



AI on IBM Z

# Health insurance claims solution template

This solution template provides an example on how to deploy AI using an IBM Z environment, while making use of open source frameworks, Machine Learning for IBM z/OS (MLz), and more.

Within this solution template, there are various phases of the AI lifecycle included. Work through each of the following steps to deploy your own health insurance claims solution on IBM Z.

Table of contents

AI model training.....3

AI model deployment.....7

AI model integration.....11



## Step 1

# AI model training

We will build a health insurance claims AI model by training with the provided Rapid AI on IBM Z Development Jupyter notebook. Simply point the Jupyter notebook to your dataset and run it to generate your AI model. This trained AI model can then be deployed with MLz.

All sample code for this section is within

```
ai-st-health-insurance-claims/zST-model-training-jupyter
```

## Prerequisites

1. Must have Python (3.9 or 3.10) installed

## Dataset guidance

Sample health insurance claims dataset can be found on Kaggle -

[https://www.kaggle.com/datasets/grvmishra7/health-insurance-prediction?select=train\\_Df64byy.csv](https://www.kaggle.com/datasets/grvmishra7/health-insurance-prediction?select=train_Df64byy.csv)

## Required features

- applicant\_id
- years\_of\_insurance\_with\_us
- regular\_checkup\_last\_year
- adventure\_sports
- occupation
- visited\_doctor\_last\_1\_year
- cholesterol\_level
- daily\_avg\_steps
- age
- heart\_decs\_history
- any\_other\_major\_decs\_history
- gender
- avg\_glucose\_level
- bmi
- smoking\_status
- year\_last\_admitted
- location
- weight
- covered\_by\_any\_other\_company
- alcohol
- exercise
- weight\_change\_in\_last\_one\_year
- fat\_percentage
- insurance\_cost

[Access rapid AI on IBM Z development environment](#)

[Provide data](#)

[Model training](#)

[Access trained AI model](#)

## Access rapid AI on IBM Z development environment

1. Create and activate Python virtual environment

```
python -m venv env
source env/bin/activate
```

2. Install required Python packages

```
pip install -r requirements.txt
```

3. Run Jupyter

```
jupyter notebook
```

4. View Jupyter interface

Go to [localhost:8888](http://localhost:8888) in a web browser

5. Click on ai\_on\_z\_model\_dev.ipynb in web browser

```

AI on IBM Z Model Development

Import required python packages

In [ ]: import numpy as np
import pandas as pd
import sys
import time

# Model training
from sklearn.pipeline import Pipeline
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.ensemble import GradientBoostingClassifier, RandomForestClassifier
from sklearn.metrics import precision_score, recall_score, f1_score, accuracy_score

# Data preprocessing
from sklearn.preprocessing import OrdinalEncoder, StandardScaler

# FMML
from sklearn.pipeline import Pipeline
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline

Input dataset and label

In [ ]: # User must provide filepath to dataset and label name
DATASET_FILENAME = 'datasets/credit_card_transactions.csv'
DATASET_LABEL_NAME = 'ix_fmml'

Split features and labels from dataset

In [ ]: def split_features_and_labels(dataset_df, label):
    features = dataset_df.copy()
    labels = features.pop(label)
    return features, labels
  
```

## Provide data

1. Add your input dataset (csv) into datasets/ directory
2. Add input data to Jupyter notebook
  - Set DATASET\_FILENAME to the path to your dataset
  - Set DATASET\_LABEL\_NAME to the name of the column you're predicting from the dataset



[Access rapid AI on IBM Z development environment](#)

[Provide data](#)

[Model training](#)

[Access trained AI model](#)

## Model training

1. Step through and run Jupyter notebook from web browser

```

# Model training
from sklearn.pipeline import Pipeline
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import GradientBoostingClassifier, RandomForestClassifier
from sklearn.metrics import precision_score, recall_score, f1_score, accuracy_score

# Data preprocessing
from sklearn.preprocessing import OrdinalEncoder, StandardScaler

# Train
from sklearn.dummy import sklearn_dummy
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline

# Import required libraries
import numpy
import pandas
import json
import time

# You must provide filepath to dataset and label name
DATASET_FILEPATH = 'dataset/cowditi_xavi_transactions.csv'
DATASET_LABEL_NAME = 'Is Fraud?'

def split_features_and_labels(dataset_filepath, label):
    features = dataset_df.copy()
    labels = features.pop(label)
    return features, labels
    
```

## Access trained AI model

1. Once training is complete, you can find your AI models within the models/directory (choose one for the following AI model deployment step)



---

✓ 1. AI model training

○ 2. AI model deployment

○ 3. AI model integration

✓ AI model training complete



### Prerequisites

1. Must have MLz installed

### Step 2

## AI model deployment

We will deploy our health insurance AI model using MLz. We can utilize the model import functionality on the MLz UI. This deployed AI model can then be integrated into applications within the IBM Z environment.

[Go to MLz UI](#)

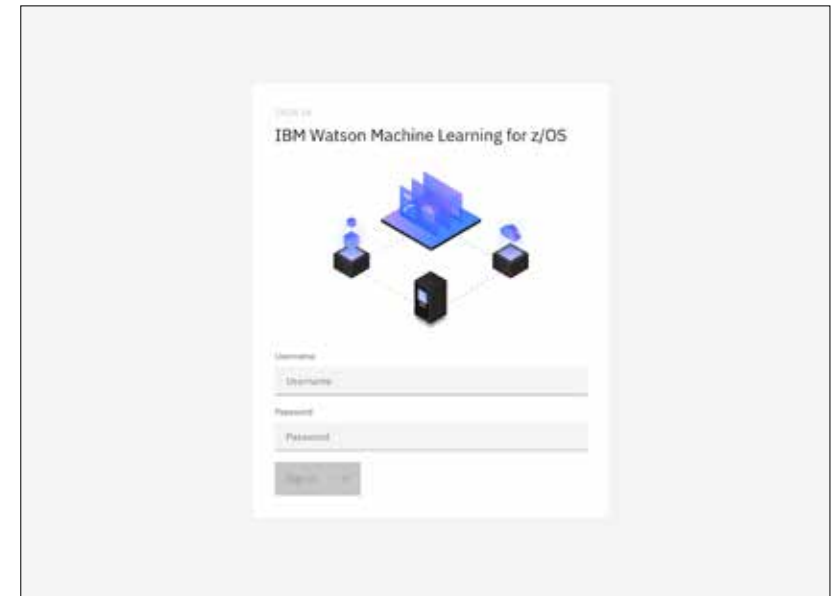
[Import AI model](#)

[Deploy AI model](#)

[View deployed AI model](#)

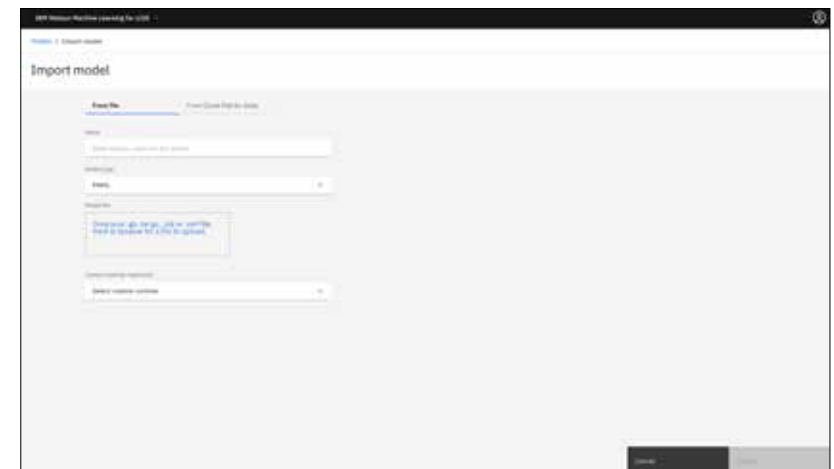
## Go to MLz UI

1. Sign in with username/password



## Import AI model

1. Go to models tab
2. Click import model
3. Enter model name
4. Choose model type  
Choose PMML if using your previously trained model
5. Drag and drop model file  
Use your previously trained model
6. Click import





[Go to MLz UI](#)

[Import AI model](#)

[Deploy AI model](#)

[View deployed AI model](#)

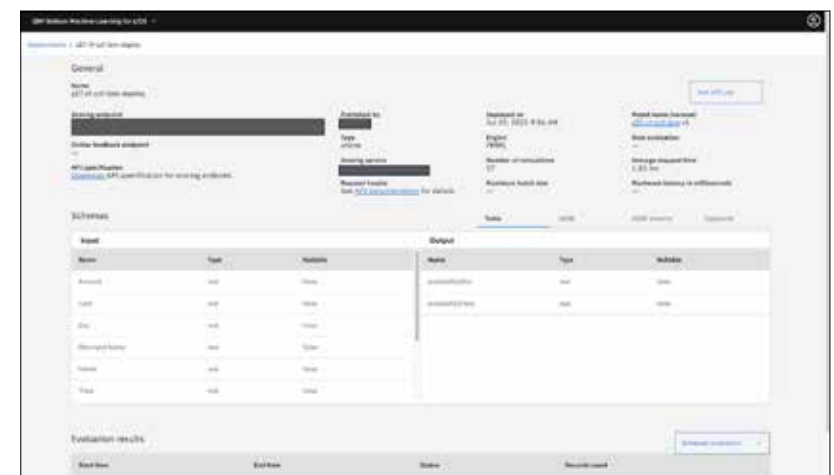
## Deploy AI model

1. Go to models tab
2. Click action button for your model (on right side)
3. Click deploy
4. Enter deployment name
5. Choose deployment type
6. Choose model version
7. Choose scoring service  
Note: you should choose the correct scoring service based on your application (e.g. CICS or REST)
8. Click create



## View deployed AI model

1. Go to deployments tab
2. Click on action button for your deployed model (on right side)
3. Click view details



---

✓ 1. AI model training

✓ 2. AI model deployment

○ 3. AI model integration

✓ AI model deployment complete



### Prerequisites

1. Must have node.js v16 or newer installed
2. Must have Docker installed
3. Must have Git installed

### Step 3

## AI model integration

We can use our deployed MLz health insurance claims AI model and integrate it into different types of applications. Guidance on integrating the AI model into a sample health insurance claims application is below.

All sample code for this section is within

```
ai-st-health-insurance-claims/zST-model-integration-HIC
```

[Get model details for inferencing](#)

### Configure sample application

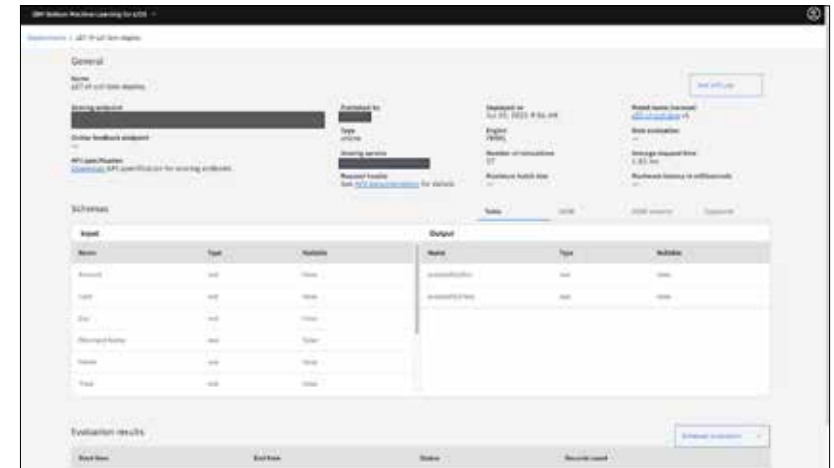
### Build sample application

## Deploy sample application

### Access sample application

## Get model details for inferencing

1. Go to MLz UI
2. Go to deployments tab
3. Click on action button for your deployed model (on right side)
4. Click view details
5. Copy scoring endpoint



## Configure sample application

## Set the environment variables within env.list file

- WML\_URL (MLz UI URL)
- WML\_USER (username for MLz)
- WML\_PASS (password for MLz user)
- SCORING\_URL (scoring endpoint for deployed AI model)

## Build sample application

1. Run command in terminal

```
docker build -t health-insurance .
```

[Get model details for inferencing](#)

[Configure sample application](#)

[Build sample application](#)

[Deploy sample application](#)

[Access sample application](#)

## Deploy sample application

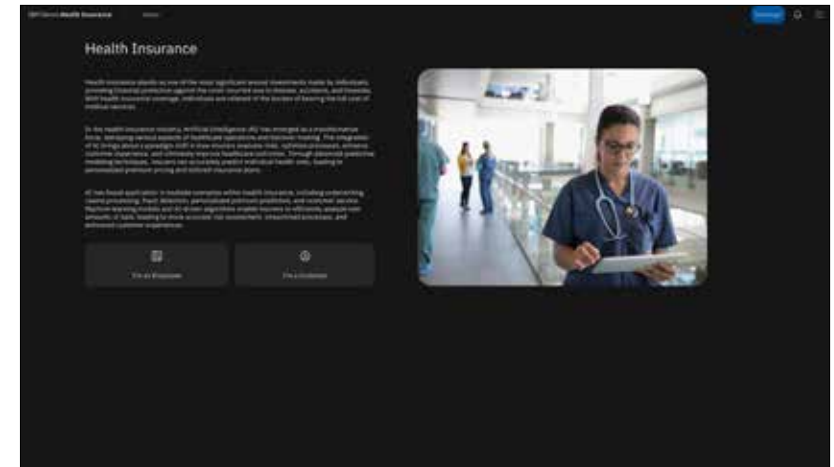
1. Run command in terminal (e.g. port 9000)

```
docker run -p 9000:80 --env-file env.list
--name health-insurance-app -d health-in-
surance
```

## Access sample application

View the following URL in a web browser: `http://{ip address}:{port}/ui`

- ip address: IP of server you deployed application in
- port: port you used with docker run



---

✓ 1. AI model training

✓ 2. AI model deployment

✓ 3. AI model integration

✓ AI model integration complete