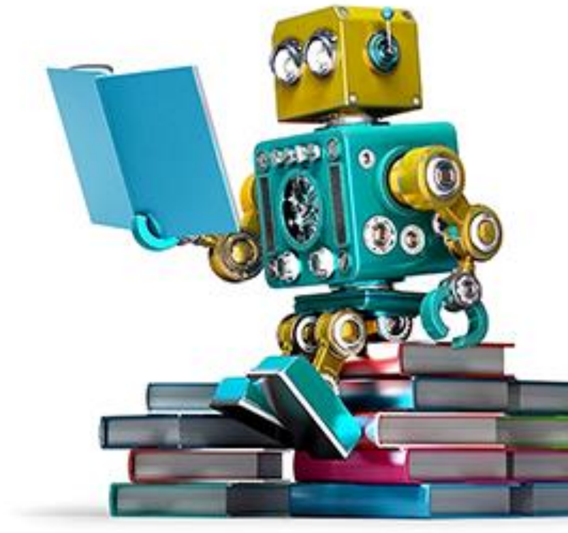




Why ML?

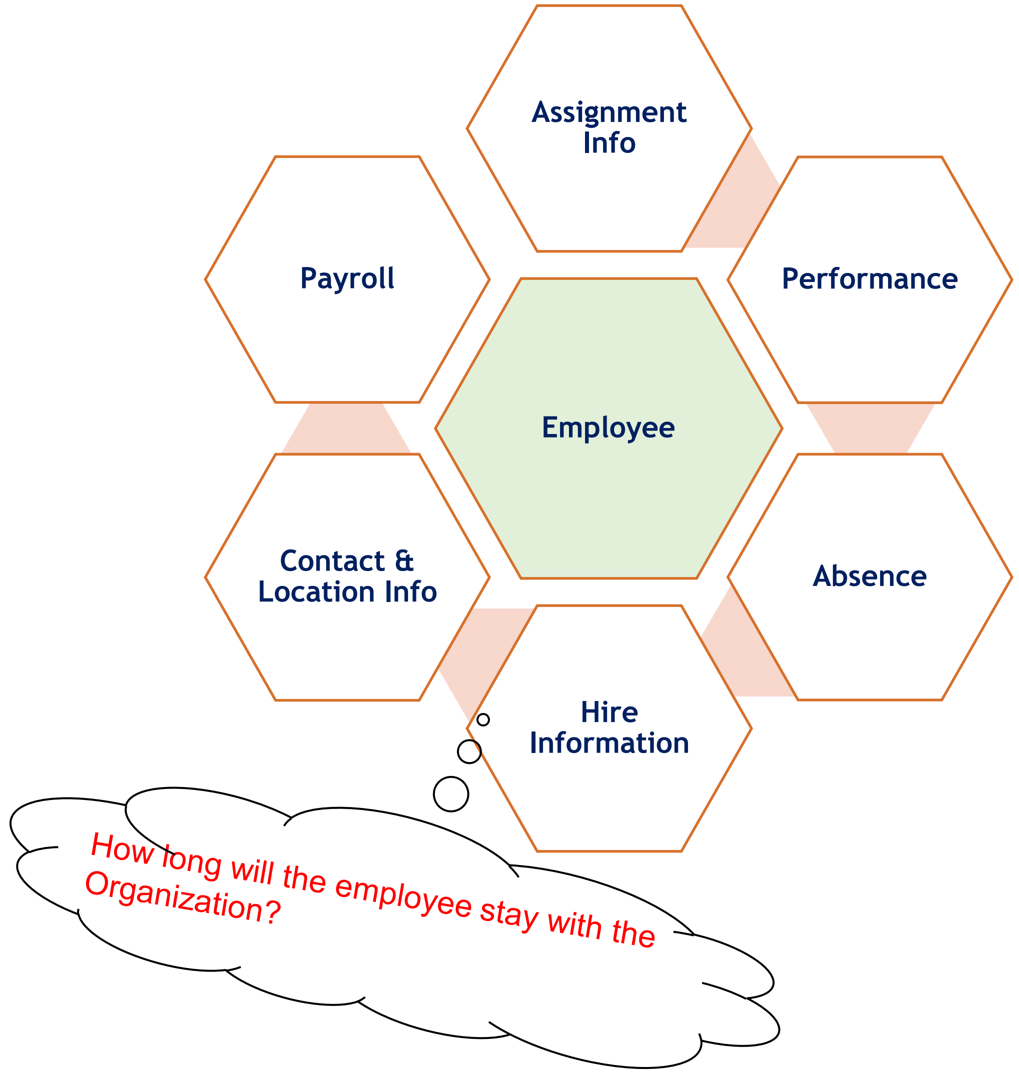
- ML provides the system , Ability to learn without explicit programming
- ML is the Subset of Artificial Intelligence (AI) which allows systems to mimic Human Intelligence
- Builds the platform for developing AI based Features / Solutions

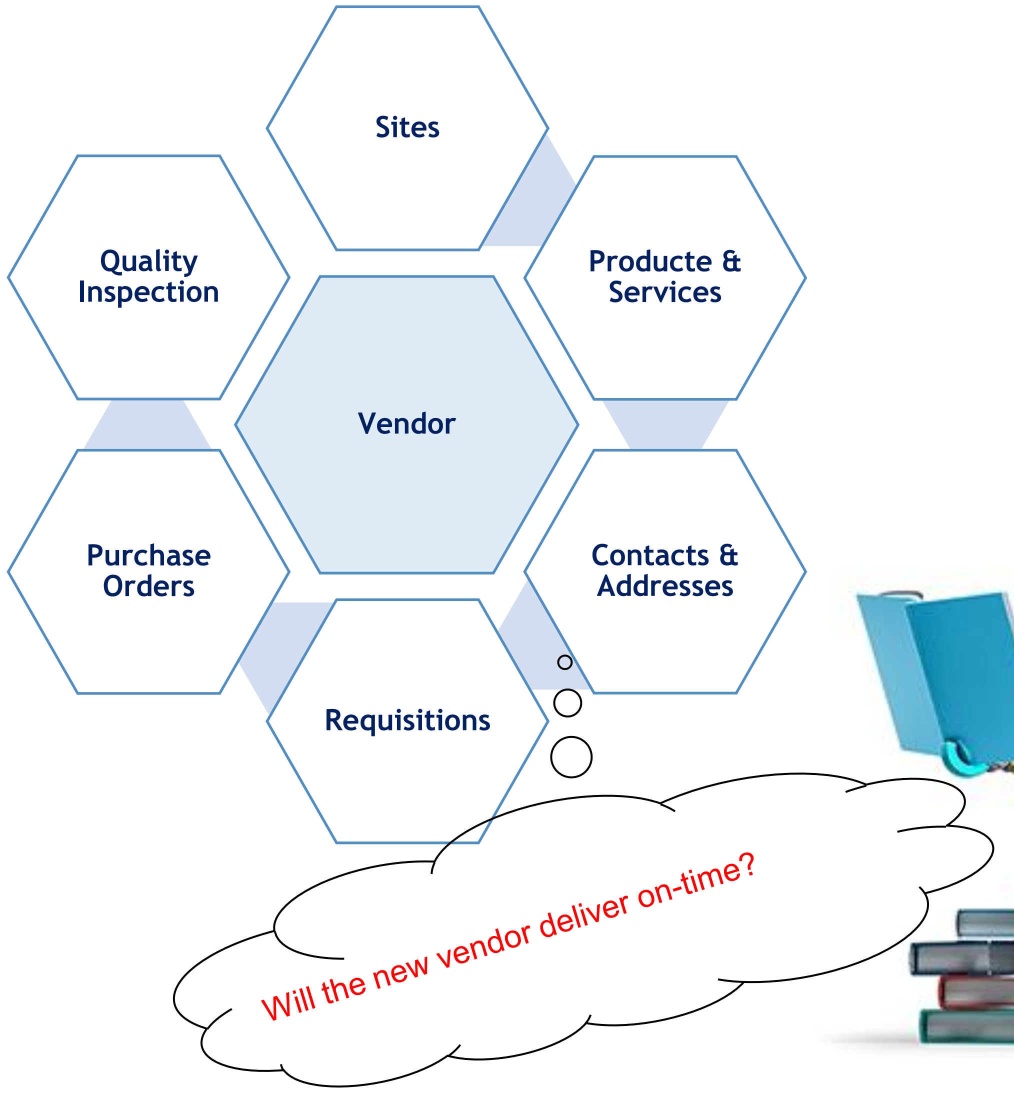


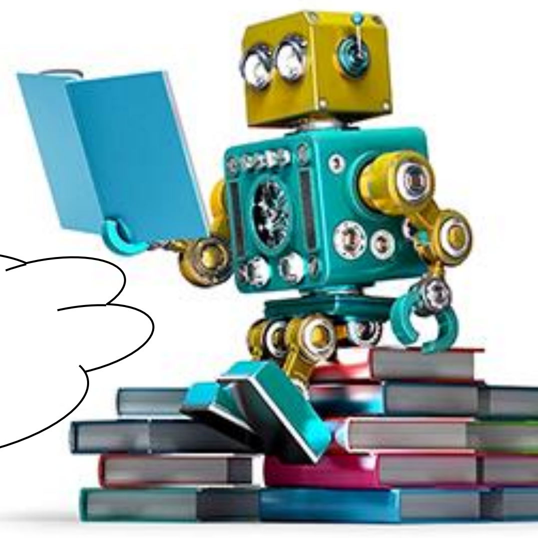


Why ML?

- EBS has the Data and ML has the Algorithms to bring - ***Insights and Predictability***

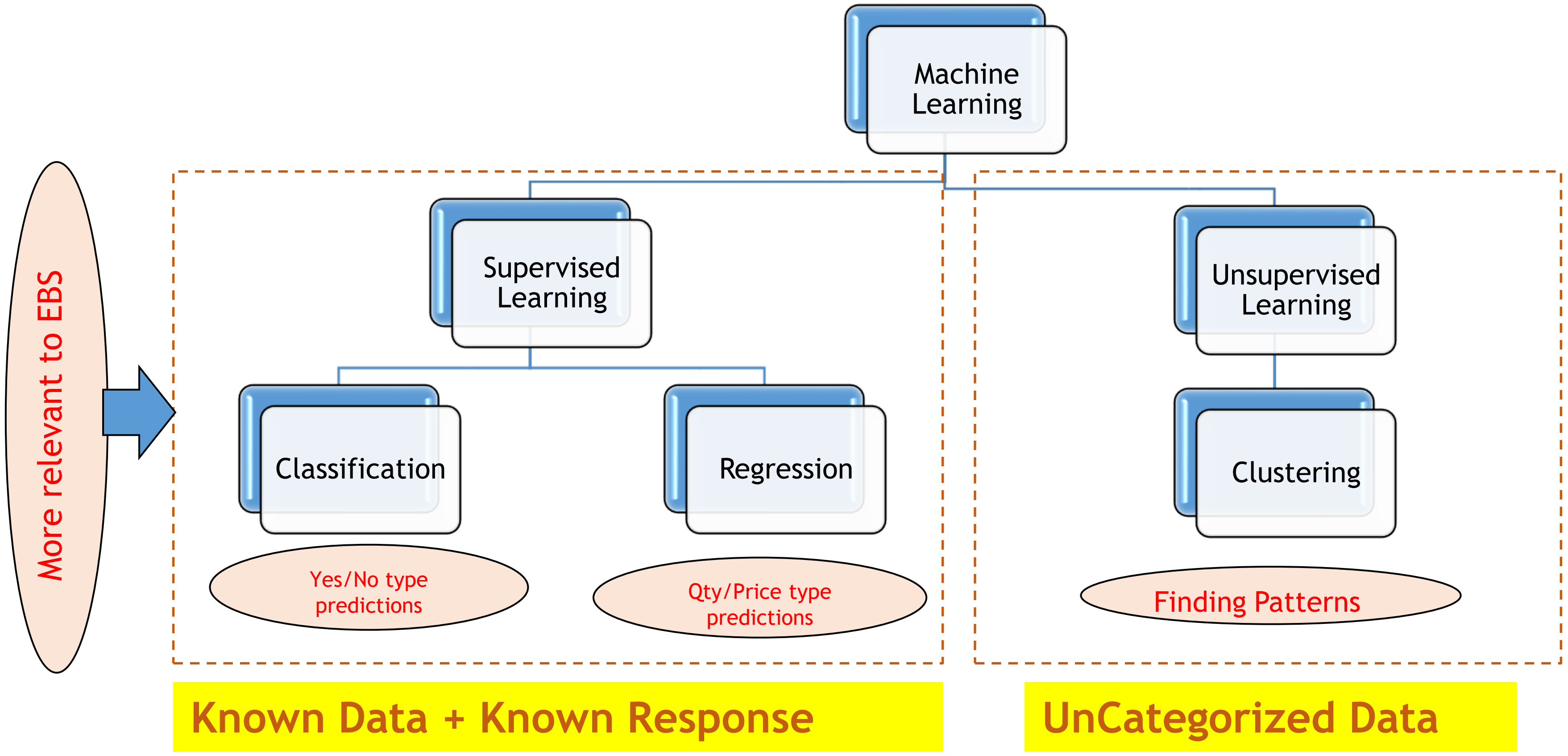


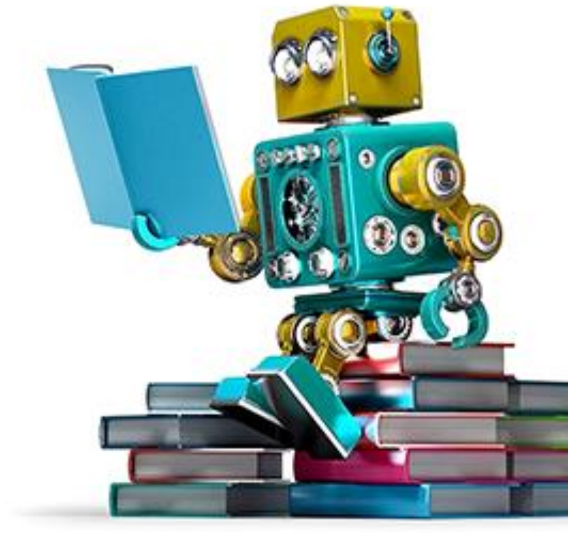




How ML can Augment EBS?

- ❑ EBS has Structured data and can leverage on “Supervised Learning” ML Algorithms





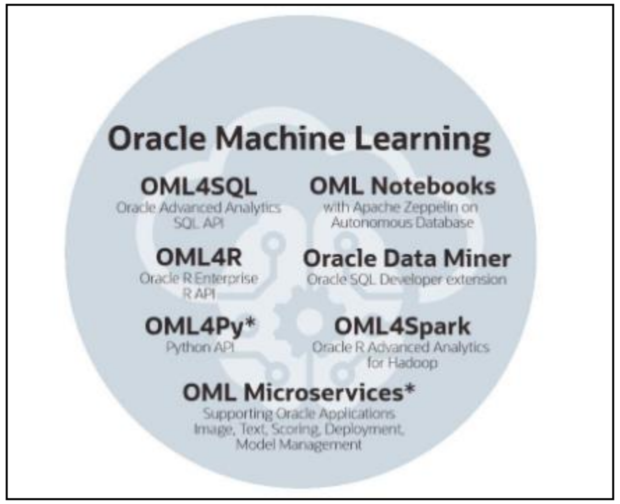
ML Tools & Options

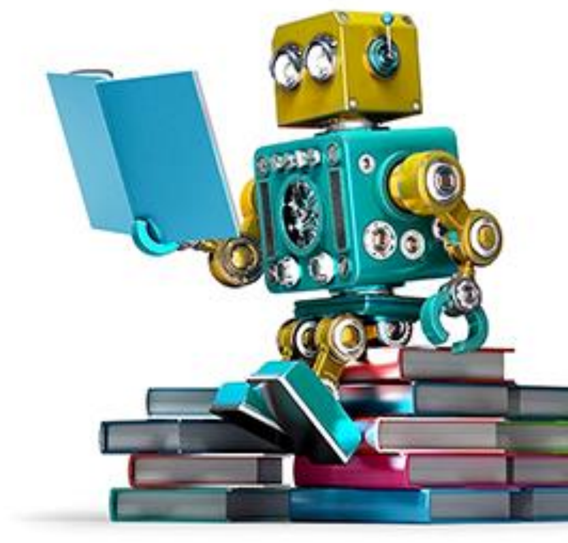
- ❑ **Oracle Machine Learning for SQL (OML4SQL)**
  - ❑ Algorithms are implemented as SQL functions and leverage the strengths of Oracle DB
  - ❑ Supports a "drag and drop" graphical user interface that is integrated with Oracle SQL Developer and is capable of generating SQL scripts from user-created analytics workflows.

- ❑ **Oracle Machine Learning for R** (OML4R)
  - ❑ R provides a suite of software packages for data manipulation, graphics, statistical functions, and machine learning algorithms
  - ❑ OML4R extends R's capabilities through direct DB access,in-database ML algorithms

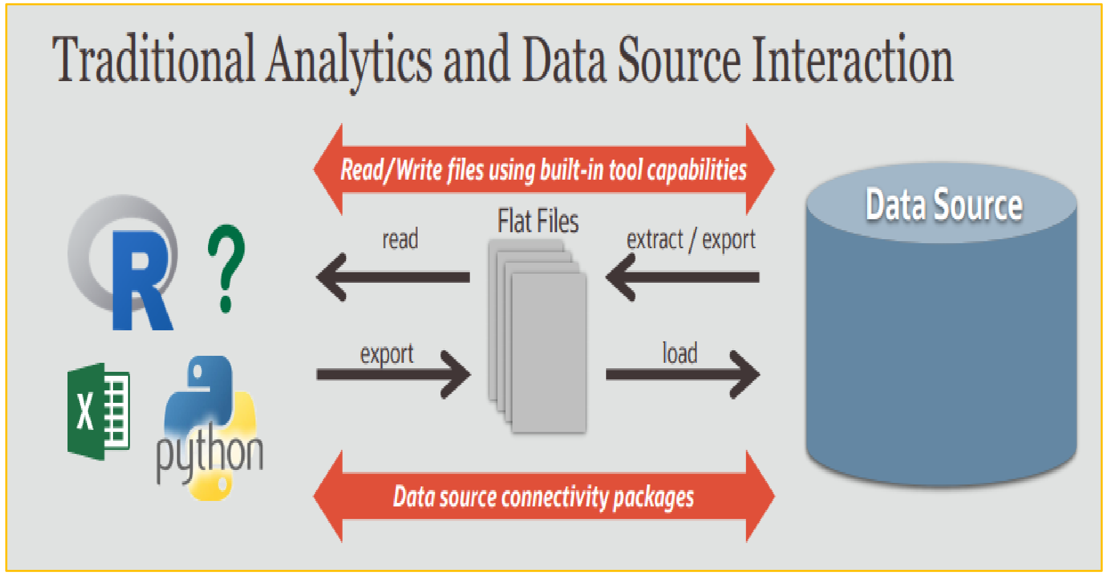
- ❑ **Oracle Machine Learning for Python**(OML4Py)
  - ❑ a component of the Oracle Advanced Analytics Option
  - ❑ To be made available as part of Oracle Database 19c

- ❑ **Oracle Machine Learning Notebooks**
  - ❑ part of Oracle Autonomous Database, providing Apache Zeppelin-based notebooks for SQL users of Oracle Autonomous Data Warehouse and Oracle Autonomous Transaction Processing


Oracle Machine Learning
OML4SQL
Oracle Advanced Analytics SQL API
OML Notebooks
with Apache Zeppelin on Autonomous Database
OML4R
Oracle R Enterprise R API
Oracle Data Miner
Oracle SQL Developer extension
OML4Py*
Python API
OML4Spark
Oracle R Advanced Analytics for Hadoop
OML Microservices*
Supporting Oracle Applications Image, Text, Scoring, Deployment, Model Management

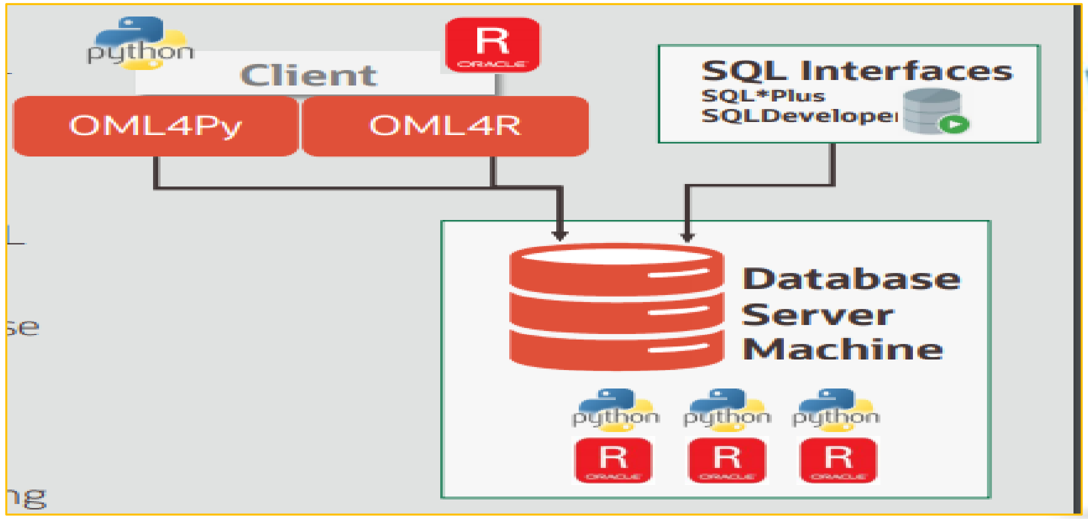


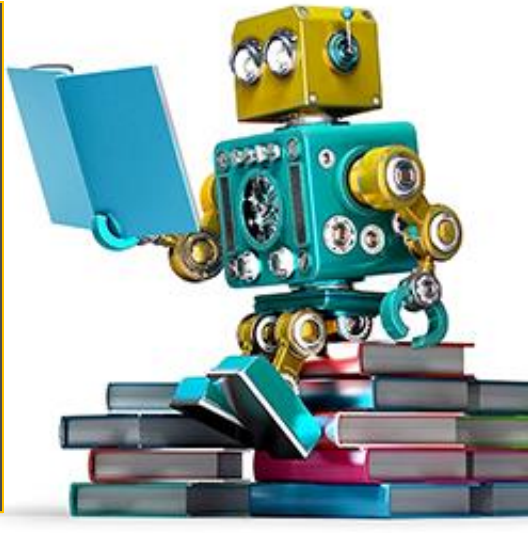
ML Tools & Options


Traditional Analytics and Data Source Interaction
Read/Write files using built-in tool capabilities
read
Flat Files
extract / export
export
load
Data Source
?
python
Data source connectivity packages

Traditional Interaction ML tools with DB

ML tools interaction with Oracle DB using OML

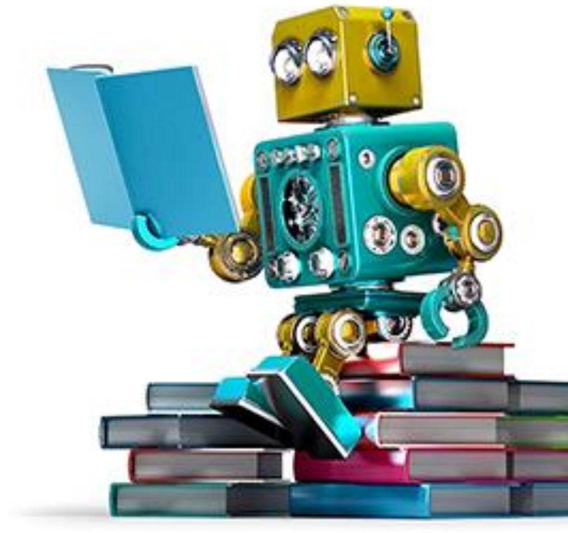




ML Tools & Options

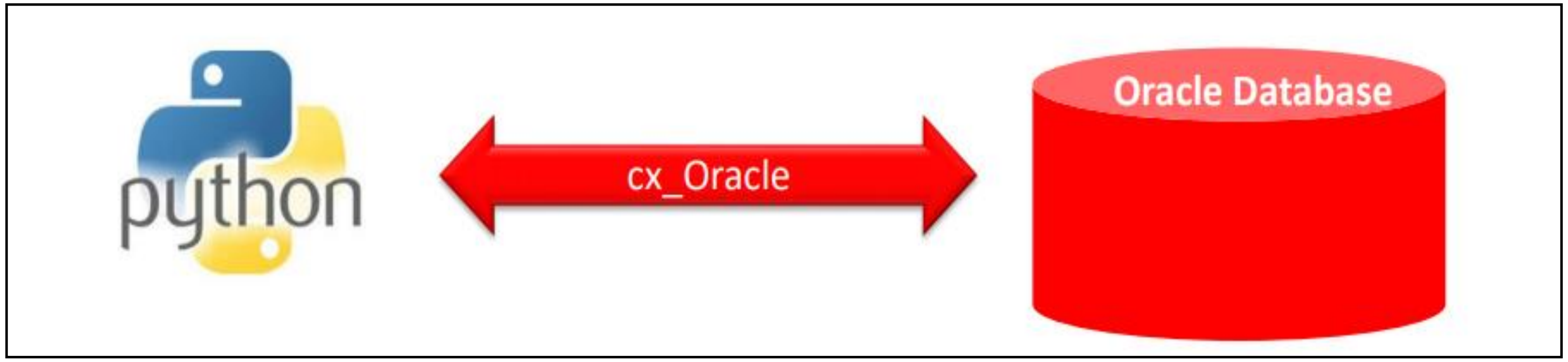
❑ Industry level common ML languages and Development Tools

ML Programming Languages	Python IDE
• R	• Jupyter Notebook
• Python	• PyCharm
• C++	• Spyder
• Java	• PyDev
• JavaScript	• Idle
• Julia	• Wing
• Scala	• eRic
• MATLAB	• Rodeo
• Shell	

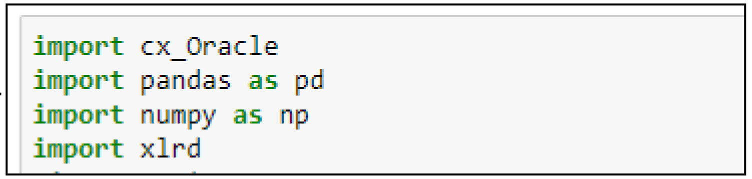


ML Tools & Options

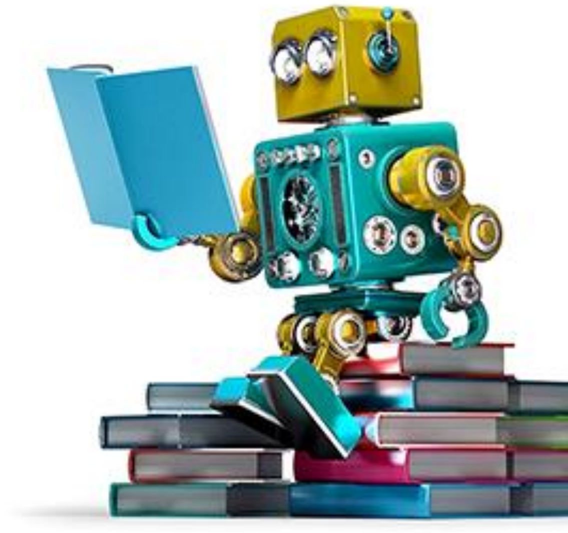
- **cx_Oracle** is a Oracle provided Python Package enabling connectivity to Oracle DB
  - Open Source , publicly available
  - Allows to Execute SQL statements from Python
  - Allows for DML Statements **Insert / Update / Delete**



Can be accessed by simple Import stmt in Python code

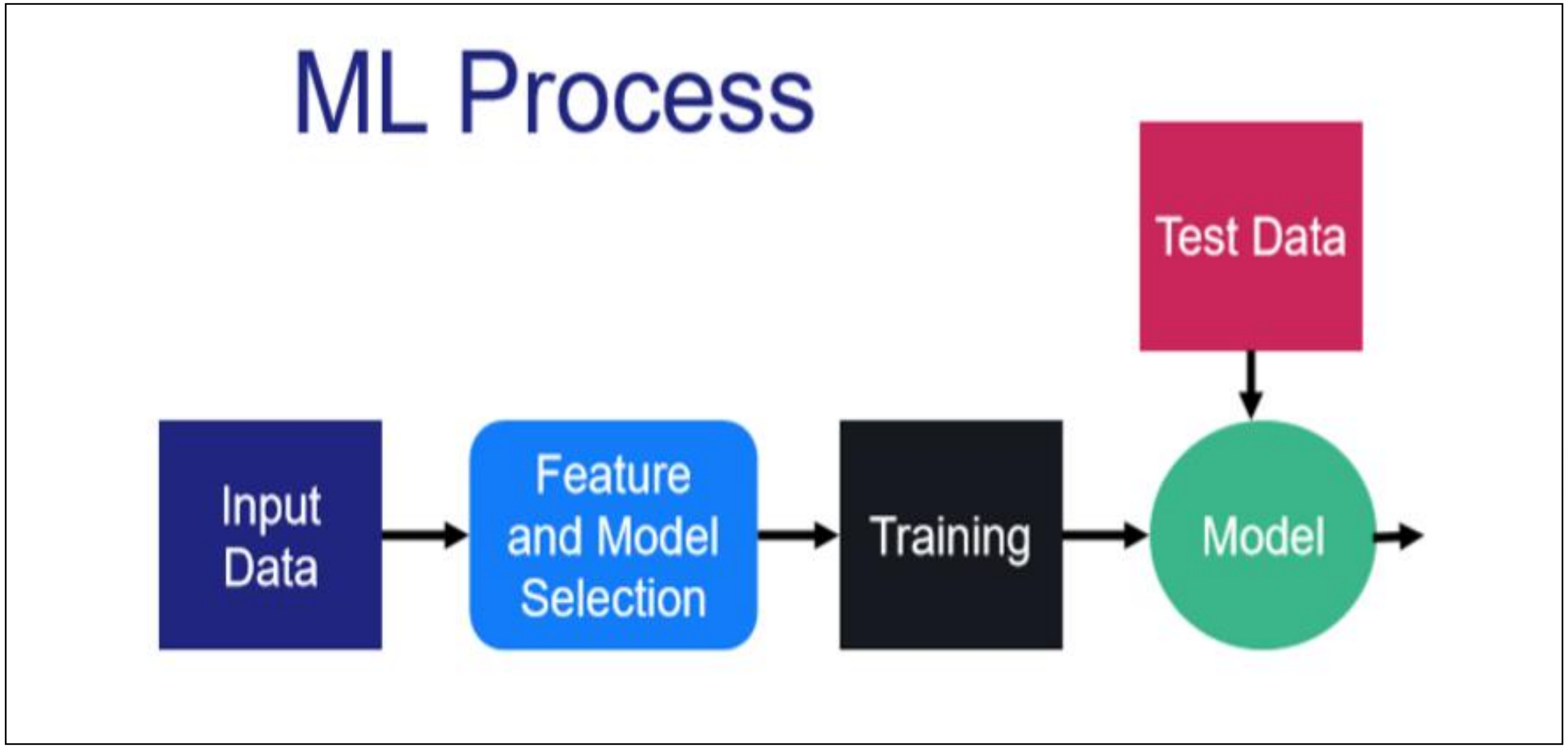


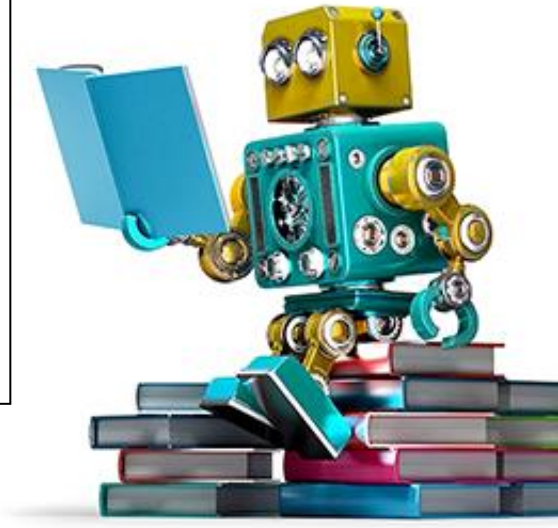
```
import cx_Oracle
import pandas as pd
import numpy as np
import xlrd
```



ML Model Development Process

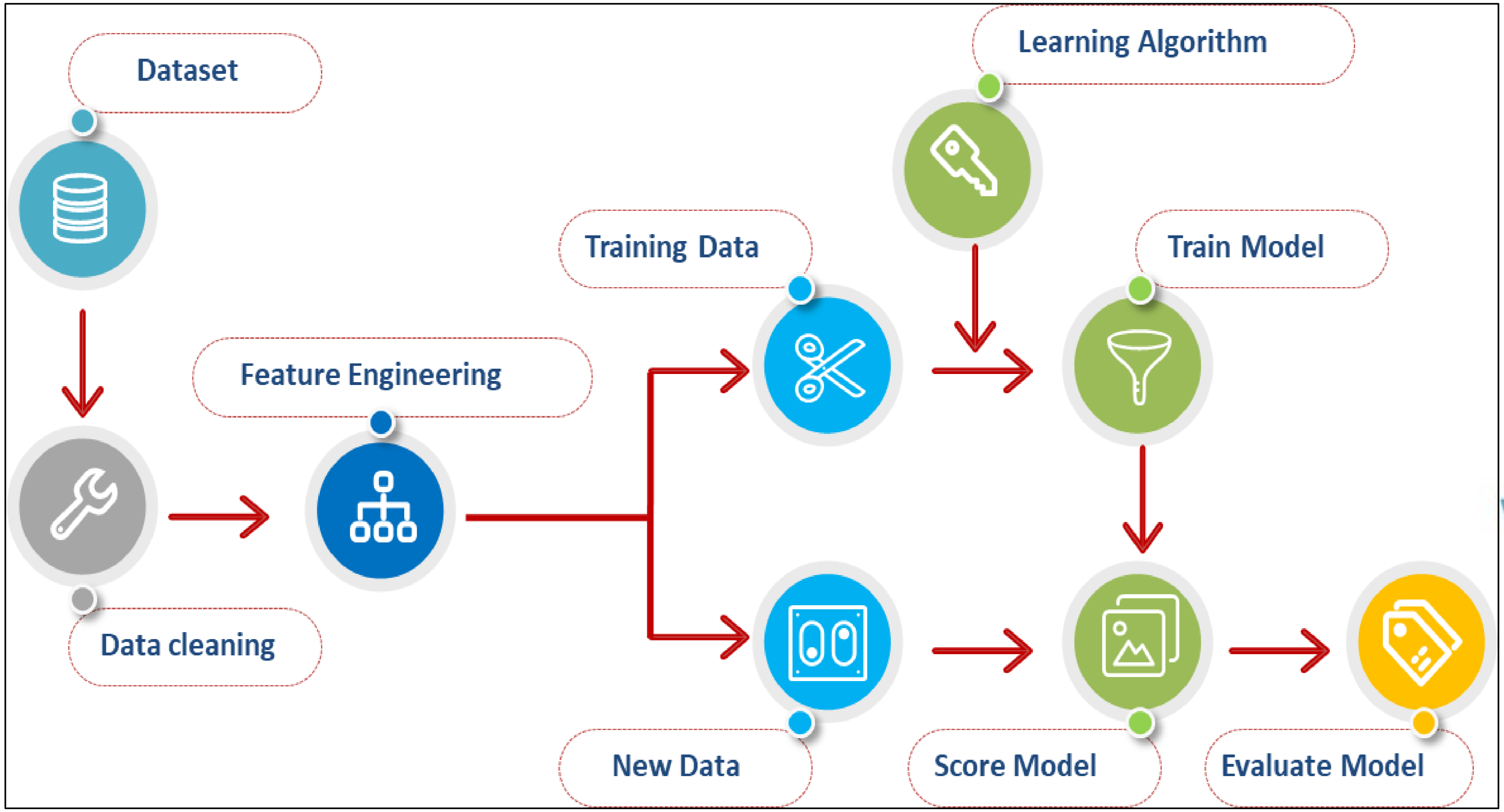
- ❑ Common Practice followed in development of ML based models

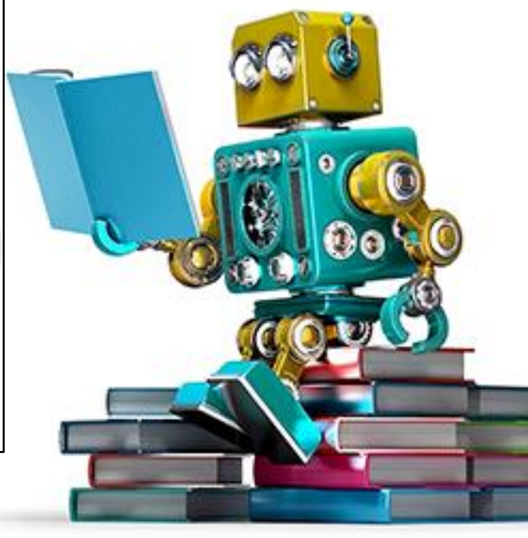




ML Model Development Process

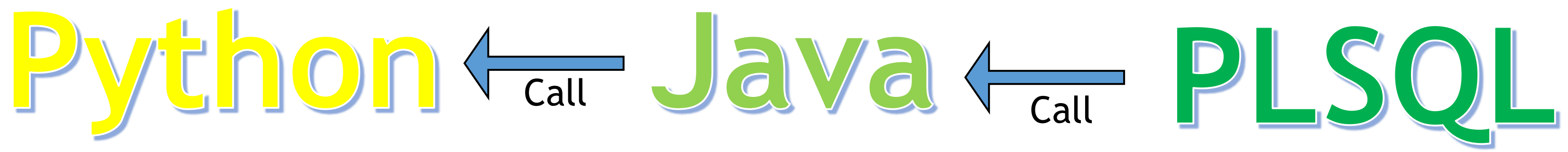
❑ Common Practice followed for Handling the Data

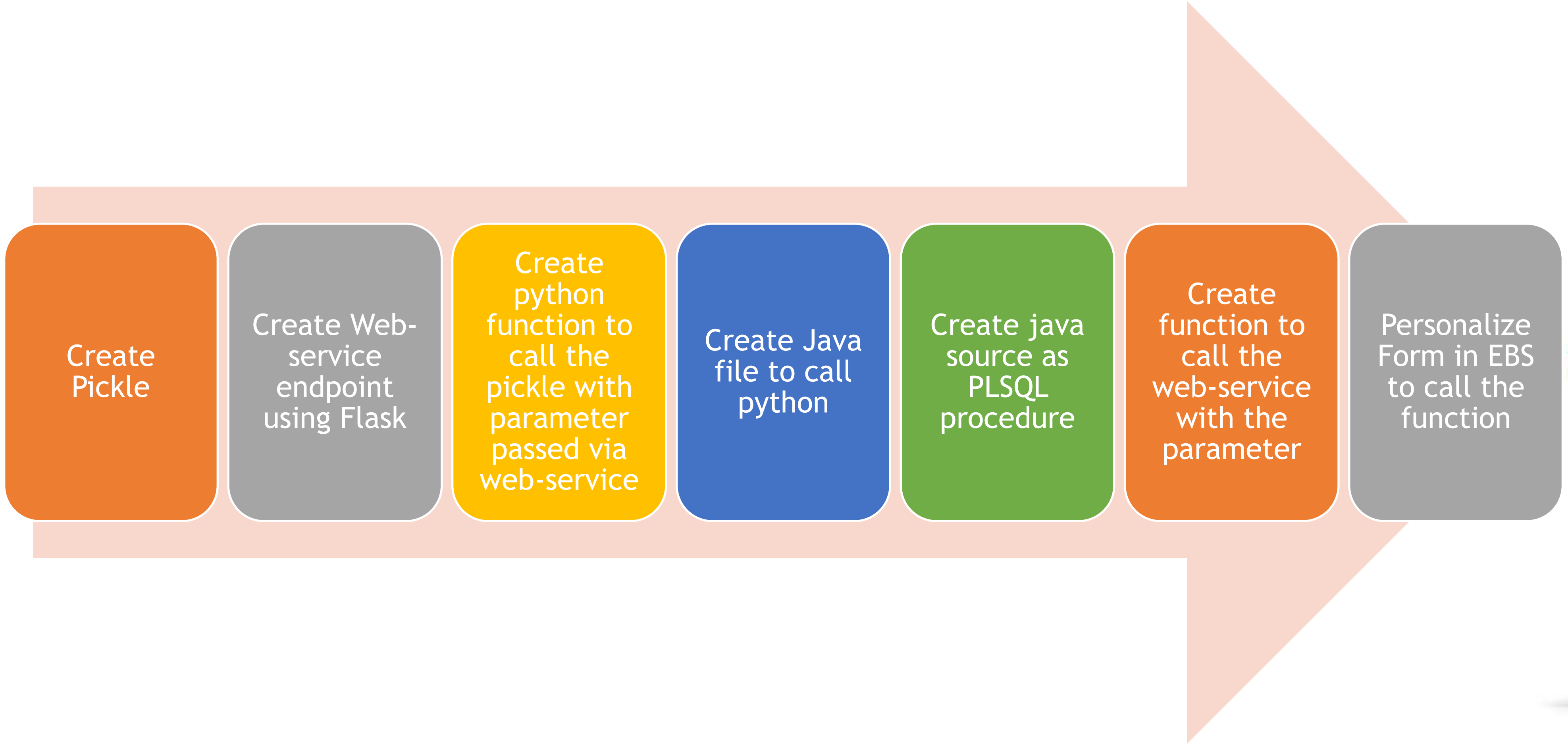


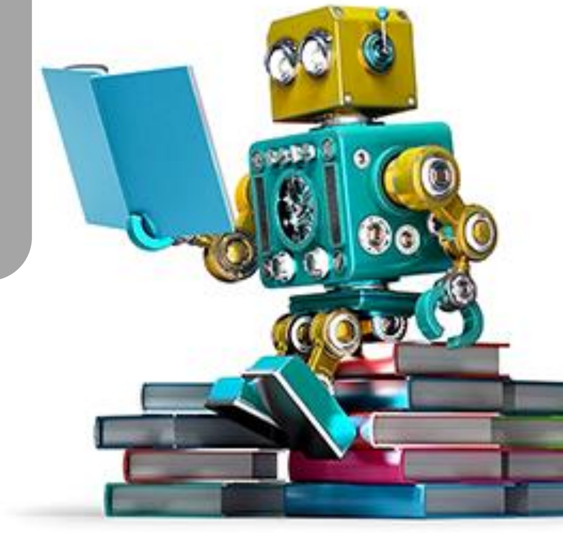


ML Deployment Options

- Option-1 : Using Web-services

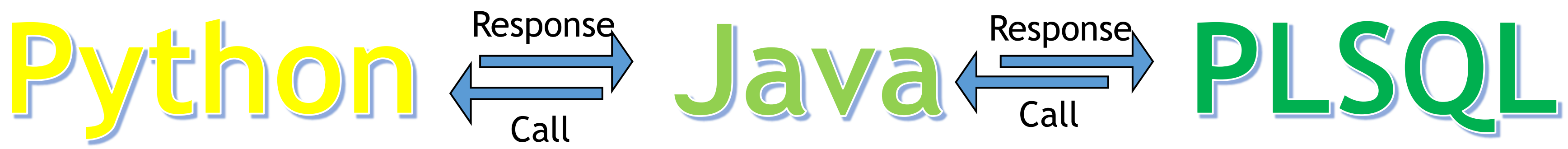


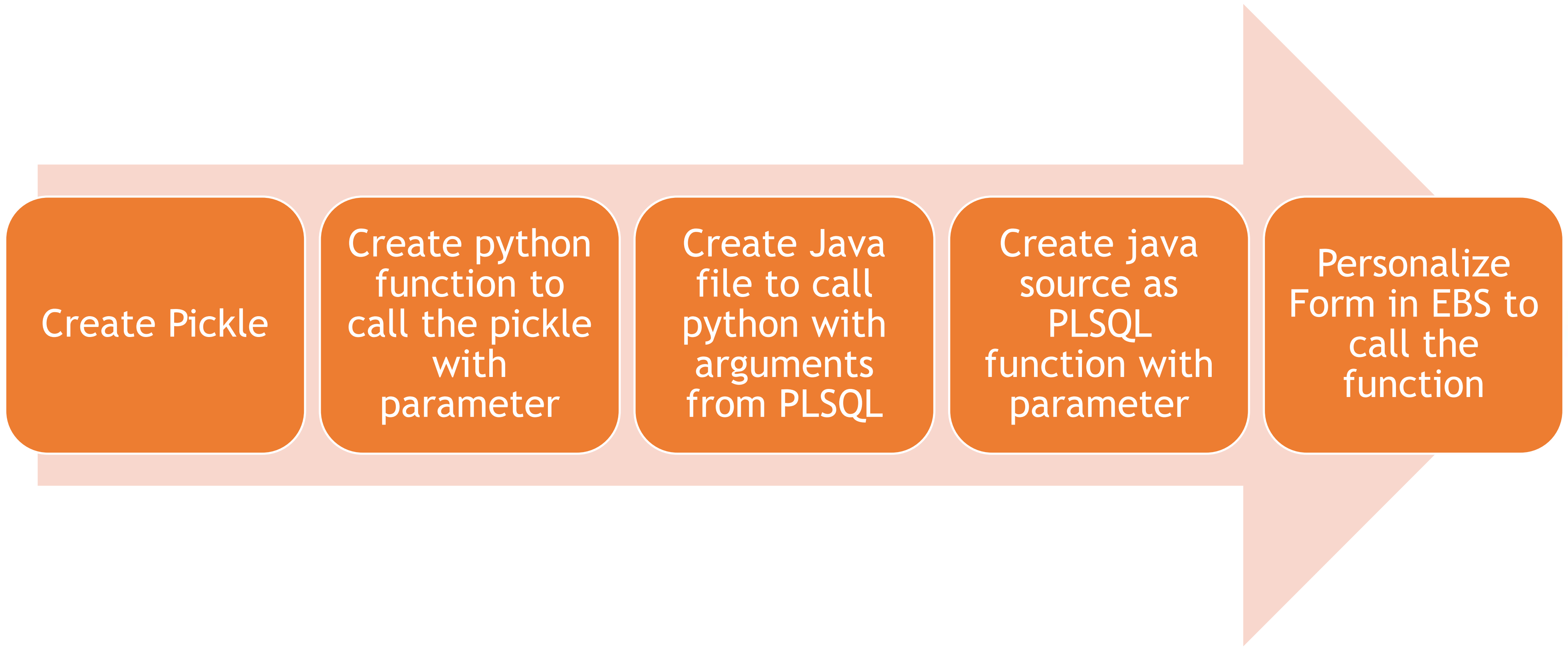

Create Pickle
Create Web-service endpoint using Flask
Create python function to call the pickle with parameter passed via web-service
Create Java file to call python
Create java source as PLSQL procedure
Create function to call the web-service with the parameter
Personalize Form in EBS to call the function

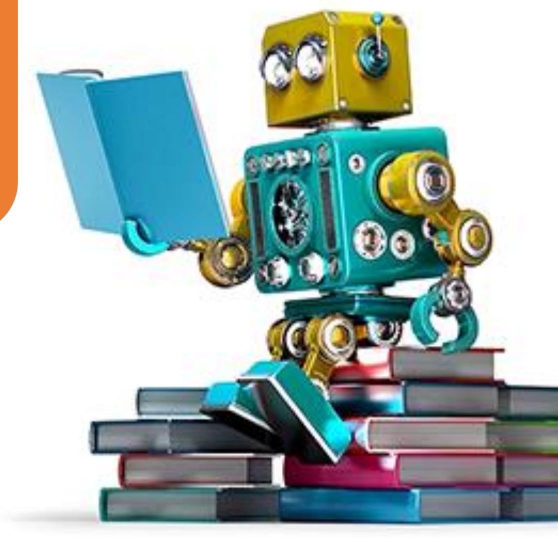


ML Deployment Options

- Option-2 : Direct invocation

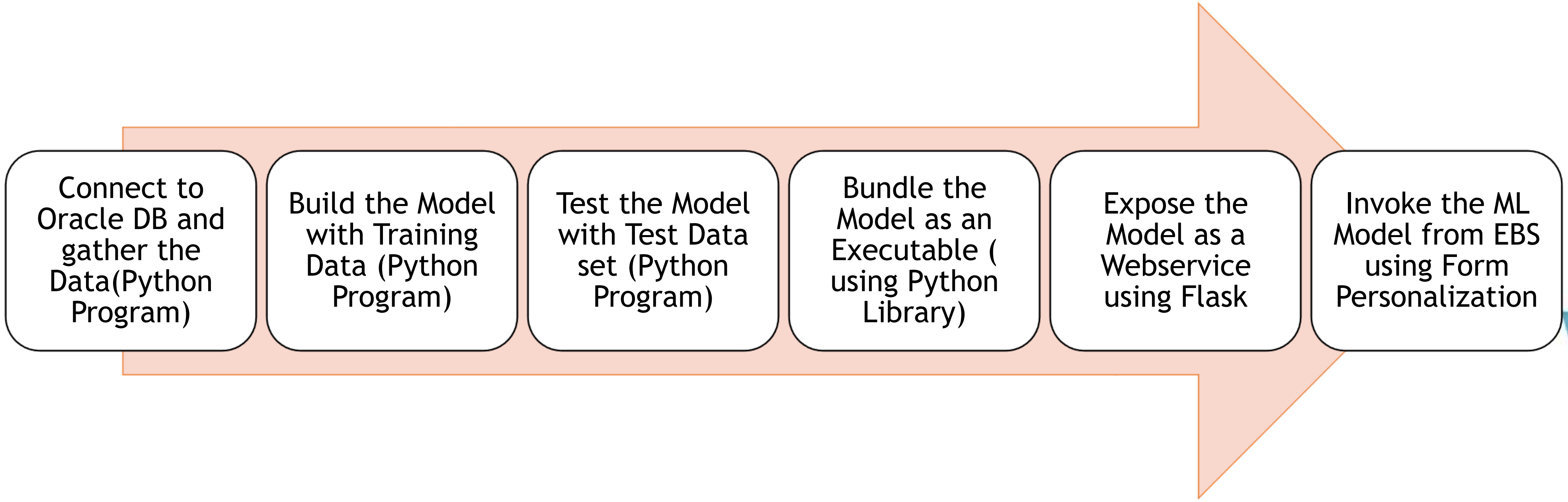



Create Pickle
Create python function to call the pickle with parameter
Create Java file to call python with arguments from PLSQL
Create java source as PLSQL function with parameter
Personalize Form in EBS to call the function

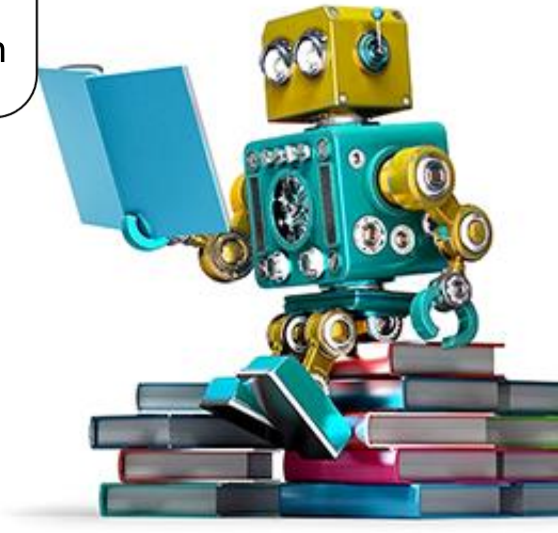


Use Case

- ❑ Classification based ML Model to predict
  - ❑ ***If the Payment from a Customer is Likely to be Delayed [Yes / No]***



Connect to Oracle DB and gather the Data(Python Program)
Build the Model with Training Data (Python Program)
Test the Model with Test Data set (Python Program)
Bundle the Model as an Executable (using Python Library)
Expose the Model as a Webservice using Flask
Invoke the ML Model from EBS using Form Personalization



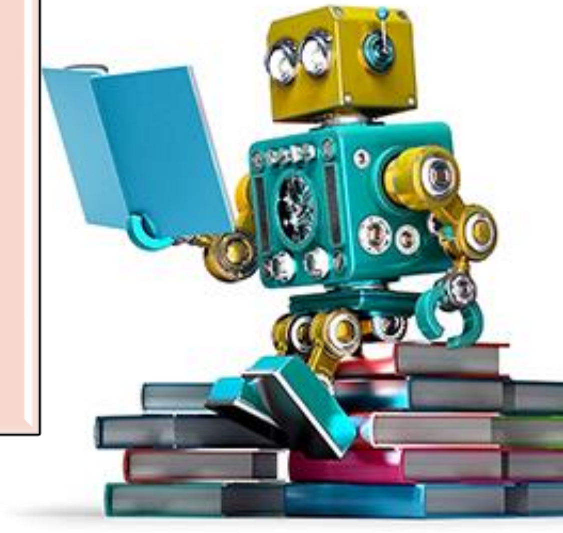
Use Case

Technologies Needed

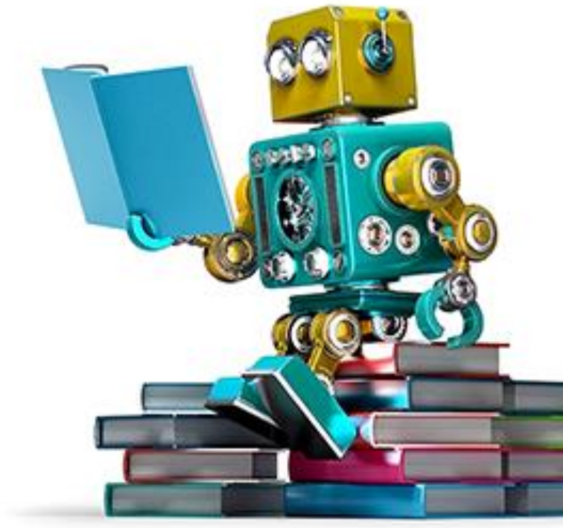
- Python (2.7 and above)
- JAVA
- PLSQL

Pre-requisites

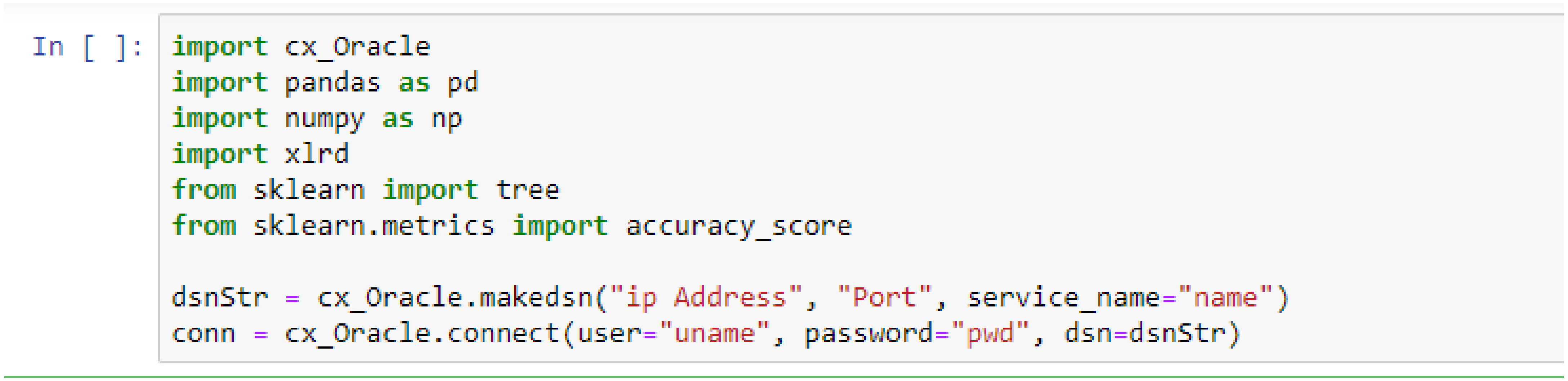
- Install Python in oracle database
- Cx_Oracle and JSON Package need to be installed in Python
- Install java in oracle database



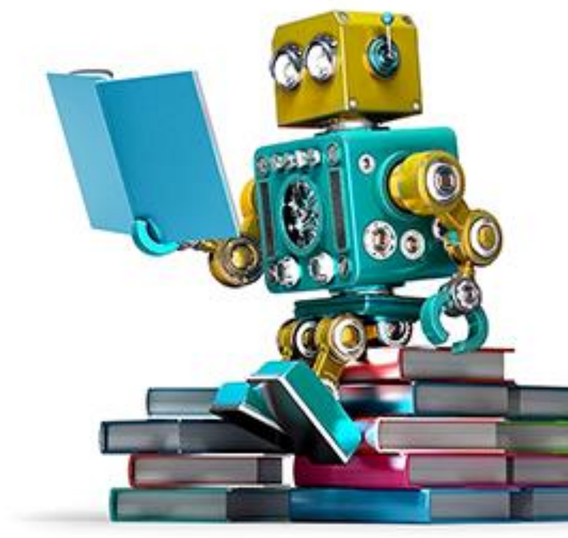
Use Case Demo



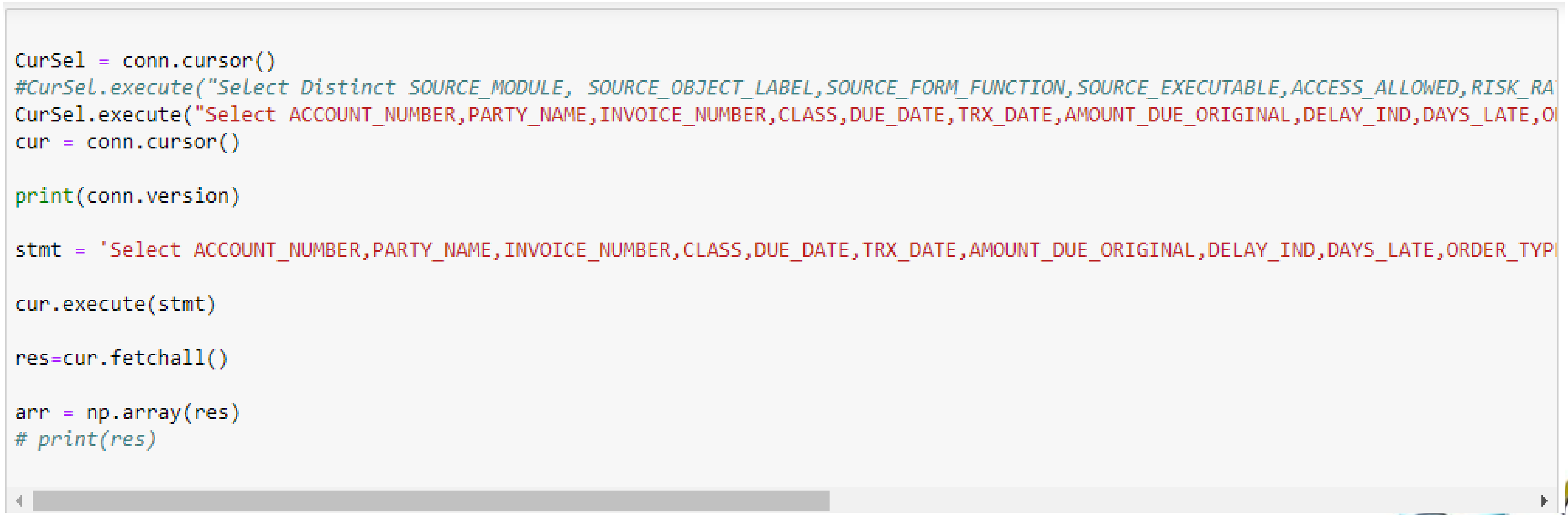
Use Case - Python Code



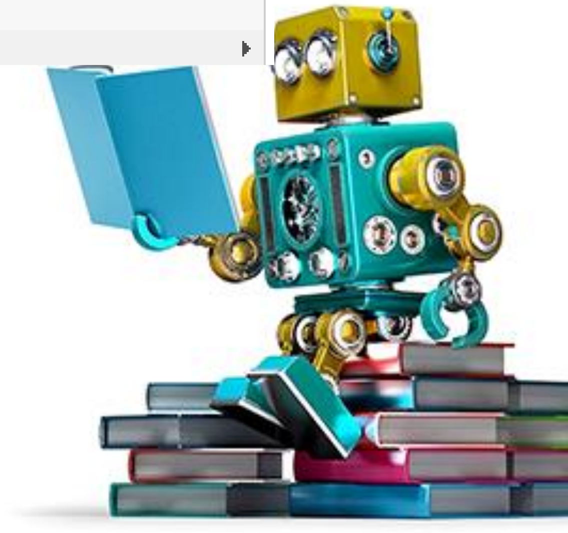
```
In [ ]: import cx_Oracle
        import pandas as pd
        import numpy as np
        import xlrd
        from sklearn import tree
        from sklearn.metrics import accuracy_score

        dsnStr = cx_Oracle.makedsn("ip Address", "Port", service_name="name")
        conn = cx_Oracle.connect(user="uname", password="pwd", dsn=dsnStr)
```



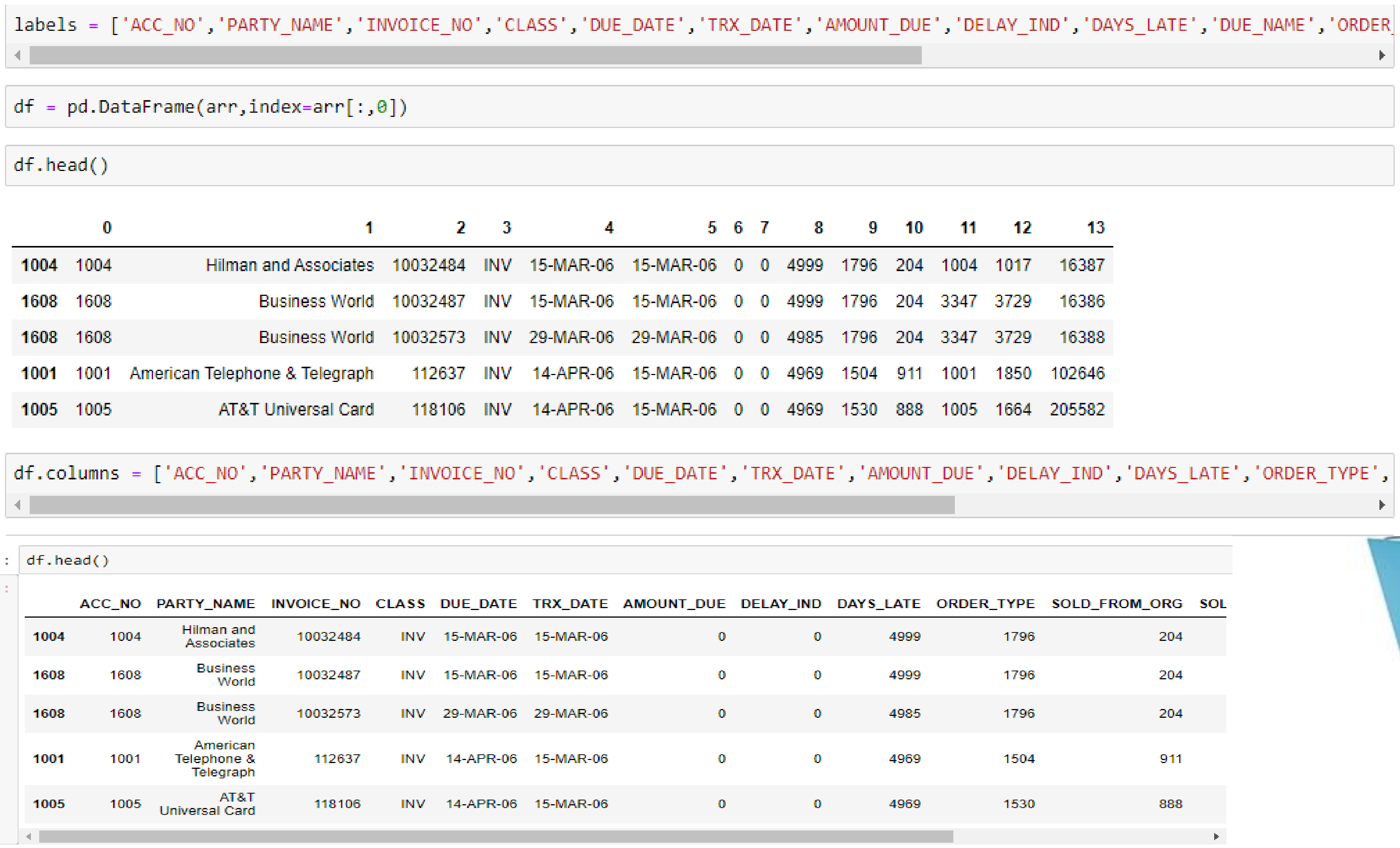
Use Case - Python Code



```
CurSel = conn.cursor()
#CurSel.execute("Select Distinct SOURCE_MODULE, SOURCE_OBJECT_LABEL,SOURCE_FORM_FUNCTION,SOURCE_EXECUTABLE,ACCESS_ALLOWED,RISK_RA
CurSel.execute("Select ACCOUNT_NUMBER,PARTY_NAME,INVOICE_NUMBER,CLASS,DUE_DATE,TRX_DATE,AMOUNT_DUE_ORIGINAL,DELAY_IND,DAYS_LATE,O
cur = conn.cursor()

print(conn.version)

stmt = 'Select ACCOUNT_NUMBER,PARTY_NAME,INVOICE_NUMBER,CLASS,DUE_DATE,TRX_DATE,AMOUNT_DUE_ORIGINAL,DELAY_IND,DAYS_LATE,ORDER_TYP

cur.execute(stmt)

res=cur.fetchall()

arr = np.array(res)
# print(res)
```



Use Case - Python Code



```
labels = ['ACC_NO','PARTY_NAME','INVOICE_NO','CLASS','DUE_DATE','TRX_DATE','AMOUNT_DUE','DELAY_IND','DAYS_LATE','DUE_NAME','ORDER
```

```
df = pd.DataFrame(arr,index=arr[:,0])
```

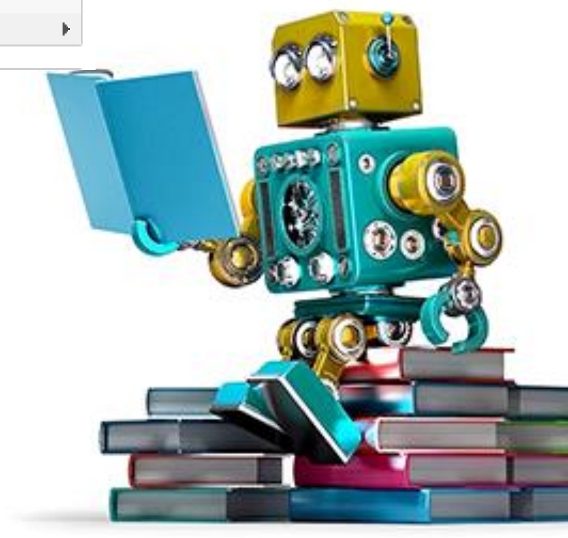
```
df.head()
```

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
1004	1004	Hilman and Associates	10032484	INV	15-MAR-06	15-MAR-06	0	0	4999	1796	204	1004	1017	16387
1608	1608	Business World	10032487	INV	15-MAR-06	15-MAR-06	0	0	4999	1796	204	3347	3729	16386
1608	1608	Business World	10032573	INV	29-MAR-06	29-MAR-06	0	0	4985	1796	204	3347	3729	16388
1001	1001	American Telephone & Telegraph	112637	INV	14-APR-06	15-MAR-06	0	0	4969	1504	911	1001	1850	102646
1005	1005	AT&T Universal Card	118106	INV	14-APR-06	15-MAR-06	0	0	4969	1530	888	1005	1664	205582

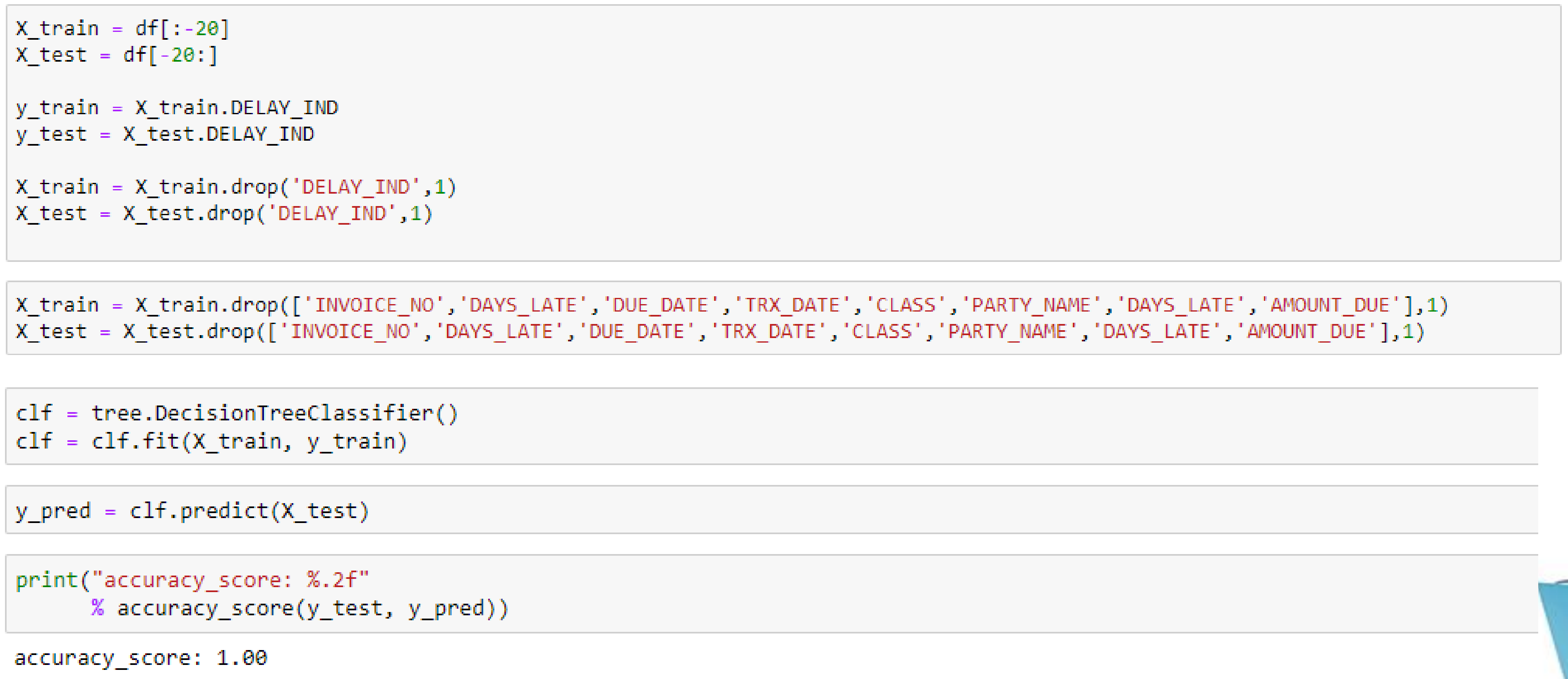
```
df.columns = ['ACC_NO','PARTY_NAME','INVOICE_NO','CLASS','DUE_DATE','TRX_DATE','AMOUNT_DUE','DELAY_IND','DAYS_LATE','ORDER_TYPE',
```

```
df.head()
```

	ACC_NO	PARTY_NAME	INVOICE_NO	CLASS	DUE_DATE	TRX_DATE	AMOUNT_DUE	DELAY_IND	DAYS_LATE	ORDER_TYPE	SOLD_FROM_ORG	SOL
1004	1004	Hilman and Associates	10032484	INV	15-MAR-06	15-MAR-06	0	0	4999	1796	204	
1608	1608	Business World	10032487	INV	15-MAR-06	15-MAR-06	0	0	4999	1796	204	
1608	1608	Business World	10032573	INV	29-MAR-06	29-MAR-06	0	0	4985	1796	204	
1001	1001	American Telephone & Telegraph	112637	INV	14-APR-06	15-MAR-06	0	0	4969	1504	911	
1005	1005	AT&T Universal Card	118106	INV	14-APR-06	15-MAR-06	0	0	4969	1530	888	



Use Case - Python Code



```
X_train = df[:-20]
X_test = df[-20:]

y_train = X_train.DELAY_IND
y_test = X_test.DELAY_IND

X_train = X_train.drop('DELAY_IND',1)
X_test = X_test.drop('DELAY_IND',1)
```

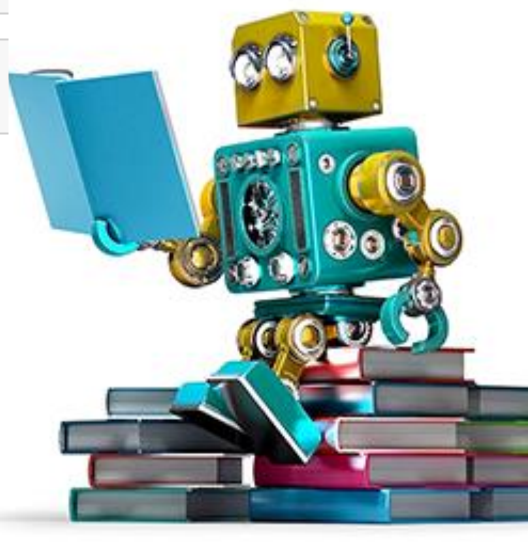
```
X_train = X_train.drop(['INVOICE_NO','DAYS_LATE','DUE_DATE','TRX_DATE','CLASS','PARTY_NAME','DAYS_LATE','AMOUNT_DUE'],1)
X_test = X_test.drop(['INVOICE_NO','DAYS_LATE','DUE_DATE','TRX_DATE','CLASS','PARTY_NAME','DAYS_LATE','AMOUNT_DUE'],1)
```

```
clf = tree.DecisionTreeClassifier()
clf = clf.fit(X_train, y_train)
```

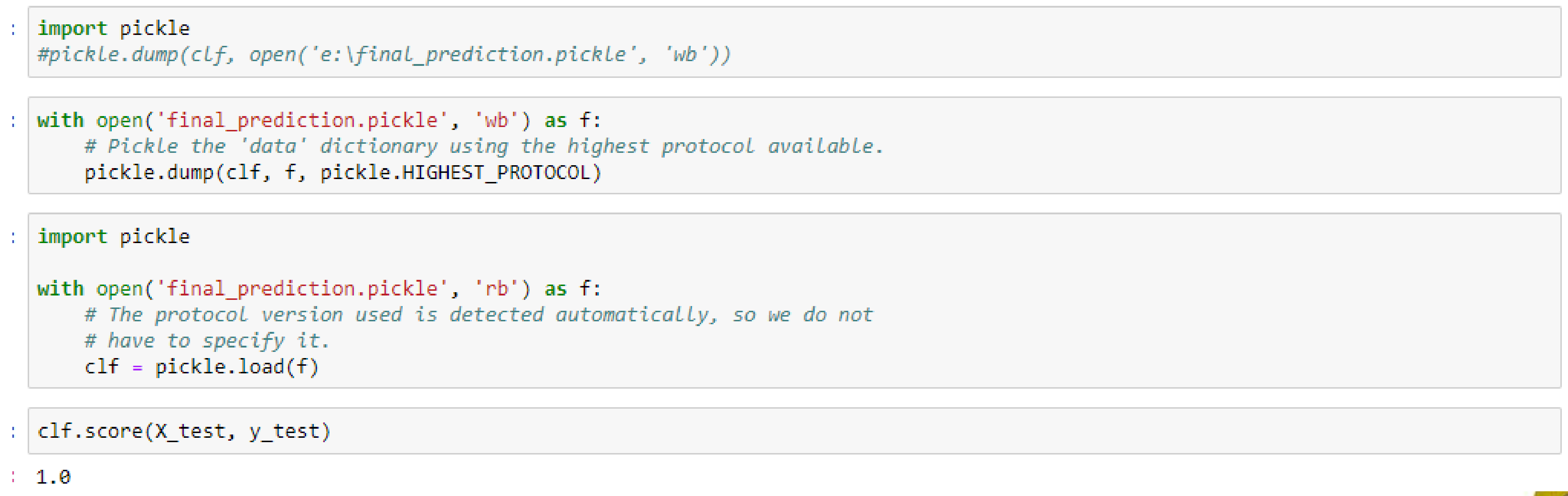
```
y_pred = clf.predict(X_test)
```

```
print("accuracy_score: %.2f"
      % accuracy_score(y_test, y_pred))
```

```
accuracy_score: 1.00
```



Use Case - Python Code



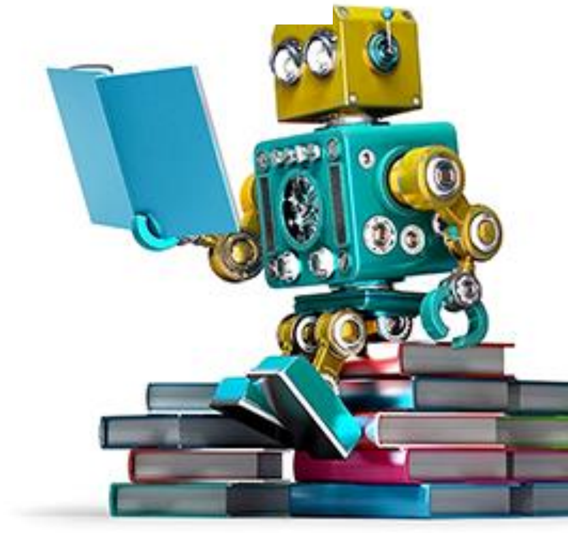
```python
import pickle
#pickle.dump(clf, open('e:\final_prediction.pickle', 'wb'))
```

```python
with open('final_prediction.pickle', 'wb') as f:
    # Pickle the 'data' dictionary using the highest protocol available.
    pickle.dump(clf, f, pickle.HIGHEST_PROTOCOL)
```

```python
import pickle

with open('final_prediction.pickle', 'rb') as f:
    # The protocol version used is detected automatically, so we do not
    # have to specify it.
    clf = pickle.load(f)
```

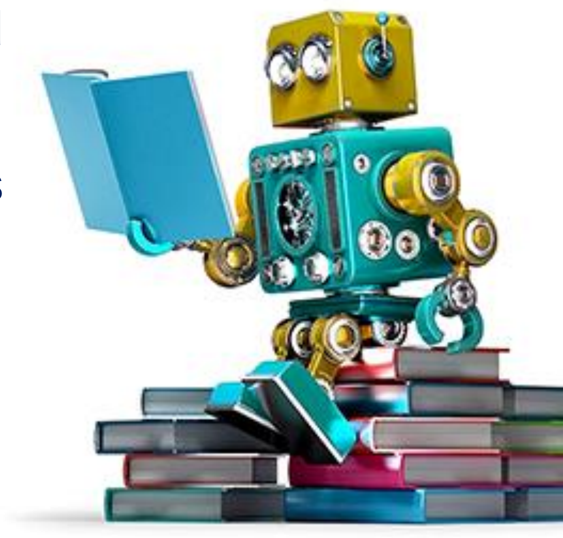
```python
clf.score(X_test, y_test)
```

```
1.0
```

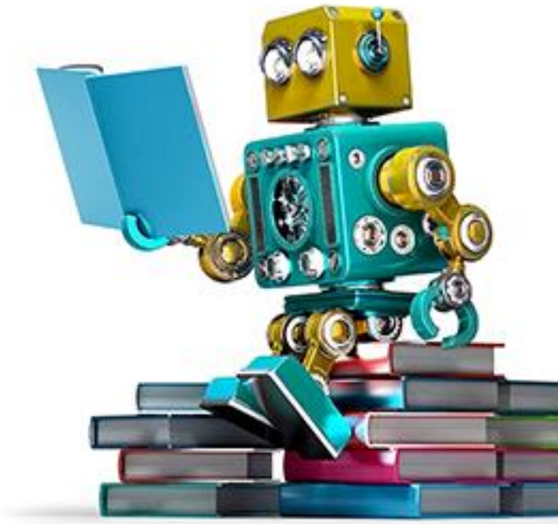


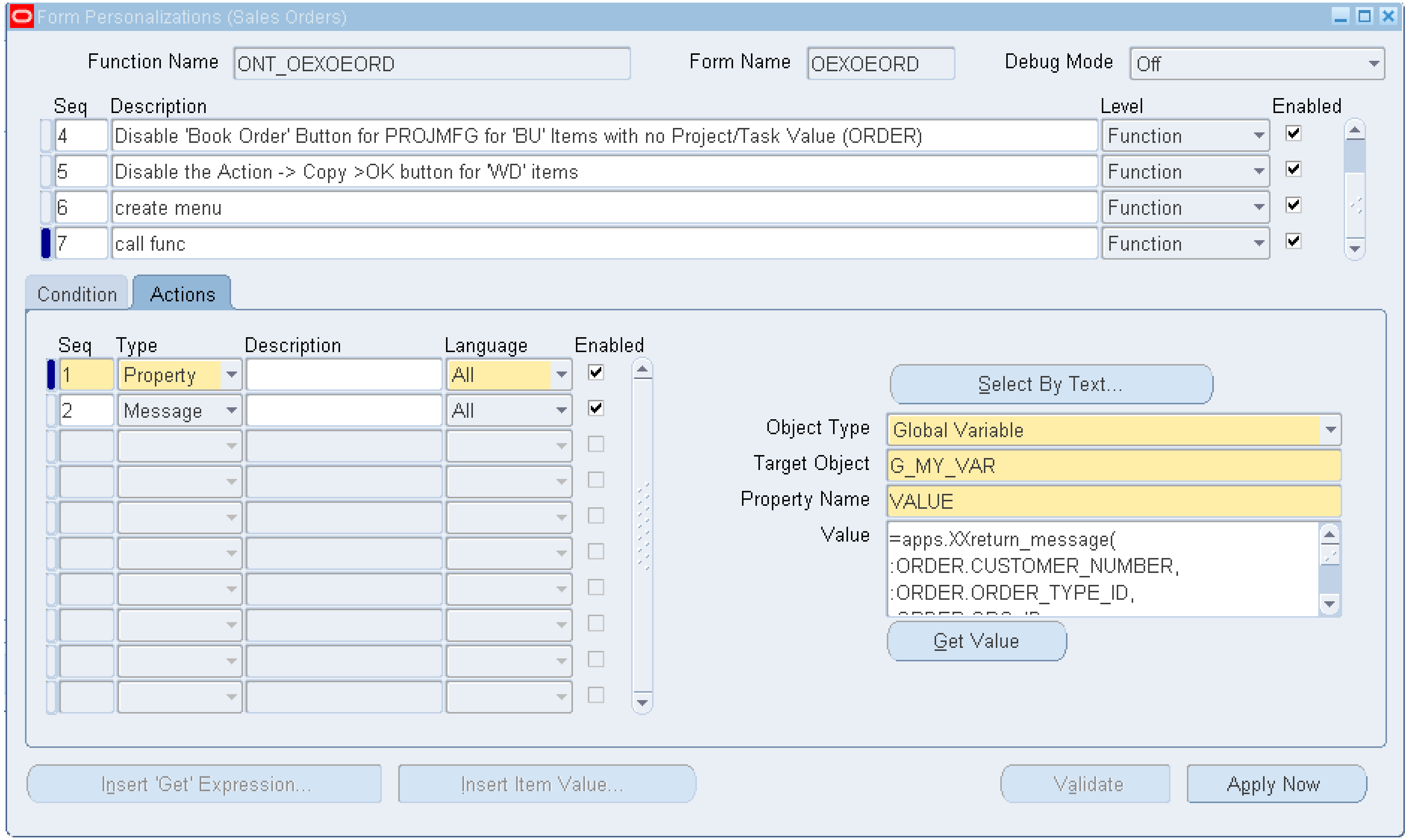
Way Forward

- ❑ Key Considerations while building the ML Model:
  - *Your model is Only as good as your Data [Quality, Quantity , Right Data Elements, etc]*
  - *Data Cleansing / Normalization is an pre-requisite to be done*
  - *Multiple Algorithms available to address a single use-case. We need to try out a multiple options & combinations to arrive at an optimal Model*
  - *Efficiency of an Model decreases over a period due to various factors. Should have the model re-evaluated/re-trained with new data set on a periodic basis*
- ❑ ML based models allows to extend to AI based solution in future
- ❑ Common ML Models can be built and exposed as an Webservice to be consumed by various applications
- ❑ Build Predictive models leveraging Industry level Benchmark data , Market trends along with EBS data



Screen Shot




Form Personalizations (Sales Orders)

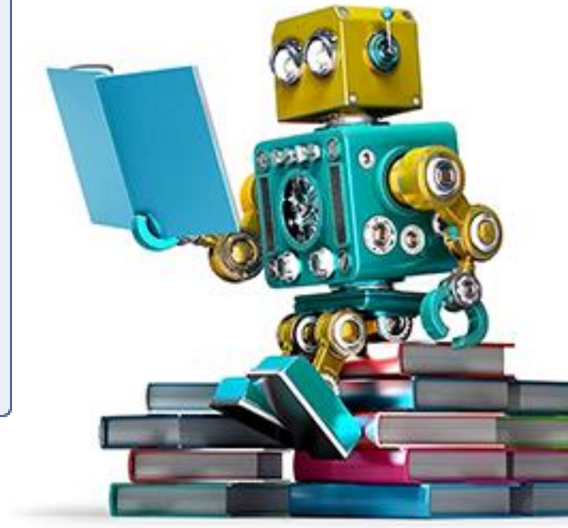
Function Name	ONT_OEXOEORD	Form Name	OEXOEORD	Debug Mode	Off

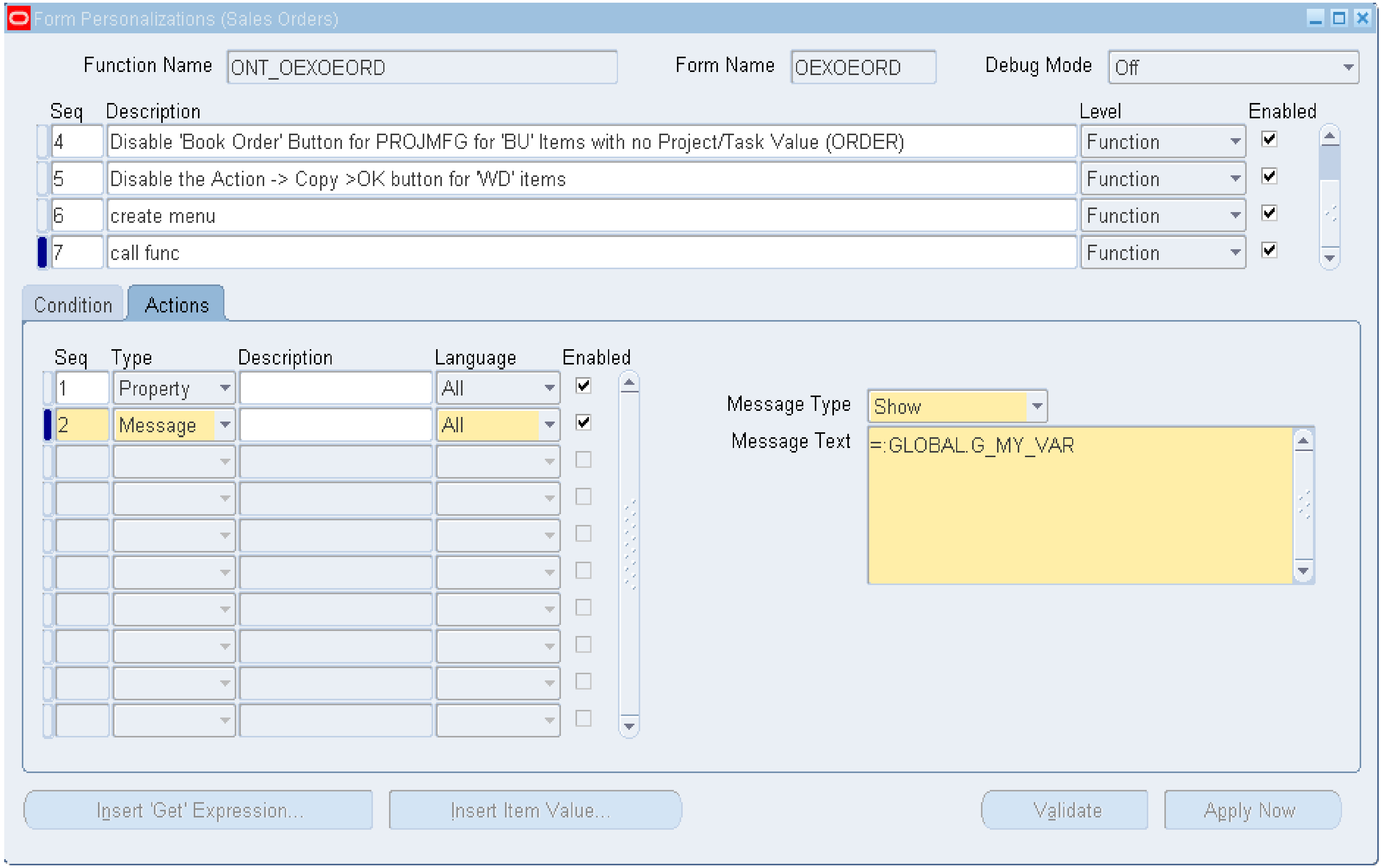
Seq	Description	Level	Enabled
4	Disable 'Book Order' Button for PROJMFG for 'BU' Items with no Project/Task Value (ORDER)	Function	✓
5	Disable the Action -> Copy >OK button for 'WD' items	Function	✓
6	create menu	Function	✓
7	call func	Function	✓

Condition | Actions

Seq	Type	Description	Language	Enabled
1	Property		All	✓
2	Message		All	✓

Object Type	Global Variable
Target Object	G_MY_VAR
Property Name	VALUE
Value	=apps.XXreturn_message(:ORDER.CUSTOMER_NUMBER, :ORDER.ORDER_TYPE_ID,




Form Personalizations (Sales Orders)

Function Name	ONT_OEXOEORD	Form Name	OEXOEORD	Debug Mode	Off

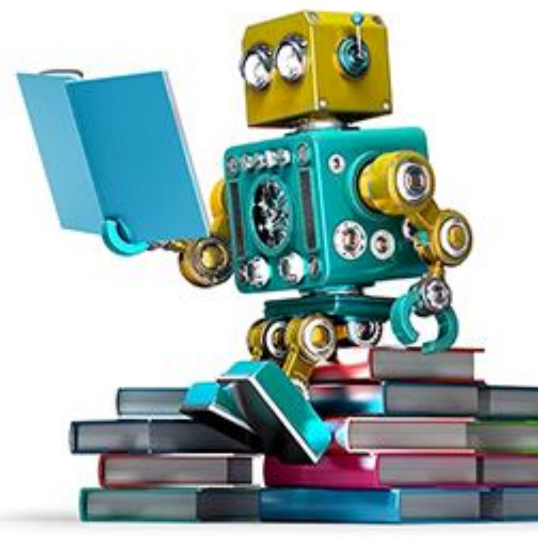
Seq	Description	Level	Enabled
4	Disable 'Book Order' Button for PROJMFG for 'BU' Items with no Project/Task Value (ORDER)	Function	✓
5	Disable the Action -> Copy >OK button for 'WD' items	Function	✓
6	create menu	Function	✓
7	call func	Function	✓

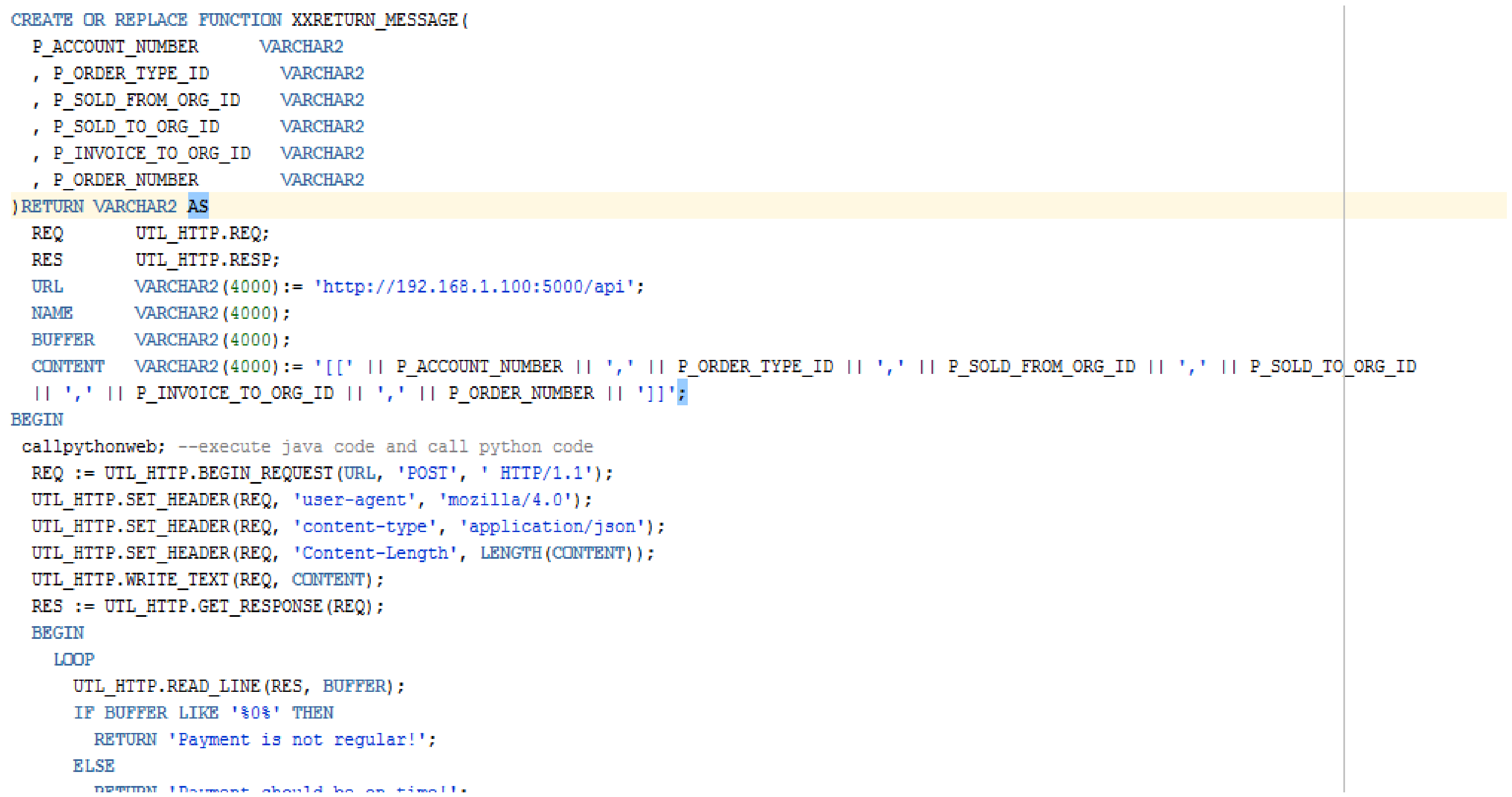
Condition | Actions

Seq	Type	Description	Language	Enabled
1	Property		All	✓
2	Message		All	✓

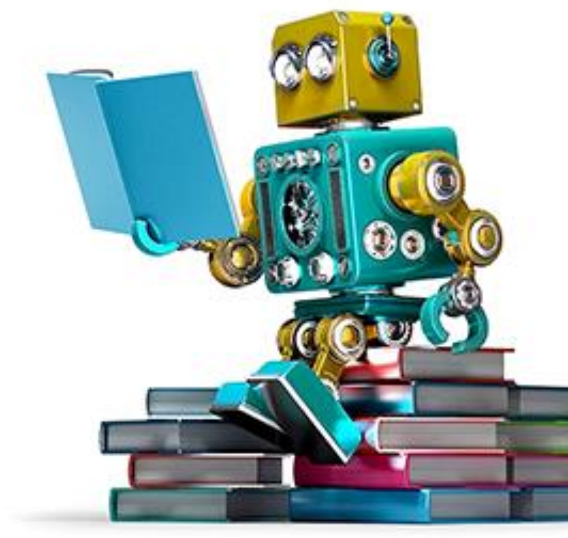
Message Type: Show

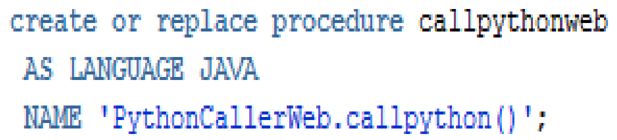
Message Text: =:GLOBAL.G_MY_VAR



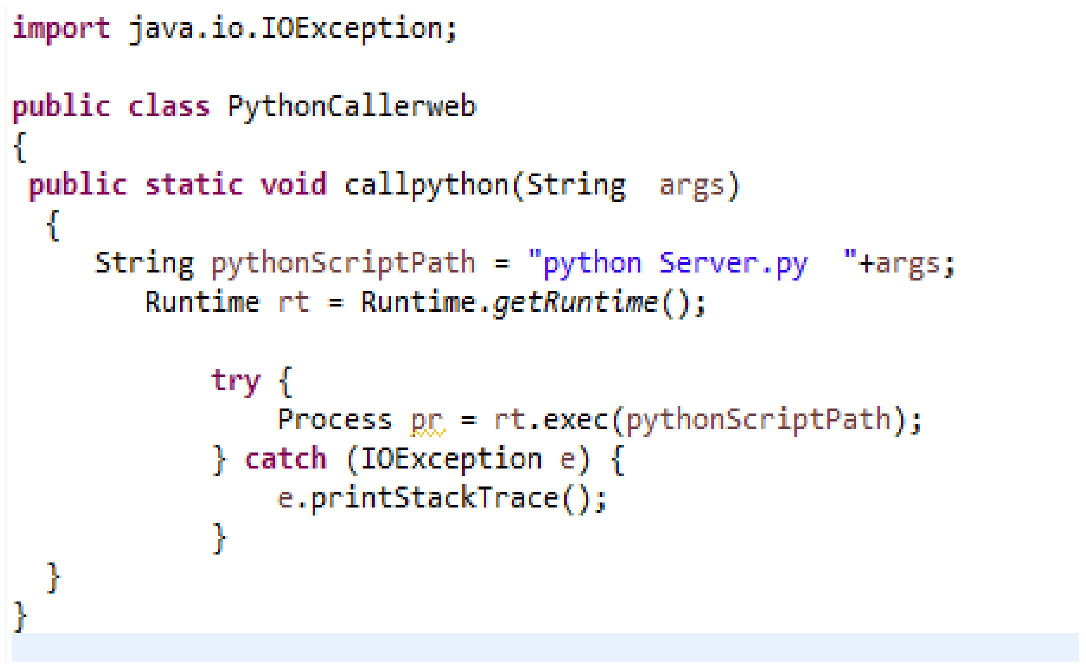


```
CREATE OR REPLACE FUNCTION XXRETURN_MESSAGE(
  P_ACCOUNT_NUMBER      VARCHAR2
  , P_ORDER_TYPE_ID       VARCHAR2
  , P_SOLD_FROM_ORG_ID    VARCHAR2
  , P_SOLD_TO_ORG_ID      VARCHAR2
  , P_INVOICE_TO_ORG_ID   VARCHAR2
  , P_ORDER_NUMBER        VARCHAR2
)RETURN VARCHAR2 AS
  REQ       UTL_HTTP.REQ;
  RES       UTL_HTTP.RESP;
  URL       VARCHAR2(4000):= 'http://192.168.1.100:5000/api';
  NAME      VARCHAR2(4000);
  BUFFER    VARCHAR2(4000);
  CONTENT   VARCHAR2(4000):= '[[' || P_ACCOUNT_NUMBER || ',' || P_ORDER_TYPE_ID || ',' || P_SOLD_FROM_ORG_ID || ',' || P_SOLD_TO_ORG_ID
  || ',' || P_INVOICE_TO_ORG_ID || ',' || P_ORDER_NUMBER || ']]';
BEGIN
 callpythonweb; --execute java code and call python code
  REQ := UTL_HTTP.BEGIN_REQUEST(URL, 'POST', ' HTTP/1.1');
  UTL_HTTP.SET_HEADER(REQ, 'user-agent', 'mozilla/4.0');
  UTL_HTTP.SET_HEADER(REQ, 'content-type', 'application/json');
  UTL_HTTP.SET_HEADER(REQ, 'Content-Length', LENGTH(CONTENT));
  UTL_HTTP.WRITE_TEXT(REQ, CONTENT);
  RES := UTL_HTTP.GET_RESPONSE(REQ);
  BEGIN
    LOOP
      UTL_HTTP.READ_LINE(RES, BUFFER);
      IF BUFFER LIKE '%0%' THEN
        RETURN 'Payment is not regular!';
      ELSE
        RETURN 'Payment should be on time!';
```

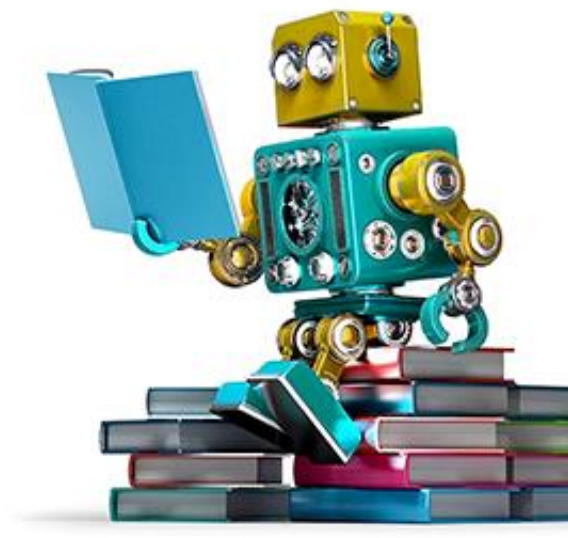


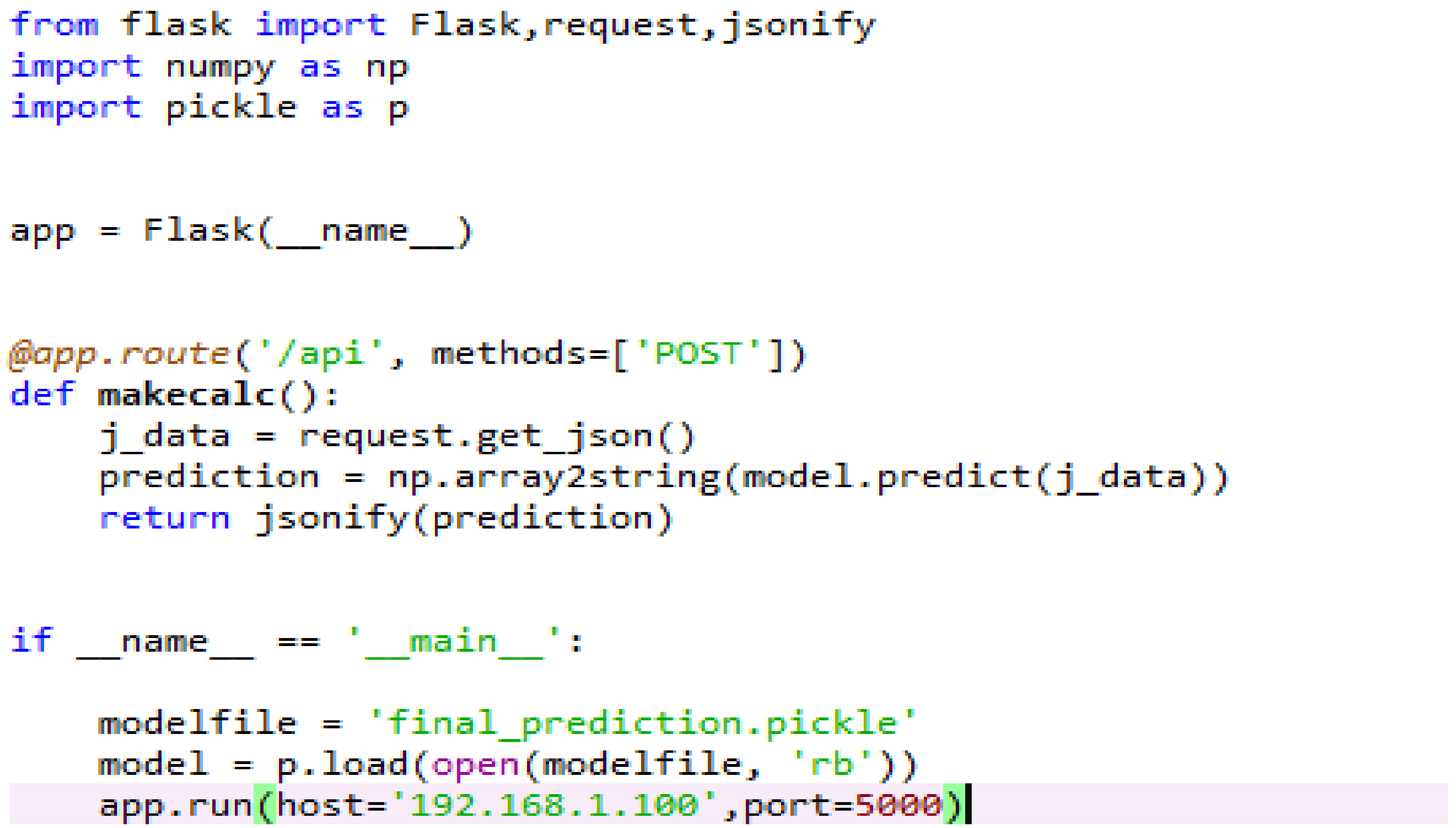


```
create or replace procedure callpythonweb
 AS LANGUAGE JAVA
 NAME 'PythonCallerWeb.callpython()';
```

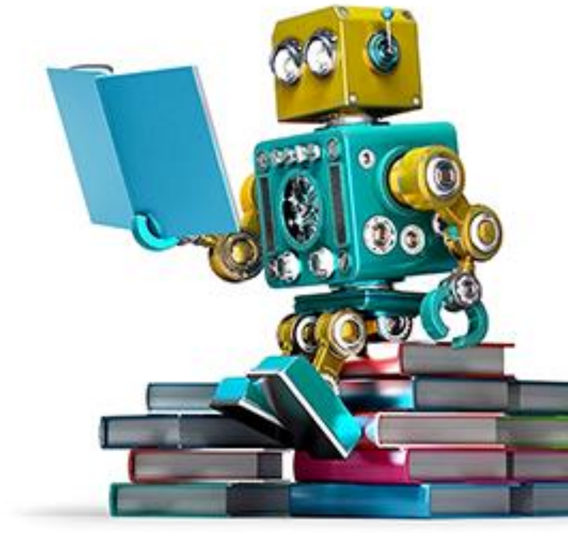


```
import java.io.IOException;

public class PythonCallerweb
{
 public static void callpython(String  args)
  {
     String pythonScriptPath = "python Server.py  "+args;
        Runtime rt = Runtime.getRuntime();

            try {
                Process pr = rt.exec(pythonScriptPath);
            } catch (IOException e) {
                e.printStackTrace();
            }
  }
}
```





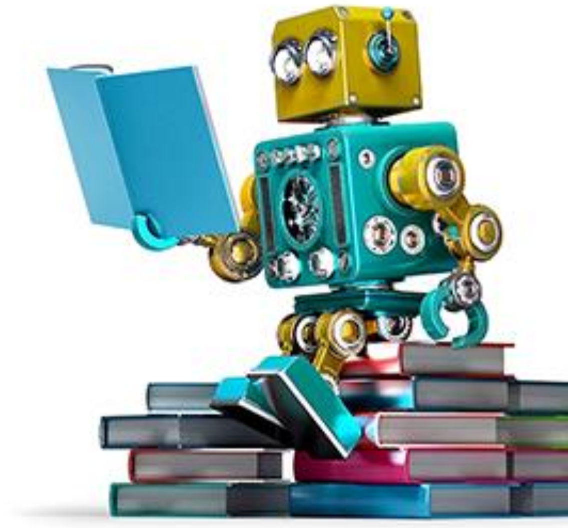
```python
from flask import Flask,request,jsonify
import numpy as np
import pickle as p


app = Flask(__name__)


@app.route('/api', methods=['POST'])
def makecalc():
    j_data = request.get_json()
    prediction = np.array2string(model.predict(j_data))
    return jsonify(prediction)


if __name__ == '__main__':

    modelfile = 'final_prediction.pickle'
    model = p.load(open(modelfile, 'rb'))
    app.run(host='192.168.1.100',port=5000)
```



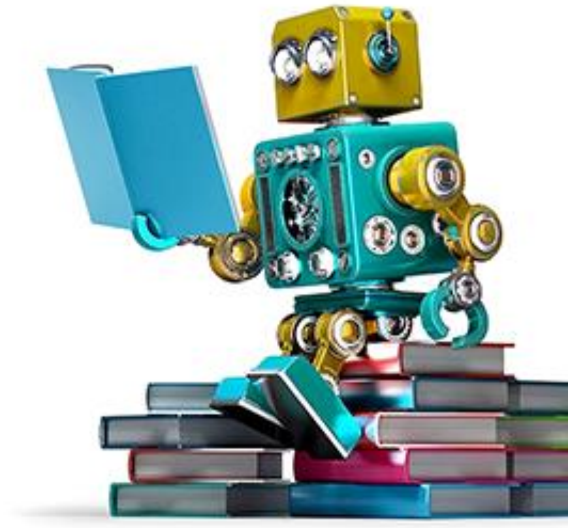
Direct Call

```
create or replace FUNCTION callpython(P_input varchar2) RETURN VARCHAR2
 AS LANGUAGE JAVA
 NAME 'PythonCaller.callpython(java.lang.String) return java.sql.String';
```

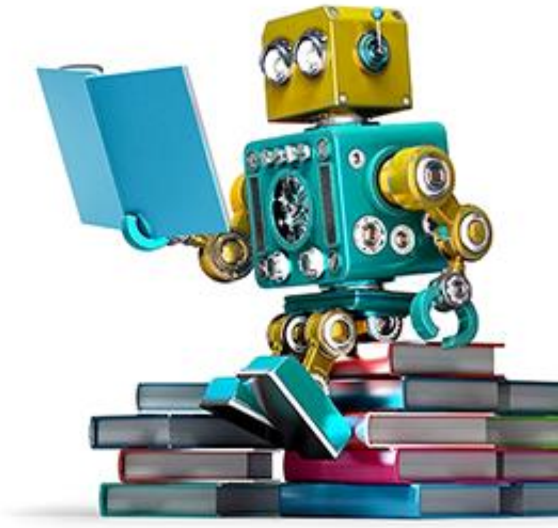
```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

public class PythonCaller
{
 public static String callpython(String  args)
  {
     String pythonScriptPath =  "python nowebservice.py   "+args;
        Runtime rt = Runtime.getRuntime();
        Process pr = null;
        try {
            pr = rt.exec(pythonScriptPath);
        } catch (IOException e) {
            e.printStackTrace(System.out);
        }
        StringBuilder everything = new StringBuilder();
        BufferedReader bfr = new BufferedReader(new InputStreamReader(pr.getInputStream()));
        String line = ""  ;
        try {
            while ((line = bfr.readLine()) != null) {
                everything.append(line);
            }
        } catch (IOException e) {
             e.printStackTrace(System.out);
        }
        return everything.toString();
  }
 }
```

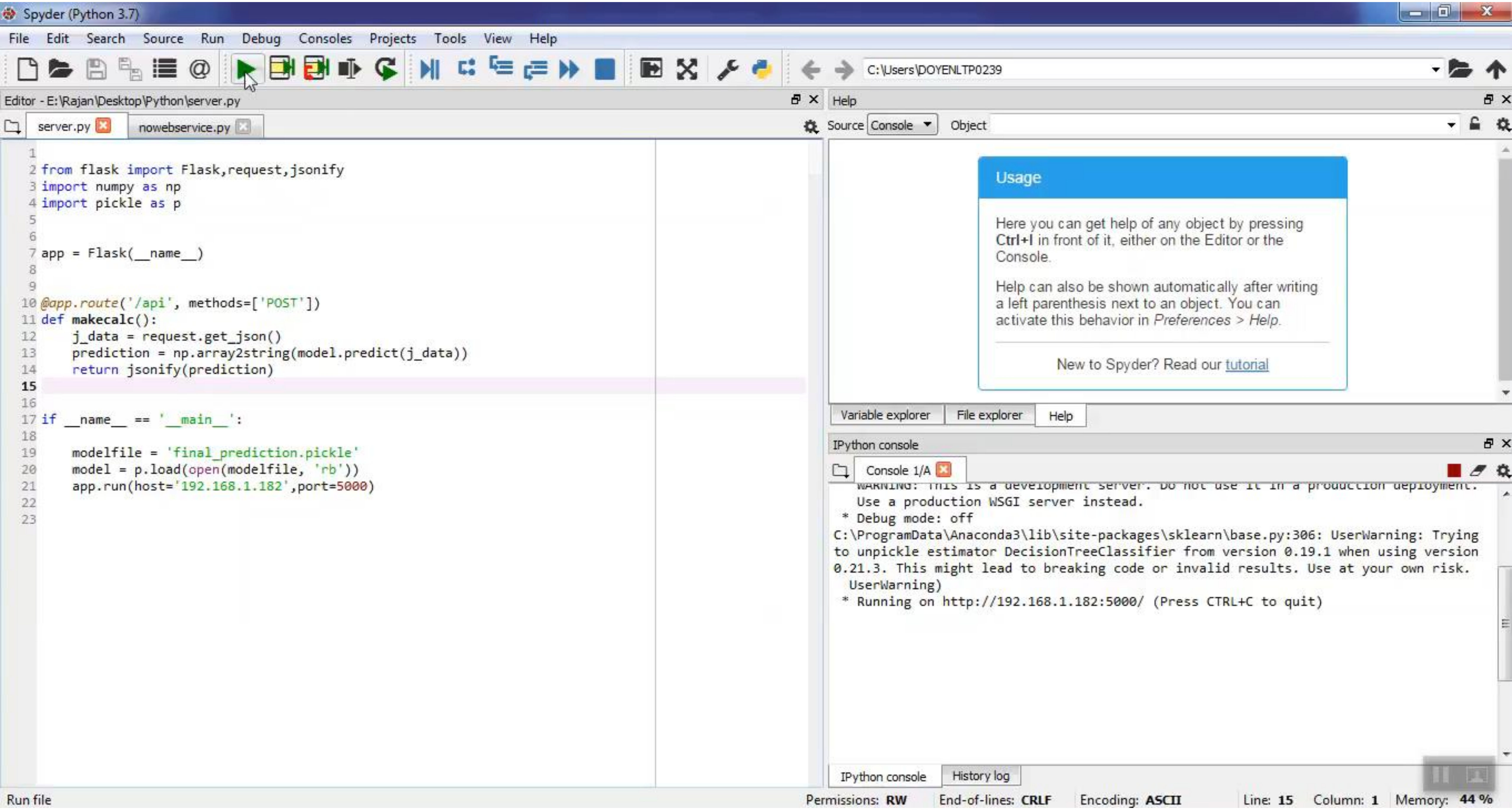


```python
from flask import jsonify
import sys
import numpy as np
import pickle as p
def makecalc(j_data):
    prediction = np.array2string(model.predict(j_data))
    return jsonify(prediction)

if __name__ == '__main__':

    modelfile = 'final_prediction.pickle'
    p.dumps(modelfile,2)
    model = p.load(open(modelfile, 'rb'))
    a =  sys.argv[0,1,2,3,4,5]
    makecalc(a)
```



Demo





Spyder (Python 3.7)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\DOYENLTP0239
Editor - E:\Rajan\Desktop\Python\server.py
server.py
nowebservice.py
Help
Source Console
Object

```python

from flask import Flask,request,jsonify
import numpy as np
import pickle as p


app = Flask(__name__)


@app.route('/api', methods=['POST'])
def makecalc():
    j_data = request.get_json()
    prediction = np.array2string(model.predict(j_data))
    return jsonify(prediction)


if __name__ == '__main__':

    modelfile = 'final_prediction.pickle'
    model = p.load(open(modelfile, 'rb'))
    app.run(host='192.168.1.182',port=5000)


```

Usage

Here you can get help of any object by pressing **Ctrl+I** in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in *Preferences > Help*.

New to Spyder? Read our tutorial

```
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\base.py:306: UserWarning: Trying
to unpickle estimator DecisionTreeClassifier from version 0.19.1 when using version
0.21.3. This might lead to breaking code or invalid results. Use at your own risk.
  UserWarning)
 * Running on http://192.168.1.182:5000/ (Press CTRL+C to quit)
```

Permissions: RW End-of-lines: CRLF Encoding: ASCII Line: 15 Column: 1 Memory: 44 %

Thank You