Python Tools of IBM Db2 for Implementing AI Systems

Shaikh Quader

AI Architect, IBM Db2

May 15, 2024

Prepared for:

CCDUG 2024

Agenda

- Introducing Key ML Concepts
- The role of RDBMS and Open-source on ML
- Key Challenges to Enterprise ML
- IBM Db2’s Support for Open-source and Enterprise ML
- Demo: Creating and Deploying a Python ML Pipeline on Db2
- Questions

Machine Learning in Action

Machine Learning across the Industries

Finance and Banking

- Risk analysis
- Credit scoring
- Client analysis
- Fraud detection
- Trading exchange forcasting

Retail and e-Commerce

- Fraud detection
- Price optimization
- Recommendations
- Demand forecasting
- Customer segmentation

Sales and Marketing

- Price optimization
- Churn rate analysis
- Upsell opportunity analysis
- Customer lifetime value prediction
- Market and customer segmentation
- Sentiment analysis in social networks

Travel and Booking

- Price optimization
- Demand forecasting
- Price forecasting (*for dynamically changing prices - eg. fuel costs*)

Healthcare and Life Sciences

- Identifying at-risk patients
- Increase diagnostic accuracy
- Insurance product cost optimization

Other

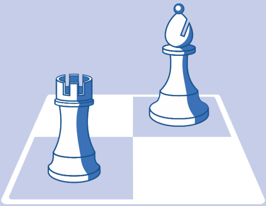
- Object recognition (*photo, video, etc.*)
- On-line dating personality matching
- Content recommendations (*music, movies, articles, news, etc.*)

Artificial Intelligence, Machine Learning, and Deep Learning

Artificial Intelligence (AI)

Human intelligence exhibited by machines

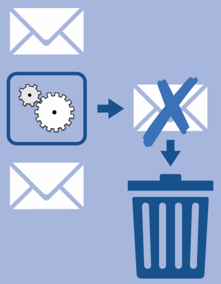
- Reasoning
- Natural Language Processing (NLP)
- Planning



Machine Learning (ML)

An approach to achieve AI

- Gradient Boosting Machine (GBM)
- Support Vector Machine (SVM)
- Logistic Regression
- Factorization Machines (FM)
- Field-aware Factorization Machines (FFM)



Deep Learning (DL)

A technique for implementing ML

- Deep Neural Networks
- Deep Belief Networks
- Recurrent Neural Networks

What is Machine Learning?

"Machine learning is an approach to (1) learn (2) complex patterns from (3) existing data and use these patterns to make (4) predictions on (5) unseen data." [1]

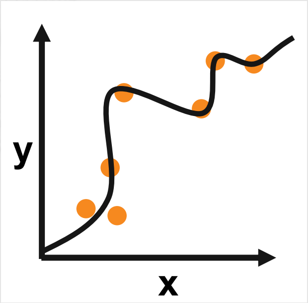
[1] [Huyen, Chip. *Designing Machine Learning Systems*. O'Reilly Media, Incorporated, 2022.]

Common Machine Learning Techniques

Regression (supervised learning)

- Predict a quantity/continuous value
- Linear regression, polynomial regression, …

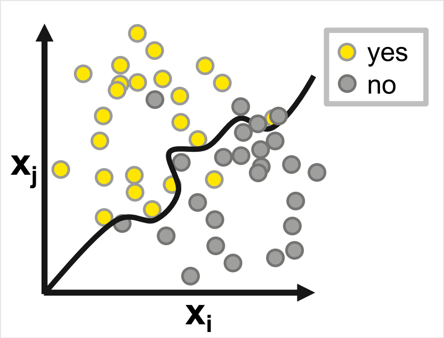
What will my future sales look like?



Classification (supervised learning)

- Predict a label/category
- Binary or multiclass

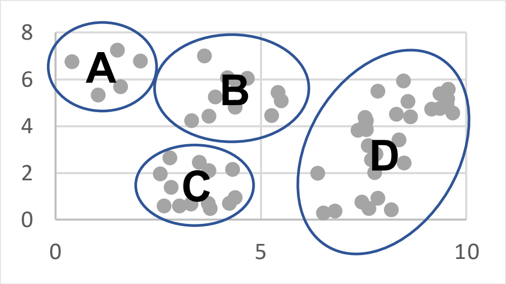
Will this client default on their loan?



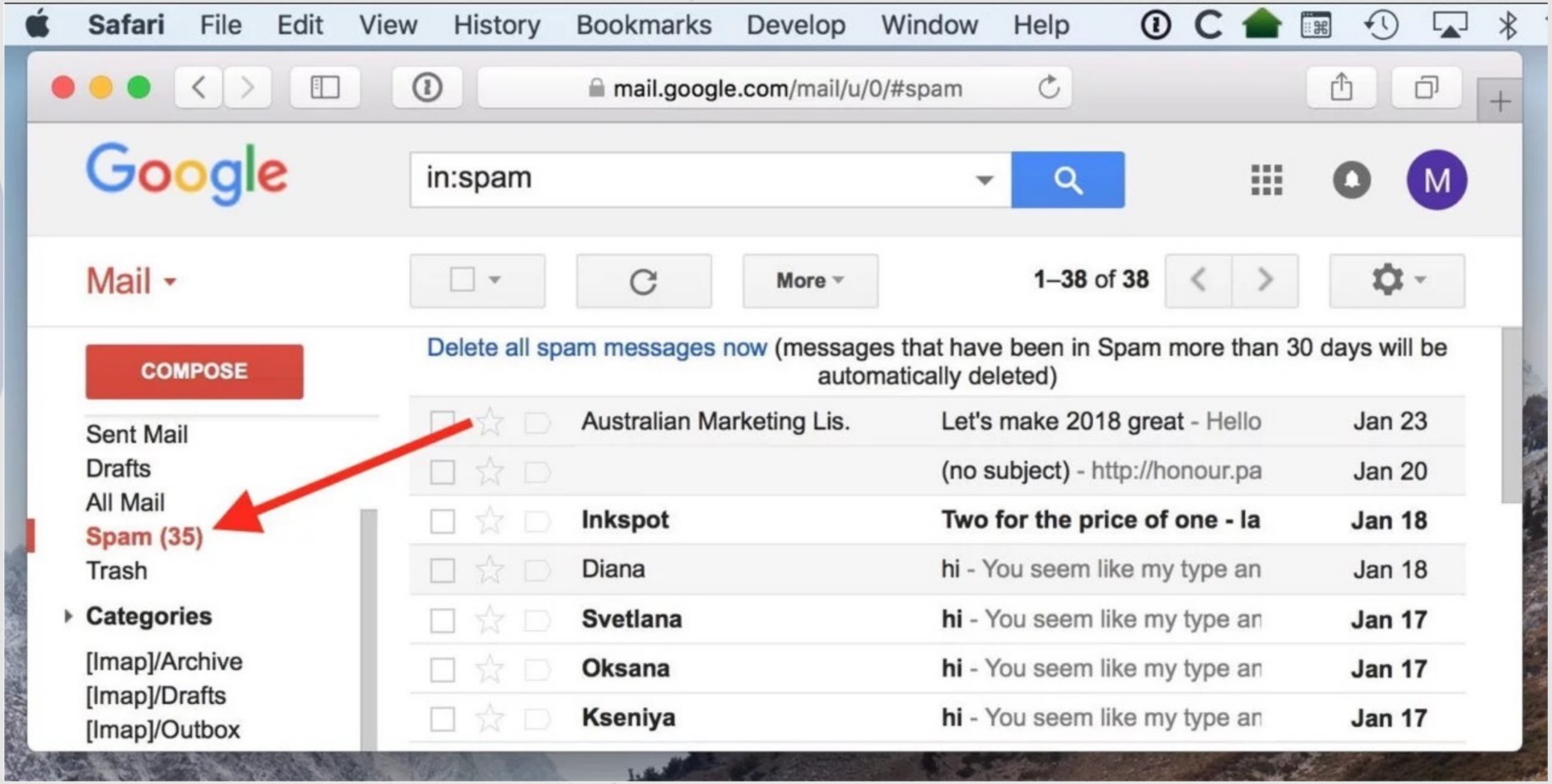
Clustering (self-supervised learning)

- Group similar objects together

Are there similarities among subgroups of customers at my company?



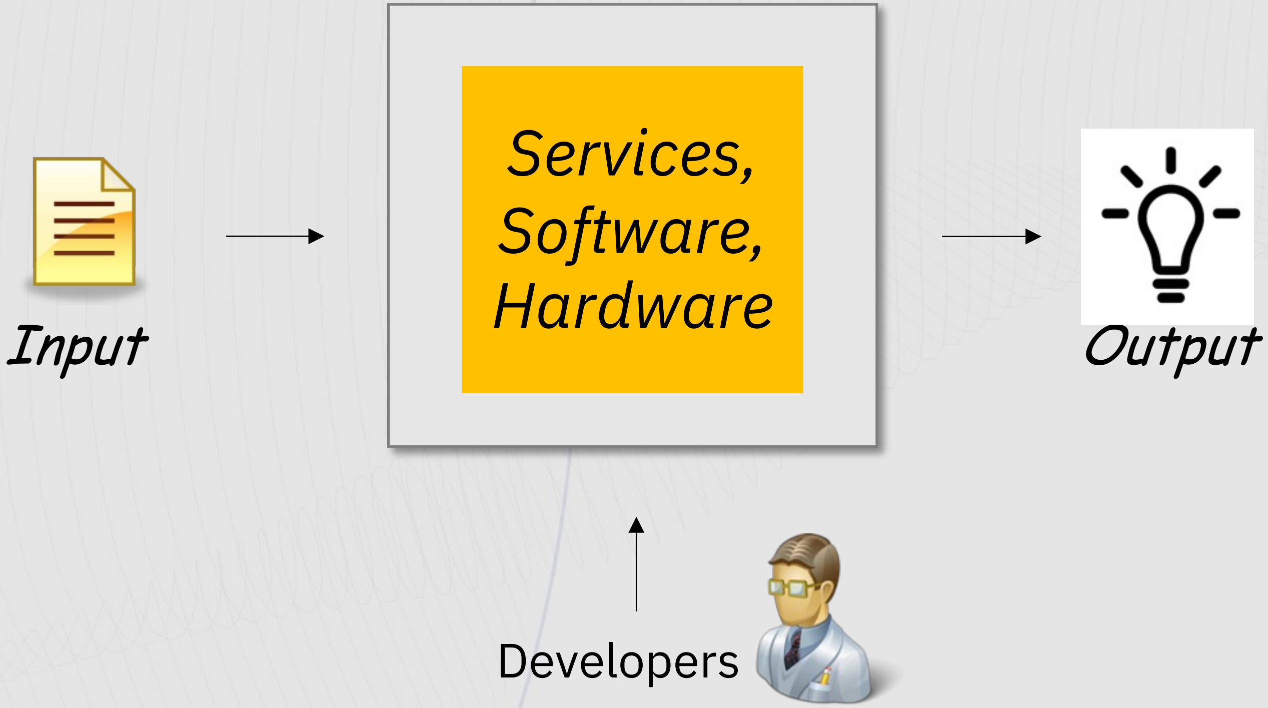
Is an email a spam or a ham? (Classification ML)



What is the right purchase price of a house? (Regression ML)

Quiz: predicting salary raise – what kind of ML is this?

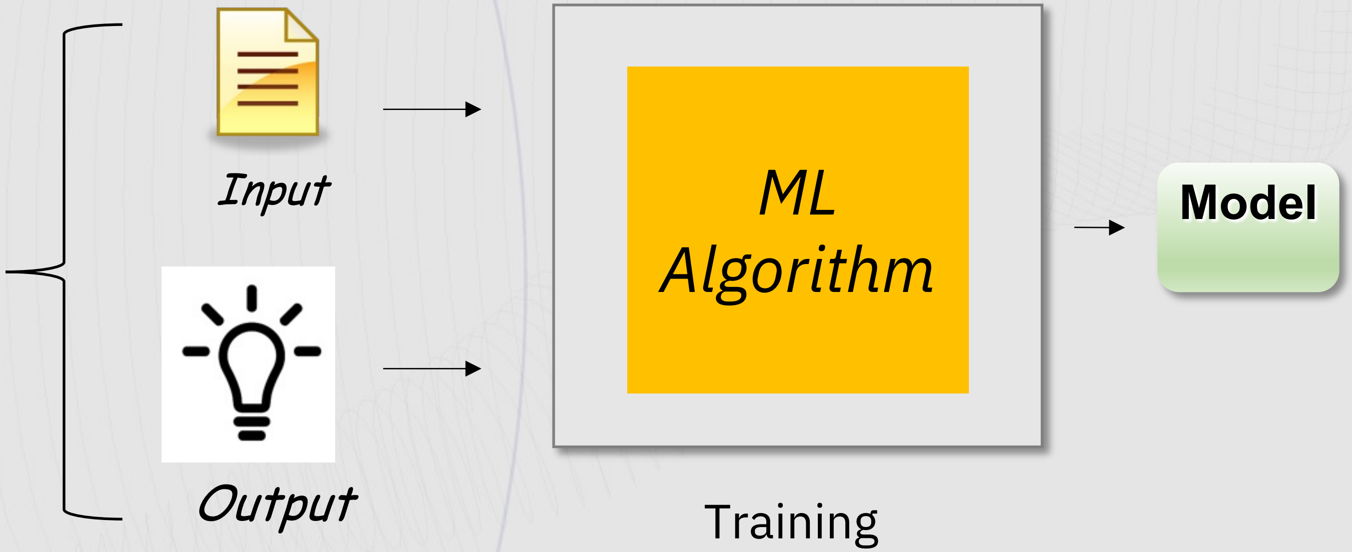
How does ML work?



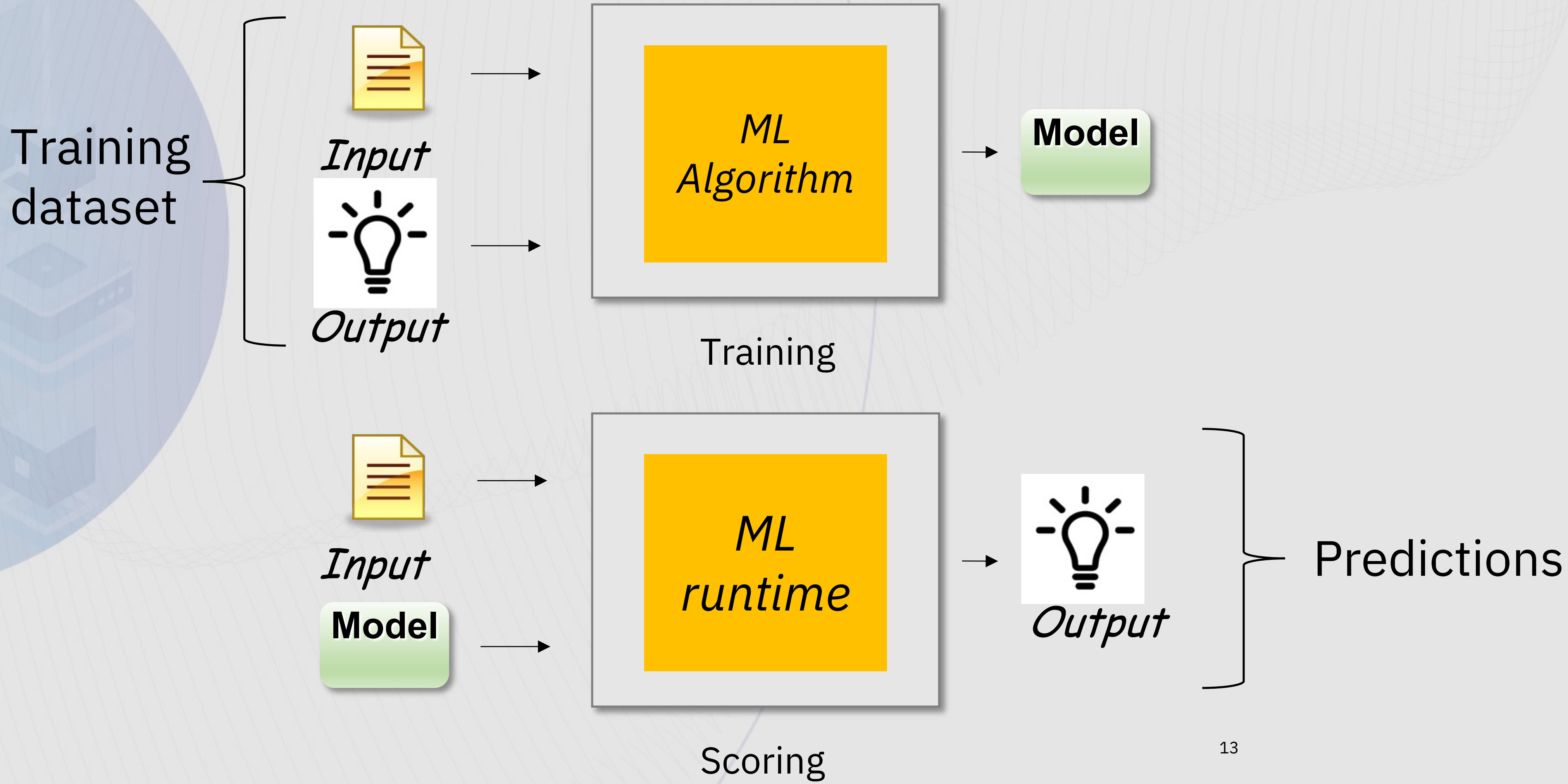
Classical Systems Development

How does ML work? [2]

Training Examples



How does ML work? [3]



Lifecyle Phases of an Enterprise ML System

Research

Experiment

Train

Production

Deploy

Inference

Finding the best ML pipeline

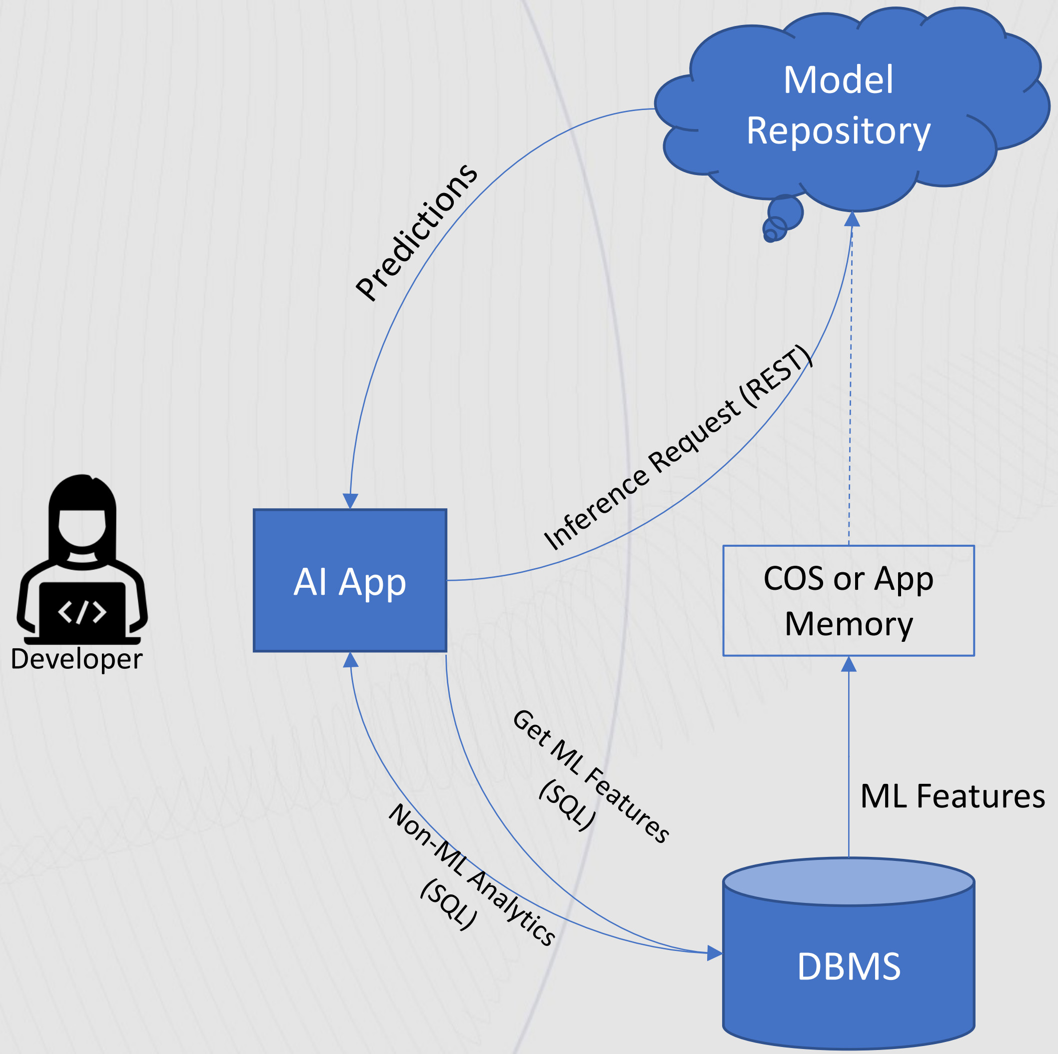
Training a model using the best pipeline

Making the trained model available for use

Generating predictions from the model

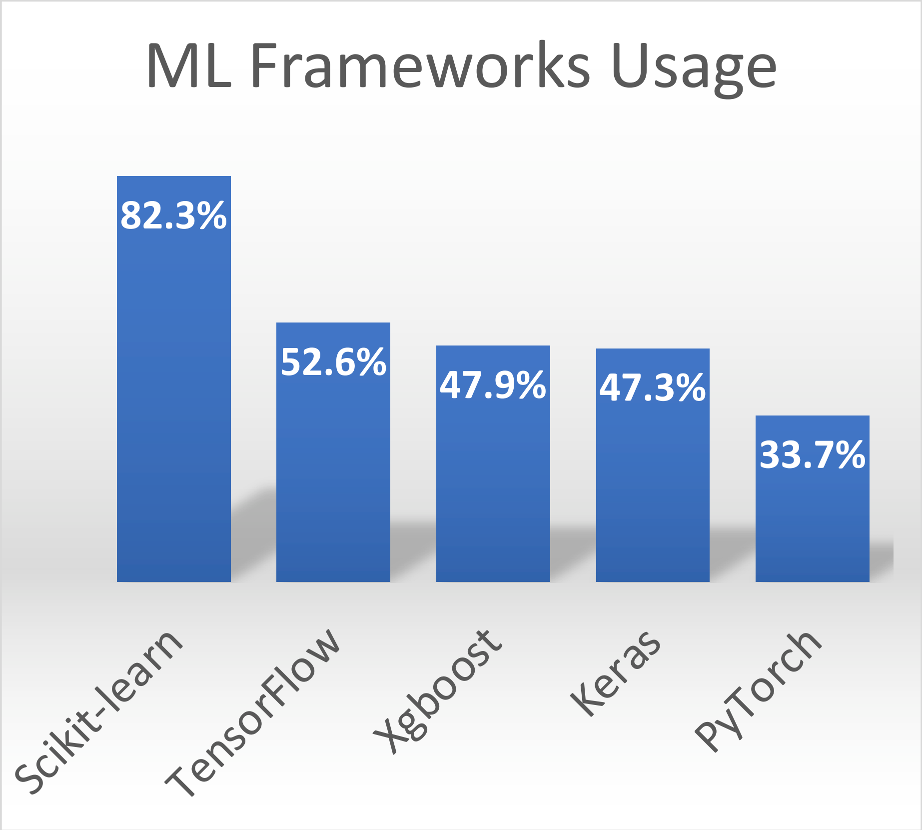
Data – storage, regulations, scale, quality
Model – infrastructure, compute resources, latency, integration

Integrating ML with Apps is both complex and slow



- ML Features
- Non-ML Analytics

Python-based ML Frameworks Are Most Popular



[2021 Kaggle Data Science Survey]

66%

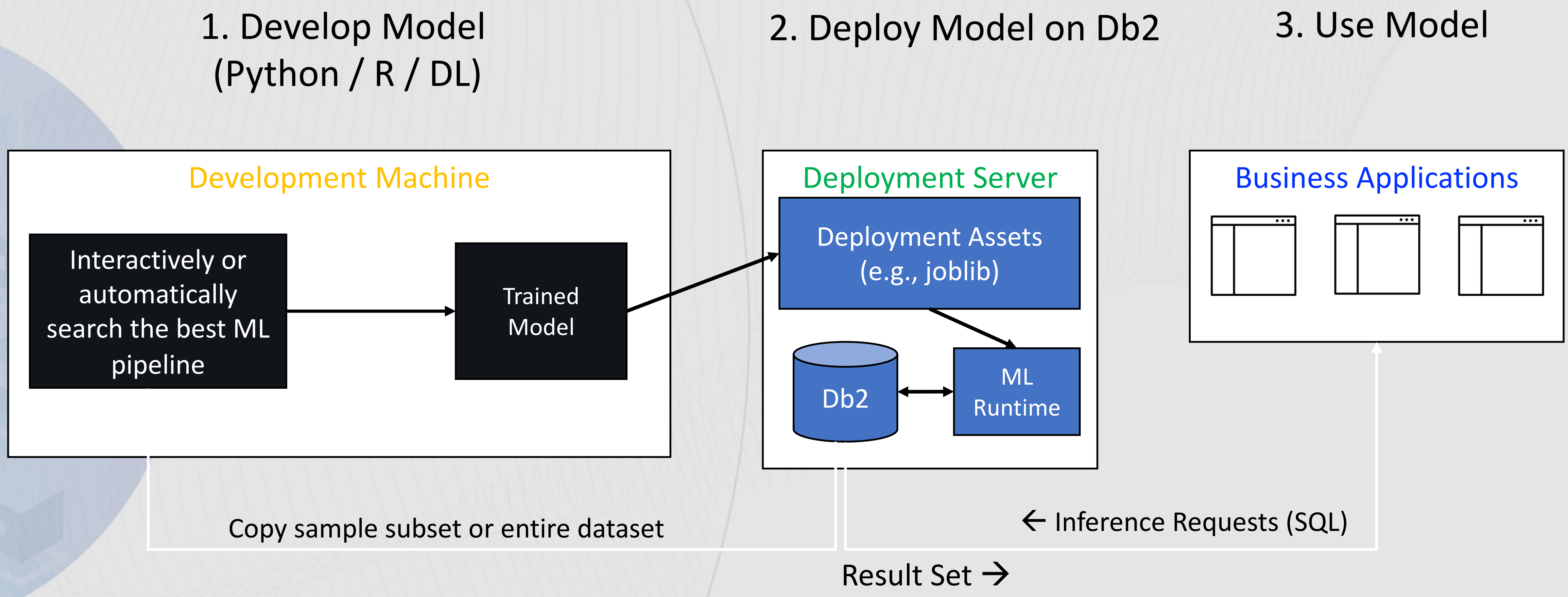
of enterprises rely on relational data for their ML models.

Source: The State of Data Science & Machine Learning 2017, Kaggle, October 2017
(based on 2017 Kaggle survey of 16,000 ML practitioners)

Open-Source Machine Learning with IBM Db2

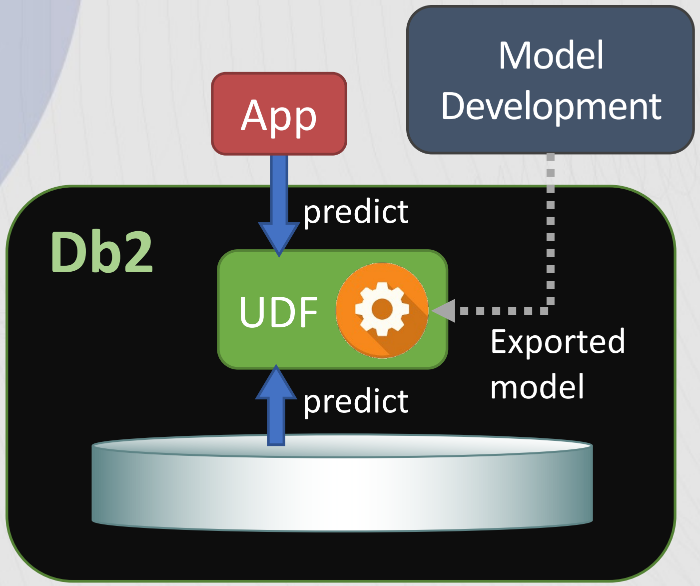
Data Access	Data Exploration	Data Cleansing	Model Development	Model Deployment
		Transformations *Missing Values* *Normalization*		
Python & R Drivers REST API Db2 Magic Commands	In-Db2 ML Stored Procedures		With any open-source and proprietary Python and R IDEs, including out-of-box integration w/ Watson Studio	Python UDF, R UDF

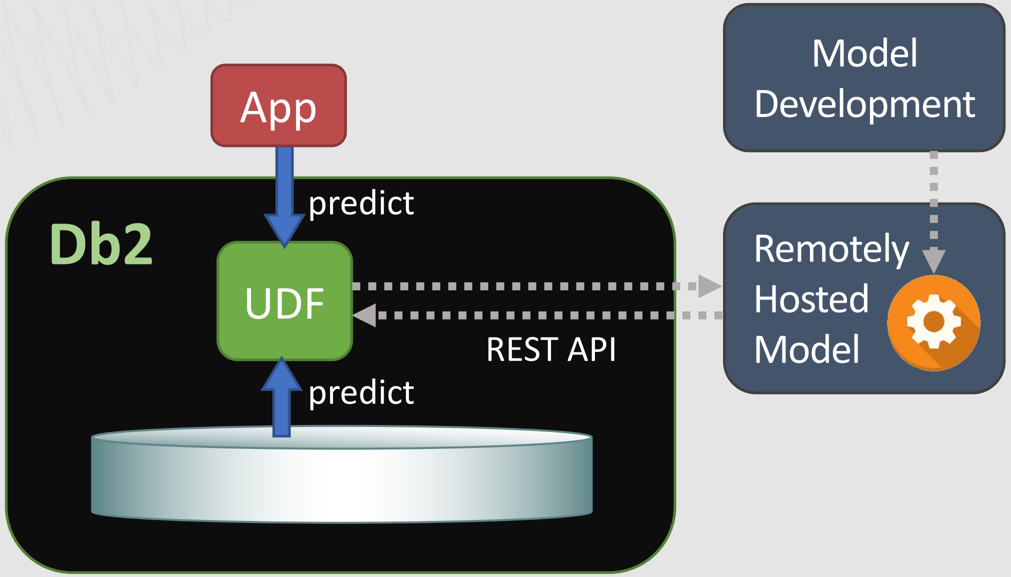
Open-source Models Integration Support in Db2



Benefits of Machine Learning with Python UDFs

- Train in your model development environment of choice
  - Training can be accelerated via GPUs
- Allows for transformations and algorithms not found natively in Db2
- Low-latency predictions, inferencing where the data lives
- Predict (in-DB, call from app) via SQL
- Incorporate into data entry workflow through triggers (insert and score)



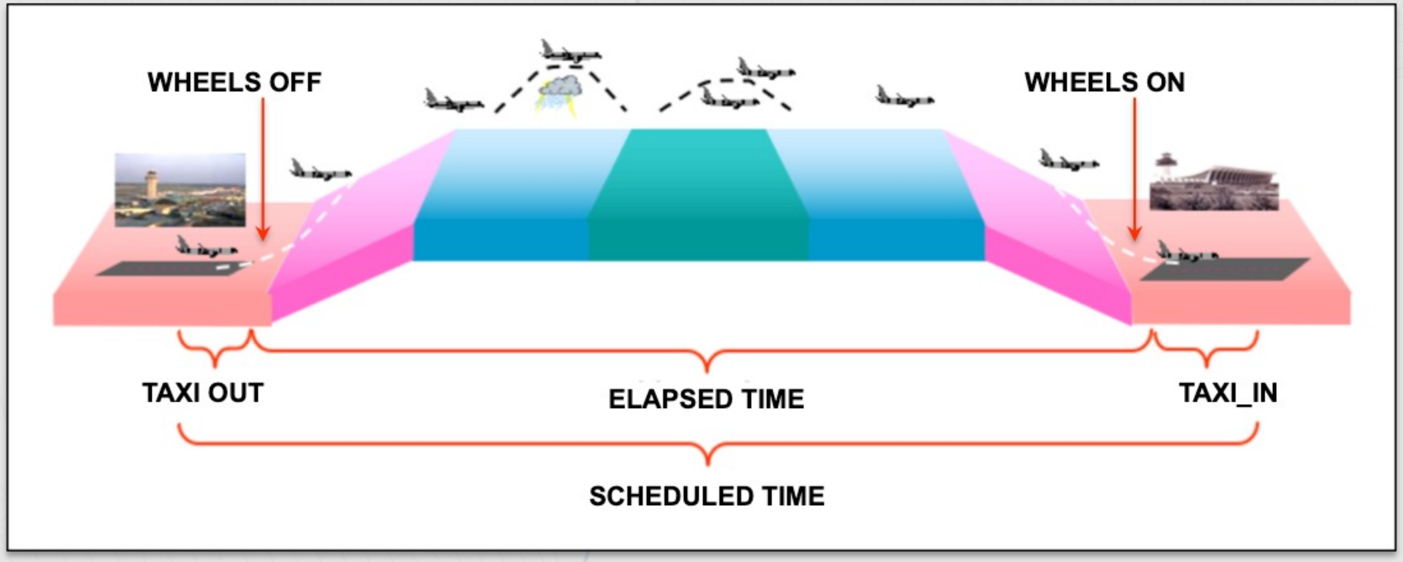


Python UDFs – The Basics

- Partially documented in the Db2 documentation; more information (e.g. PYTHON_PATH) in the Db2 Warehouse documentation)
- Linux-only - X86 and PPCLE
- Python 3.6+ works
- Function types: Scalar (UDSF), Table (UDTF), Aggregate (UDAF)
- UDF must be defined as `fenced`
- DPF/pureScale: Copy Python source file (.py) to same location on each node
- Set `PYTHON_PATH` to your Python runtime <u>executable</u>
  - e.g. `db2 update dbm cfg using python_path /usr/bin/python3`
  - Python executable must be accessible by the fenced user ID

Demo: Creating a Scikit-Learn (Python) Model and Deploying the Model on IBM Db2

ML Use Case: Predicting If a Flight will Arrive On Time



Demo Steps

1. Set up Python Notebook and Db2 Connection
2. Load ML Data from Db2 into Python Runtime
3. Exploring the Data Set – Finding Issues
4. Creating a ML Pipeline
5. Defining a Python UDF for ML Inferencing
6. Registering the Python UDF on Db2
7. Creating a Db2 Table to Store Predictions
8. Retrieving Predictions from Db2

Imports and Db2 Connection

Import Python Packages:

```
import pandas as pd
from sklearn.compose import ColumnTransformer, make_column_selector, make_column_transformer
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import MaxAbsScaler
from sklearn.preprocessing import OneHotEncoder
from sklearn.impute import SimpleImputer
import numpy as np
from joblib import dump
from sklearn.linear_model import LogisticRegression
```

Load Db2 Magic Commands:

```
!wget https://raw.githubusercontent.com/IBM/db2-jupyter/master/db2.ipynb
```

```
%run db2.ipynb
```

Connect to Db2 database:

```
%sql CONNECT TO db2sample USER user1 USING password HOST mydb2host.mycompany.com PORT 50000
```

Bringing ML Training Dataset from Db2

SQL Query:

```
query = %sql SELECT * FROM IDUG.FLIGHTS2
df = pd.DataFrame(query)
```

(Rows, Columns):

```
df.shape
```

```
(1000000, 19)
```

Sample Rows with a Subset of Columns

```
df[cols_show].sample(5)
```

	MONTH	DAYOFWEEK	UNIQUECARRIER	ORIGIN	DEST	DEPDELAY	TAXIOUT	WHEELSOFF	FLIGHTSTATUS
923730	3	1	YV	PHX	PSP	0.0	26	2031	1
47590	7	4	FL	TPA	ATL	-7.0	9	627	0
35844	4	6	DL	LAX	JFK	-5.0	24	1259	0
141217	3	1	WN	LAS	OMA	10.0	11	1526	0
132676	1	2	WN	DAL	LBB	6.0	8	2024	1

Checking For Missing Values

```
df.isnull().sum()
```

```
YEAR              0
QUARTER           0
MONTH             0
DAYOFMONTH        0
DAYOFWEEK         0
UNIQUECARRIER     0
ORIGIN            0
DEST              0
CRSDEPTIME        0
DEPTIME           0
DEPDELAY         63
DEPDEL15         63
TAXIOUT           0
WHEELSOFF         0
CRSARRTIME        0
CRSELAPSEDTIME    0
AIRTIME           0
DISTANCEGROUP     0
FLIGHTSTATUS      0
```

Data Preprocessing To-Dos:

- [ ] 1. Replace missing values in DEPDELAY and DEPDELAY15 columns

Checking Column Types

```
df.dtypes
```

```
YEAR               int64
QUARTER            int64
MONTH              int64
DAYOFMONTH         int64
DAYOFWEEK          int64
UNIQUECARRIER     object
ORIGIN            object
DEST              object
CRSDEPTIME         int64
DEPTIME            int64
DEPDELAY         float64
DEPDEL15         float64
TAXIOUT            int64
WHEELSOFF          int64
CRSARRTIME         int64
CRSELAPSEDTIME     int64
AIRTIME            int64
DISTANCEGROUP      int64
FLIGHTSTATUS       int64
```

Data Preprocessing To-Dos:

- ❑ 1. Replace missing values in DEPDELAY and DEPDELAY15 columns
- ❑ 2. Convert object (string) columns into numeric

Checking Value Range in the Numeric Cols

```
df.select_dtypes(include=np.number).describe().iloc[[3, 7]].transpose()
```

	min	max
YEAR	2009.0	2018.0
QUARTER	1.0	4.0
MONTH	1.0	12.0
DAYOFMONTH	1.0	31.0
DAYOFWEEK	1.0	7.0
CRSDEPTIME	1.0	2359.0
DEPTIME	1.0	2400.0
DEPDELAY	-62.0	1767.0
DEPDEL15	0.0	1.0
TAXIOUT	1.0	232.0
WHEELSOFF	1.0	2400.0
CRSARRTIME	1.0	2400.0
CRSELAPSEDTIME	18.0	700.0
AIRTIME	7.0	703.0
DISTANCEGROUP	1.0	11.0
FLIGHTSTATUS	0.0	1.0

Data Preprocessing To-Dos:

- ❑ 1. Replace missing values in DEPDELAY and DEPDELAY15 columns
- ❑ 2. Convert categorical (object) columns into numeric
- ❑ 3. Scale numeric values into the same range

Creating a Data Preprocessing Pipeline

Data Preprocessing:

- ✓ 1. Replace missing values in DEPDELAY and DEPDELAY15 columns
- ✓ 2. Convert categorical (object) columns into numeric
- ✓ 3. Scale numeric values into the same range

```
num_pipeline = make_pipeline(SimpleImputer(strategy='constant', fill_value=0),
                             MaxAbsScaler())
cat_pipeline = make_pipeline(SimpleImputer(strategy='most_frequent'),
                             OneHotEncoder(handle_unknown='ignore'))

preprocessing = make_column_transformer(
    (num_pipeline, make_column_selector(dtype_include=np.number)),
    (cat_pipeline, make_column_selector(dtype_include=np.object))
)
```

Creating a ML Classification Pipeline using Scikit-Learn

Define the ML Pipeline

Train a ML Model

```
pipe_lr = make_pipeline(preprocessing,
                        LogisticRegression(random_state=1,
                                           solver='lbfgs'))
pipe_lr.fit(X, y)
```

Evaluate the Trained Model

```
predictions = pipe_lr.predict(X_test)

pipe_lr.score(X_test, y_test) * 100
```

```
89.4365
```

Export the Trained ML Pipeline

```
dump(pipe_lr, 'pipe_lr.joblib')
```

```python
%%writefile myUDF.py

#Imports
import nzae
import pandas as pd
from joblib import load
import numpy as np
import sklearn

class full_pipeline(nzae.Ae):
    def _runUdtf(self):
        # Define static variables
        input_cols = ['YEAR','QUARTER', 'MONTH', 'DAYOFMONTH', 'DAYOFWEEK','UNIQUECARRIER',
                      'ORIGIN', 'DEST', 'CRSDEPTIME', 'DEPDELAY', 'DEPDEL15','TAXIOUT','WHEELSOFF',
                      'CRSARRTIME', 'CRSELAPSEDTIME', 'AIRTIME', 'DISTANCEGROUP']

        # Load deployment assets
        pipe_lr = load('/home/db2inst1/pipe_lr.joblib')

        # Collect rows into a single batch
        batchsize = 10000
        rownum = 0
        row_list = []
        for row in self:
            row_list.append(row)
            rownum = rownum+1

            if rownum==batchsize:
                ########################
                ### BATCHING ###
                ########################

                # Collect the rows into a dataframe
                df = pd.DataFrame(row_list, columns=input_cols)

                # Save the original datestring (yyyymmdd), ORIGIN, DEST, UNIQUECARRIER, CRSDEPTIME, CRSARRTIME
                # to return with the model prediction
                dates = [int(year + month + day) for year, month, day in
                         zip(map(str,list(df['YEAR'])), map(str,["%02d" %i for i in list(df['MONTH'])]),
                             map(str,["%02d" %i for i in list(df['DAYOFMONTH'])]))]
                origins = list(df['ORIGIN'])
                dests = list(df['DEST'])
                carriers = list(df['UNIQUECARRIER'])
                deptimes = list(df['CRSDEPTIME'])
                arrtimes = list(df['CRSARRTIME'])

                ##############################
                ### MODEL SCORING & OUTPUT ###
                ##############################

                # Call model to make prediction
                predictions = pipe_lr.predict(df)

                # Calculate probability that flight will be delayed
                probability_delayed = pipe_lr.predict_proba(df)[:,1]

                # Return the result
                for x in range(predictions.shape[0]):
                    self.output(int(dates[x]),str(origins[x]),str(dests[x]),str(carriers[x]),
                                int(deptimes[x]),int(arrtimes[x]),int(predictions[x]), float(probability_delayed[x]))

                row_list = []
                rownum = 0
        self.done()
full_pipeline.run()
```

Registering the Python UDF

```
%%sql
CREATE OR REPLACE FUNCTION
FLIGHT_PREDICTER(SMALLINT,SMALLINT,SMALLINT,SMALLINT,SMALLINT,VARCHAR(8),VARCHAR(3),
              VARCHAR(3),SMALLINT,SMALLINT,SMALLINT,SMALLINT,SMALLINT,SMALLINT,SMALLINT,
              SMALLINT,SMALLINT)
RETURNS TABLE (DATE INTEGER,ORIGIN VARCHAR(3), DEST VARCHAR(3), CARRIER VARCHAR(3),
CRSDEPTIME SMALLINT, CRSARRTIME SMALLINT,PREDICTION SMALLINT,PROB_DELAYED DOUBLE)
LANGUAGE PYTHON PARAMETER STYLE NPSGENERIC  FENCED  NOT THREADSAFE  NO FINAL CALL  DISALLOW PARALLEL  NO DBINFO
DETERMINISTIC NO EXTERNAL ACTION CALLED ON NULL INPUT
NO SQL EXTERNAL NAME '/home/db2inst1/myUDF.py'
```

Creating a Db2 Table to Store Predictions

```
%%sql
DROP TABLE IF EXISTS IDUG.PREDICTIONS
```

```
%%sql
CREATE TABLE IDUG.PREDICTIONS(DATE INTEGER,
                              ORIGIN VARCHAR(3),
                              DEST VARCHAR(3),
                              CARRIER VARCHAR(3),
                              CRSDEPTIME SMALLINT,
                              CRSARRTIME SMALLINT,
                              PREDICTION SMALLINT,
                              PROB_DELAYED DOUBLE)
```

Generating Batch Predictions Using Scikit-Learn Pipeline Deployed on Db2

```
%%time
%%sql
INSERT INTO IDUG.PREDICTIONS(DATE,ORIGIN,DEST,CARRIER,CRSDEPTIME,CRSARRTIME,PREDICTION,PROB_DELAYED)
SELECT f.* from IDUG.FLIGHTS2 i,
TABLE(FLIGHT_PREDICTER(i.YEAR, i.QUARTER, i.MONTH,
                 i.DAYOFMONTH, i.DAYOFWEEK,i.UNIQUECARRIER,
                 i.ORIGIN, i.DEST, i.CRSDEPTIME,
                 i.DEPDELAY, i.DEPDEL15,i.TAXIOUT,i.WHEELSOFF,
                 i.CRSARRTIME, i.CRSELAPSEDTIME, i.AIRTIME, i.DISTANCEGROUP)) f
```

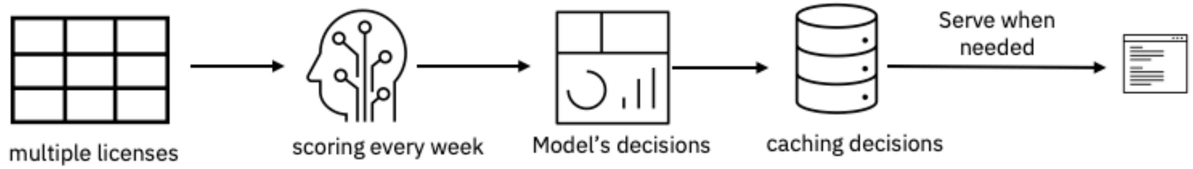
Retrieving the Predictions from Db2

```
query = %sql SELECT * FROM IDUG.PREDICTIONS
result = pd.DataFrame(query)
```

```
# Lets look at the first 10 rows of our UDTF's output
result.head(10)
```

	DATE	ORIGIN	DEST	CARRIER	CRSDEPTIME	CRSARRTIME	PREDICTION	PROB_DELAYED
0	20201215	SFO	MIA	AA	2350	805	1	0.86
1	20200614	HOU	DCA	WN	2000	2355	1	0.77
2	20200627	DFW	PSP	AA	1655	1747	1	0.99
3	20200203	DFW	SLC	OO	1233	1425	1	0.89
4	20200108	PHX	ORD	AA	145	605	0	0.07
5	20201016	ORD	EWR	UA	1812	2132	0	0.07
6	20200914	ORD	BOS	AA	2155	100	1	0.94
7	20200609	PDX	PHX	WN	2005	2240	0	0.25
8	20200226	BOS	TPA	B6	2015	2335	0	0.35
9	20200608	BUF	BOS	B6	1510	1633	0	0.00

Large Batch Scoring with Db2



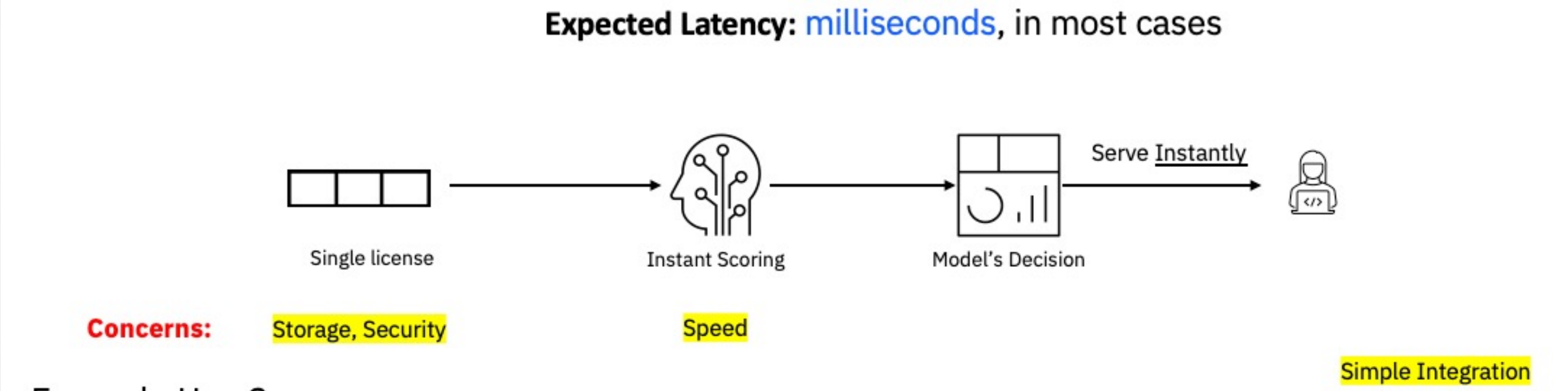
Needs: Storage, Security | Scale | Storage | Simple Integration

Example Use Cases:

- Weekly Personalized Product / Service Recommendations
- Weekly Demand Forecasting
- Quarterly Sales Pipeline Prioritization
- Customers Churn Prediction
- Daily Support Tickets Prioritization
- Product Feature Requests Prioritization

Db2's Solution: Parallelism and Secure Storage

Real-time Scoring with Db2



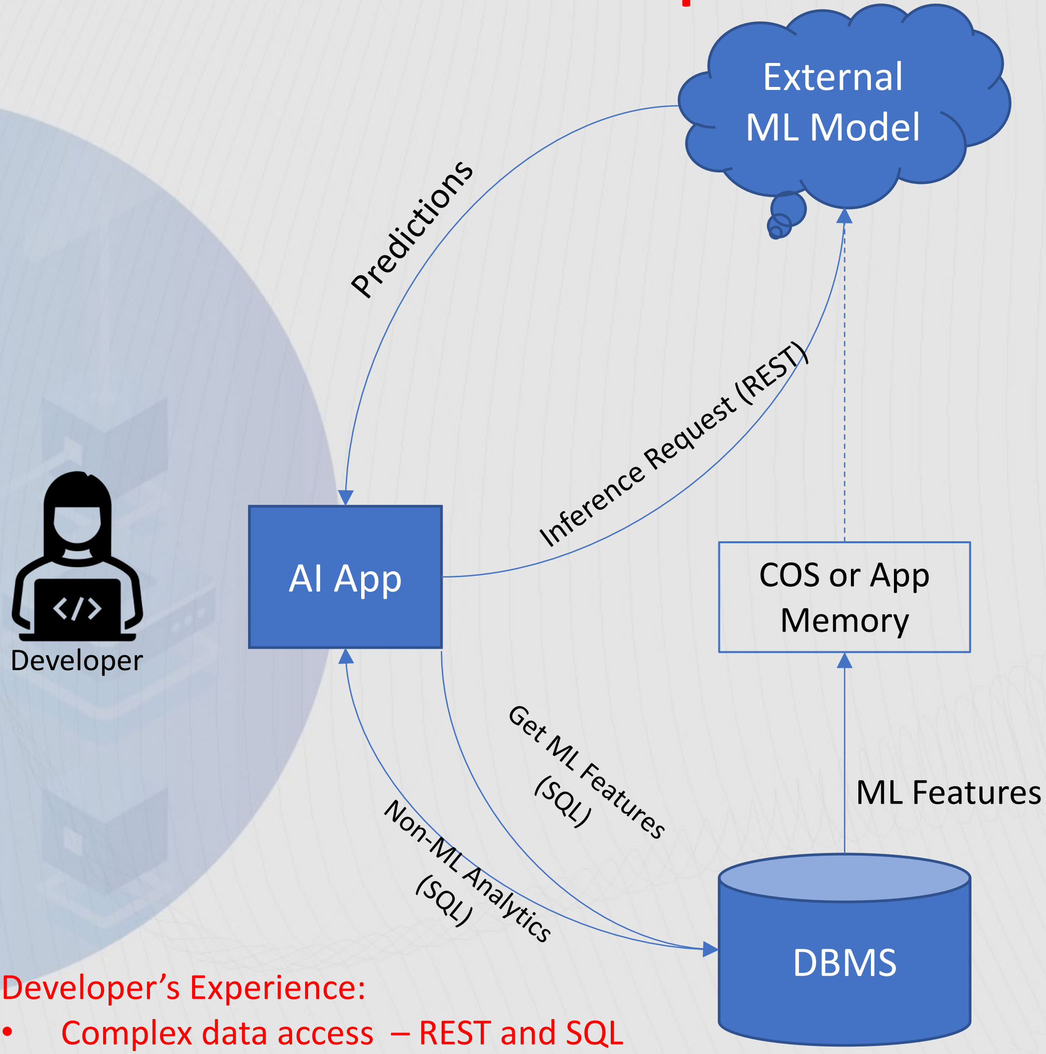
Example Use Cases:

- Fraud Detection
- Network Anomaly Detection
- Product Recommendations
- Instant Mortgage Pre-Qualification

Db2's Solution: co-location of data and ML in the db and trigger

AI App Development Experience

Common Experience



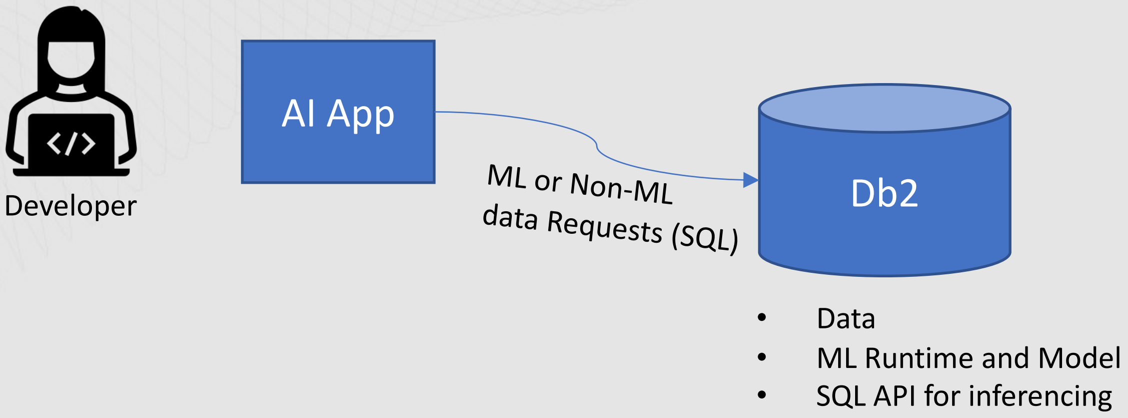
- ML Features
- Non-ML Analytics

Developer's Experience:

- Complex data access – REST and SQL
- COS – governance, $, and setup efforts
- In-App heavy data processing – filtering, combining

Db2 Experience

Goal: make consuming AI as simple as writing a SQL Query



Developer's Experience:

- Simple data access – SQL
- ~~COS – governance, $, and setup efforts~~
- ~~In-App heavy data processing – filtering, combining~~
- Security
- Scalability

Demos and **Tutorials**

Demos:

- Build a Customer Segmentation Model with Db2 (K-Means Clustering)
- Build a Classification Model with Db2 (Decision Tree)
- Build a Regression Model with Db2 (Linear Regression)
- Integrate a Db2-native model with a Cognos Dashboard
- Deploying a ML Model Trained on Cloud Pak for Data to Db2
- Automated AI Model Development with IBM Cloud Pak for Data and Db2

Tutorials:

- Tutorials and Jupyter Notebooks
- How to Build an in-database Linear Regression Model with IBM Db2
- How to build a decision tree model in IBM Db2

Documentation:

- Db2 11.5 Knowledge Center

Thank You

Speaker: Shaikh Quader

Company: IBM

Email: shaikhq@ca.ibm.com

LinkedIn: /in/shaikhquader

Medium: @shaikhquader