TECH_321: Using Machine Learning Tools on z/OS

Speakers:
Dave Levish, Revanth Akella

Introduction

We developed an ML-based estimation function for a product that works with Db2. For this project, we had several goals:

- Fast model training on hundreds of thousands to millions of rows of training data
- Model execution must also be fast
- Data should remain on z/OS from beginning to end
- Model training and execution flows should be kept simple
- Leverage tools that already exist on z/OS

Advantages of native ML development on z/OS

1. Avoiding data conversion issues:
   a. Different integer formats (little-endian vs. big-endian)
   b. Different floating point formats: HFP is z/Architecture specific
   c. Character set conversion issues
2. Multi-Platform Security Issues:
   a. Potentially sensitive data in Db2 tables – Keeping data processing, training data extraction, model training and model useage on "native" z/OS and USS removes the need for remote or REST based solutions.
   b. Avoid the need to configure consistent security across multiple platforms with differing security mechanisms.

Programming Language: Python

Python has many, industry-standard, data science and machine learning libraries. IBM has ported some of these to z/OS USS with the Python AI Toolkit for z/OS (5698-PAL).

Data processing libraries such as pandas, and ML libraries like scikit learn, are commonly used across the industry and are taught to students in universities.

The tools make working with data and training models straightforward. These tools are also well supported and open source.

ML Tools we used that are ported to USS:

1. ML Prototyping and Visualization: Jupyter Notebook
2. Data cleaning and transformation: pandas
3. Data visualization: matplotlib, seaborn
4. Modeling and evaluation: scikit-learn

These tools enable us to run all ML operations on z/OS without needing to move data between multiple platforms.

Important: Tasks requiring Deep Learning – a subset of Machine Learning that uses Deep Neural Networks – would require libraries such as tensorflow and pytorch. These are currently not available on USS via the Python AI Toolkit for z/OS. Similarly, the option to run arbitrary Large Language Models locally on USS is not available at the time this session was created.

Types of ML Tasks:

Supervised Learning:

Machine learning from examples. There is a target variable that is being predicted based on existing information in the data.

Examples: Housing price prediction, spam detection, etc.

Unsupervised Learning:

Machine learning to find patterns in the data. There are no target variables, but we have features that can be used to create groupings.

Examples: Recommendation systems, customer segmentation

The focus of this session is supervised learning.

Supervised Learning

There are 2 popular types of supervised learning:

- Classification
- Regression

We will cover both and go over the process of each type of learning using Jupyter.

Classification

The task of predicting which category a data point falls into. This is done by training with data that already has these categories assigned to data points. The features are used to draw relationships with the categories. These models are used to predict which category a new data point belongs to.

If there are 2 categories, the classification is called Binary Classification.

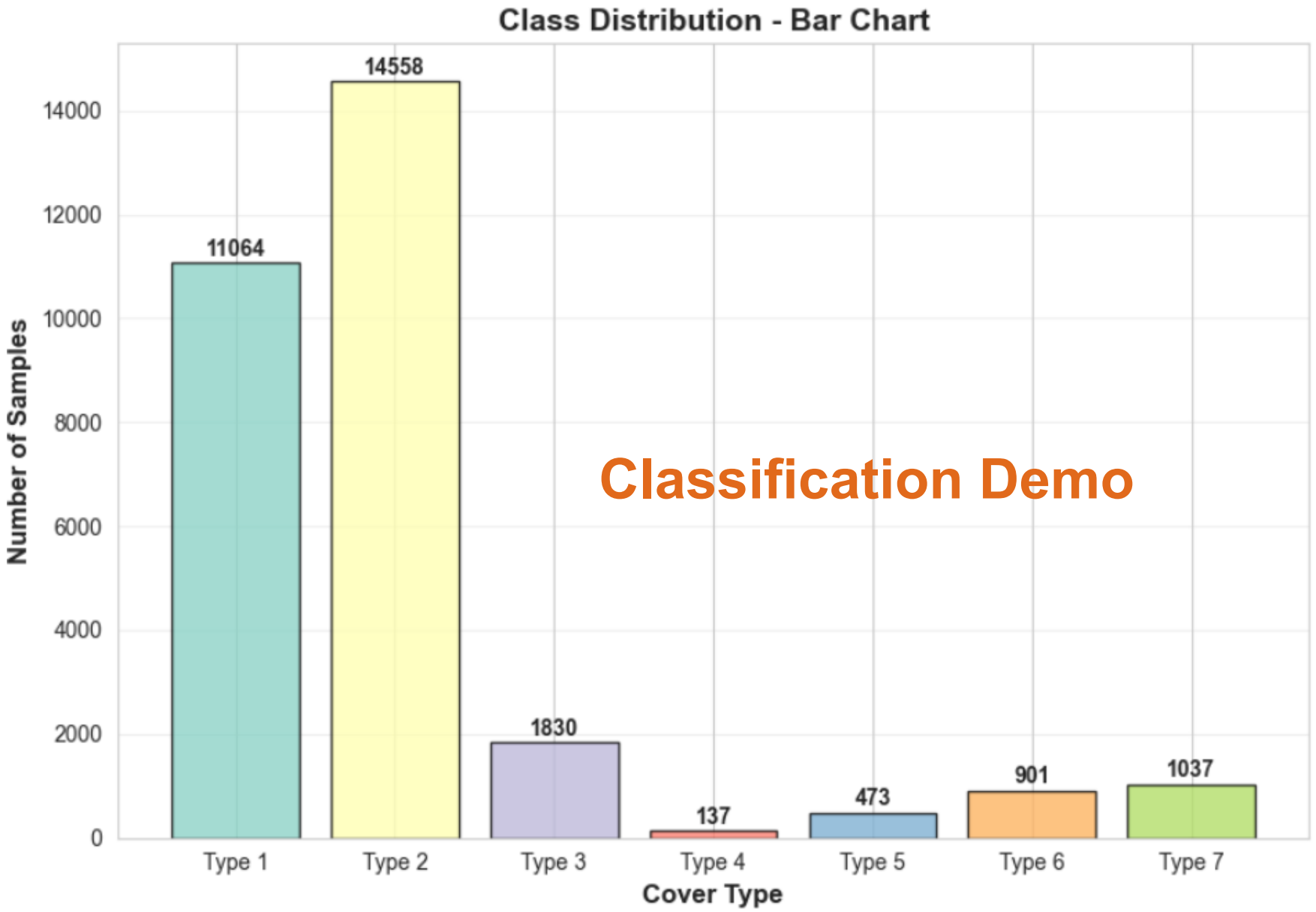
E.g., Spam Detection

If there are more than 2, then the classification is called a Multi-Class Classification.

E.g., Categorizing News topics into categories.

Next:

Classification Example walk-through in a Jupyter Notebook – Forest Cover Prediction



Classification Demo

Basis:
UC Irvine Covertype dataset (1998)

Classification Training Script

```
Training data:
    Elevation  Aspect  Slope  ...  Wilderness_Area3  Wilderness_Area4  Cover_Type
0        2596      51      3  ...                 0                 0           5
1        2590      56      2  ...                 0                 0           5
2        2804     139      9  ...                 0                 0           2
3        2785     155     18  ...                 0                 0           2
4        2595      45      2  ...                 0                 0           5

[5 rows x 55 columns]
Running training. Please wait.
Training complete in:  13.87  seconds
Model trained on  464799 rows
accuracy score:  0.95
Model saved as  classification_model.mdl
accuracy score:  0.95
Time to save model:  8.14  seconds
training time:  26.82  seconds
```

Classification Test Data

```
Test data:
    Elevation  Aspect  ...  Wilderness_Area3  Wilderness_Area4
0        2426     168  ...                 1                 0
1        2423     169  ...                 1                 0
2        2421     172  ...                 1                 0
3        2419     168  ...                 1                 0
4        2415     161  ...                 1                 0

[5 rows x 54 columns]
```

Classification Estimation Script

```
Running classification estimation script.

Loading model for estimation. Please wait.

Model classification_model.mdl loaded in  2.74  seconds


Estimated forest cover type:
    Elevation  Aspect  Slope  ...  Wilderness_Area3  Wilderness_Area4  Cover_Type
0        2426     168     24  ...                 1                 0           1
1        2423     169     24  ...                 1                 0           1
2        2421     172     25  ...                 1                 0           1
3        2419     168     25  ...                 1                 0           1
4        2415     161     25  ...                 1                 0           1

[5 rows x 55 columns]
Total Estimation time:  6.65  seconds
```

Regression

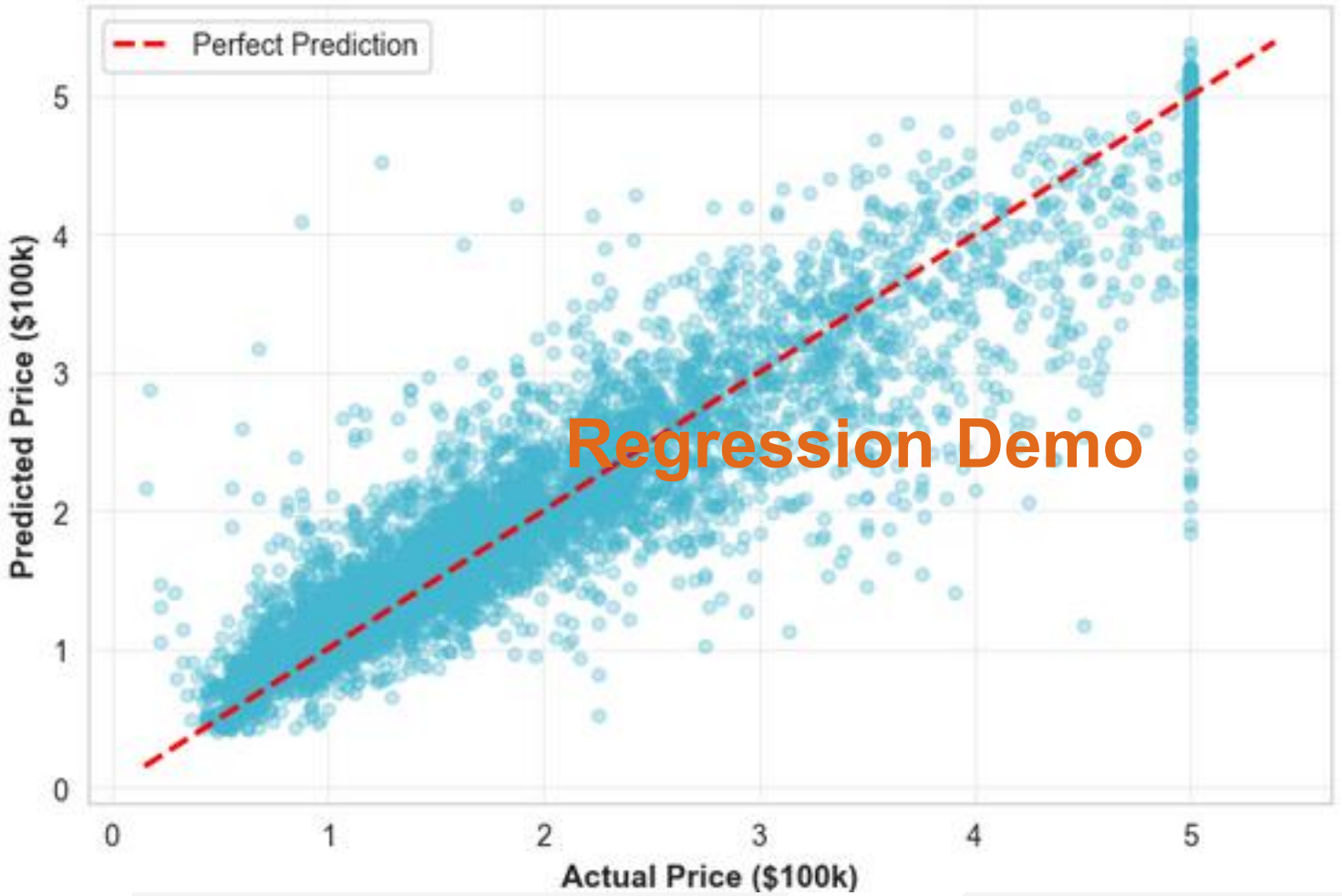
Regression is the task of predicting continuous value outputs. There are no classes or labels but a continuous value that is the predicted output. A good example is predicting housing prices. Based on features in the data such as area/zip code, house size, lot size, state, proximity to schools and parks, we can predict the price of the home.

Regression with a single feature is called **univariate** regression.

When we use multiple features as in the case of housing prices its **multivariate** regression.

Next:

Multivariate regression example walk-through in a Jupyter Notebook – California Housing Price Prediction.



Basis: Sample of 1990 census data

Regression Training Script

```
Training data:
    MedInc  AveRooms  AveOccup  Latitude  Longitude MedHouseVal  GeoCluster
0  8.3252  6.984127  2.555556     37.88    -122.23   $452600.0           1
1  8.3014  6.238137  2.109842     37.86    -122.22   $358500.0           1
2  7.2574  8.288136  2.802260     37.85    -122.24   $352100.0           1
3  5.6431  5.817352  2.547945     37.85    -122.25   $341300.0           1
4  3.8462  6.281853  2.181467     37.85    -122.25   $342200.0           1
Running training. Please wait.
Training complete in:  0.33  seconds
Model trained on  16503 rows
KNN model saved as  regression_model.mdl
R2 score:  0.76
Time to save model:  0.01  seconds
total training time:  2.88  seconds
```

Regression Test Data

```
Test data:
    MedInc  AveRooms  AveOccup  Latitude  Longitude  GeoCluster
0   2.0943  5.519802  3.801980     39.12    -121.39           8
1   3.5673  5.932584  2.824719     39.29    -121.32           8
2   3.5179  6.145833  2.777778     39.33    -121.40           3
3   3.1250  6.023377  2.719481     39.26    -121.45           3
4   2.5495  5.445026  2.832461     39.19    -121.53           8
(test_env) (mlenv) Regression$
```

Regression Estimation Script

```
Running regression estimation script.

Loading model for estimation. Please wait.

Model regression_model.mdl loaded in  0.01  seconds


Estimated median housing values:
    MedInc  AveRooms  AveOccup  Latitude  Longitude  GeoCluster      Value
0  2.0943  5.519802  3.801980     39.12    -121.39           8   $71050.0
1  3.5673  5.932584  2.824719     39.29    -121.32           8  $137583.0
2  3.5179  6.145833  2.777778     39.33    -121.40           3  $110283.0
3  3.1250  6.023377  2.719481     39.26    -121.45           3   $85650.0
4  2.5495  5.445026  2.832461     39.19    -121.53           8  $109850.0
Total estimation time:  2.56  seconds
```

Links to tools used

1. Python AI Toolkit for z/OS: https://ibm-z-oss-oda.github.io/python_ai_toolkit_zos/
2. Scikit-Learn: https://scikit-learn.org/stable/
3. Pandas: https://pandas.pydata.org/
4. Jupyter: https://jupyter.org/

Your feedback is important!

SHARE
EDUCATE • NETWORK • INFLUENCE

Submit a session evaluation for each session you attend:

www.share.org/evaluation