



AI on Linux on Z

# Fraud detection solution template

This solution template provides an example on how to deploy AI using an Linux on Z environment, while making use of open source frameworks, Triton Inference Server (TIS), 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 fraud detection solution on IBM Z.

Table of contents

AI model training.....3

AI model deployment.....7

AI model analysis.....11

AI model integration.....16



## Step 1

# AI model training

We will build a fraud detection AI model by training with the provided rapid AI on Linux on 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 TIS.

All sample code for this section is within

```
ai-st-fraud-detection-tis/zST-model-training-jupyter
```

## Prerequisites

1. Must have Python (3.9 or 3.10) installed

## Dataset guidance

Sample open source credit card transaction dataset can be found on Kaggle -

<https://www.kaggle.com/datasets/ealtman2019/credit-card-transactions>

There are several files included within the download. You can use *credit\_card\_transactions-ibm\_v2.csv* for training. Due to the size of the sample dataset, the provided Jupyter notebook takes a subset of the data to decrease the training time. Please modify the code in the “Fetch and process data” cell of the provided Jupyter notebook later to use more data during training.

## Required features

- User (integer) – unique ID for user making transaction
- Card (integer) – unique ID for credit card
- Year (integer) – year of the transaction
- Month (integer) – month of the transaction
- Day (integer) – day of the month of the transaction
- Time (integer) - time of the transaction (HH:MM)
- Amount (float) – dollar amount of the transaction
- Use Chip (string) – the type of transaction (swipe transaction, etc)
- Merchant Name (integer) – unique ID for merchant name
- Zip (integer) – zip code of the transaction

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

[Provide data](#)

[Model training](#)

[Access trained AI model](#)

## Access rapid AI on Linux on 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

```

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

# FUNC
from sklearn.pipeline import make_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 = 'Is Fraud?'

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 Linux on Z development environment](#)

[Provide data](#)

[Model training](#)

[Access trained AI model](#)

## Model training

1. Step through and run Jupyter notebook from web browser

```

AI on IBM

Import req
In [1]: import numpy
import pandas
import json
import time

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

Input dataset and label
In [2]: # User must provide filepath to dataset and label name
DATASET_FILEPATH = 'dataset/crowdai_rapid_transactions.csv'
DATASET_LABEL_NAME = 'Is Fraud?'

Split features and labels from dataset
In [3]: def split_features_and_labels(dataset_filepath, label):
    features = dataset_filepath.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 analysis

☐ 4. AI model integration

☒ AI model training complete



### Prerequisites

1. Must have Docker or Podman installed

### Step 2

## AI model deployment

We will deploy our fraud detection AI model using TIS. We can utilize the AI Toolkit to leverage TIS for model deployment. This deployed AI model can then be integrated into applications within the Linux on Z environment.

[Build Triton Inference Server](#)

[Integrate AI model into  
Triton Inference Server](#)

[Deploy Triton  
Inference Server](#)

[Run sample test](#)

## Build Triton inference server

1. Build docker image

```
docker build -t zst-tis .
```

## Integrate AI model into Triton Inference Server

1. Add your model (.pmml file) to

```
ai-st-fraud-detection-tis/zST-model-  
deployment/zST/models
```

directory

2. Add your preprocessing .joblib file to

```
ai-st-fraud-detection-tis/zST-model-  
deployment/zST/models
```

directory

## Deploy Triton inference server

1. Create docker network

```
docker network create my-data-network
```



[Build Triton Inference Server](#)

[Integrate AI model into  
Triton Inference Server](#)

[Deploy Triton  
Inference Server](#)

[Run sample test](#)

## 2. Run docker container

```
docker run --net=my-data-network --  
shm-size 1G -u root --rm -p8000:8000  
-v//$PWD/zST/models:/models zst-tis  
tritonserver --model-repository=/models
```

## Run sample test

1. Run python script from terminal with ip/port of triton inference server

```
cd zST  
python inference_request.py
```

---

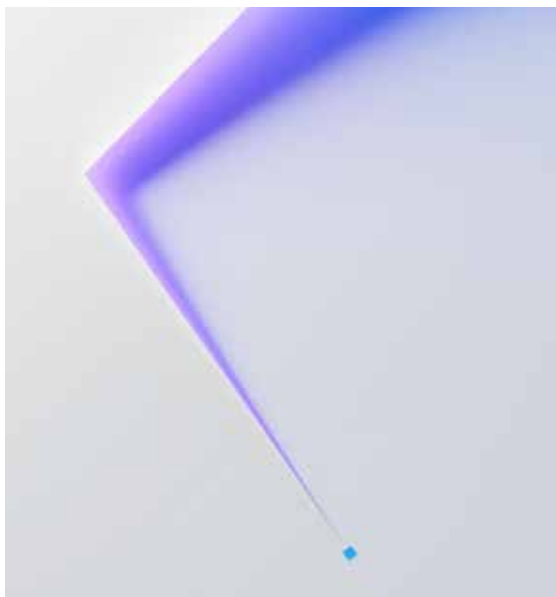
✓ 1. AI model training

✓ 2. AI model deployment

○ 3. AI model analysis

○ 4. AI model integration

✓ AI model deployment complete



### Prerequisites

1. Must have TIS installed
2. Must have Python installed
3. Must have Git installed
4. Must have Docker or Podman installed

### Step 3

## AI model analysis

We will analyze our fraud detection AI model using a sample AI on Linux on Z Fraud Detection Dashboard. We can invoke the API of this sample dashboard from another sample application to visualize the AI model inferencing.

All sample code for this section is within

```
ai-st-fraud-detection-tis/zST-model-analysis
```

[Configure sample application](#)

[Build sample application](#)

[Deploy sample application](#)

[Access sample application](#)

[Analyze credit card events](#)

[Make predication](#)

## Configure sample application

1. Set the enrionment variables within

```
ai-st-fraud-detection-tis/zST-model-  
analysis/env.list
```

- TIS\_ENDPOINT (scoring endpoint for triton)

## Build sample application

1. Run command in terminal

```
docker build -t model-analysis
```

## Deploy sample application

1. Run command in terminal (e.g. port 5002)

```
docker run --net=my-data-network --rm -  
p 5002:5002 --env-file env.list --name  
model-analysis-app model-analysis
```

[Configure sample application](#)

[Build sample application](#)

[Deploy sample application](#)

[Access sample application](#)

[Analyze credit card events](#)

[Make predication](#)

## Access sample application

1. View the following URL in a web browser  
<http://{ip address}:{port}>
  - IP address: IP of server you deployed application in
  - Port: port you used with Docker run

## Analyze credit card events

1. Go to sample insights dashboard in web browser  
<http://{ip address}:{port}>
2. Go to latest predictions tab



Sample AI on IBM Z Analytics Dashboard		
Latest Predictions		Make Prediction
Latest Predictions		
prediction	deployment_id	time
Not Fraud	742a558f-c8a3-4473-a071-6335a8a81c17	2020-08-10 13:30:28
Not Fraud	742a558f-c8a3-4473-a071-6335a8a81c17	2020-08-10 13:30:31

[Configure sample application](#)

[Build sample application](#)

[Deploy sample application](#)

[Access sample application](#)

[Analyze credit card events](#)

[Make predication](#)

## Make predication

1. Go to sample insights dashboard in web browser  
`http://{ip address}:{port}`
2. Go to latest make predication tab
3. Input json data
4. Click submit



The screenshot shows a web application titled 'Sample AI on Azure IoT Insights App' with tabs for 'Latest Predictions' and 'Make Prediction'. The 'Make Prediction' tab is active. It features a 'Prediction' section with a 'Transaction Data' label and a large text area containing a JSON array of transaction records. Below the text area is a 'Submit' button. The footer of the application reads '© 2019 All Rights Reserved'.

```
{
  "transaction": {
    "id": "1",
    "amount": 100,
    "merchant": "M1",
    "category": "C1",
    "timestamp": "2019-01-01T00:00:00Z"
  }
}
```

---

✓ 1. AI model training

✓ 2. AI model deployment

✓ 3. AI model analysis

○ 4. AI model integration

✓ AI model analysis complete



#### Step 4

## AI model integration

We can use our deployed TIS fraud detection AI model and integrate it into different types of applications. The AI model can be analyzed and/or provide inferencing APIs using a the sample AI on Linux on Z Fraud Detection Dashboard.

All sample code for this section is within

```
ai-st-fraud-detection-tis/zST-storefront-evershop
```

#### Prerequisites

1. Must have AI on IBM Z Sample Fraud Detection Dashboard deployed for inferencing and analysis
2. Must have Docker installed



[Install and deploy sample  
ecommerce application](#)

[Configure sample  
ecommerce application](#)

[Access admin panel from  
web browser](#)

[Add products](#)

[Configure store settings](#)

[Configure payment  
settings](#)

[Configure shipping  
settings](#)

[Access sampe ecommerce  
application](#)

[Use fraud detection AI model  
with EverShop Storefront](#)

## Install and deploy sample ecommerce application

1. Set app\_url\_w\_port variable in CheckoutForm.jsx to your server IP and port (ip:port)

```
ai-st-fraud-detection-tis/zST-storefront-  
evershop/packages/evershop/src/components/  
frontStore/stripe/checkout/CheckoutForm.  
jsx
```

2. Run command in terminal

```
docker-compose up
```

## Configure sample ecommerce application

### Access admin panel from web browser

1. Enter URL in web browser using app url (e.g. localhost)

```
http://localhost:3000/admin
```

2. Login with default admin credentials

- Email: admin@test.com
- Password: password

[Install and deploy sample  
ecommerce application](#)

[Configure sample  
ecommerce application](#)

[Access admin panel from  
web browser](#)

[Add products](#)

[Configure store settings](#)

[Configure payment  
settings](#)

[Configure shipping  
settings](#)

[Access sampe ecommerce  
application](#)

[Use fraud detection AI model  
with EverShop Storefront](#)

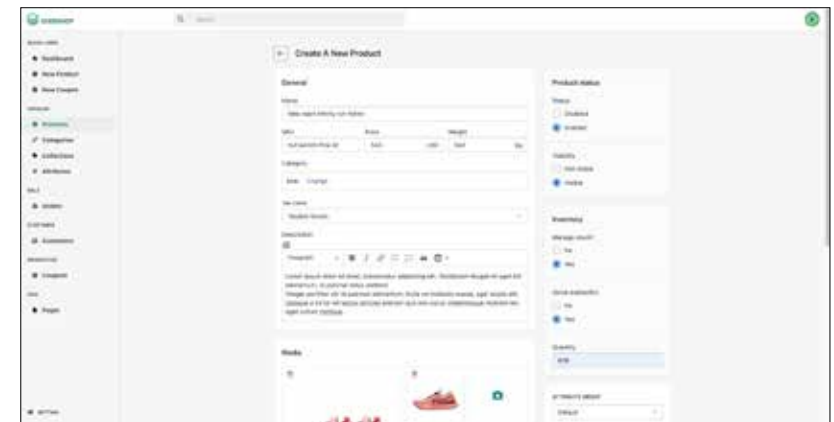
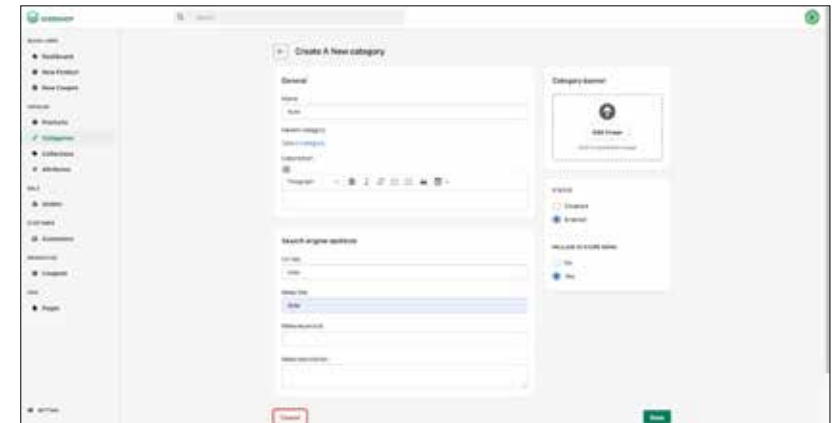
## Add products

### 1. Create categories

- Click categories from the catalog section
- Click new category
- Add category details
  - Add name (e.g. Kids)
  - Add url key
  - Add meta title
  - Change status to enabled
  - Change include in store menu to yes
- Click save

### 2. Add products

- Click products from the catalog section
- Click new product
- Add category details
  - Make sure to change add category
  - Make sure to change status to enabled
  - Make sure to change visibility to visible



Install and deploy sample  
ecommerce application

[Configure sample  
ecommerce application](#)

Access admin panel from  
web browser

Add products

[Configure store settings](#)

[Configure payment  
settings](#)

[Configure shipping  
settings](#)

Access sampe ecommerce  
application

Use fraud detection AI model  
with EverShop Storefront

## Configure store settings

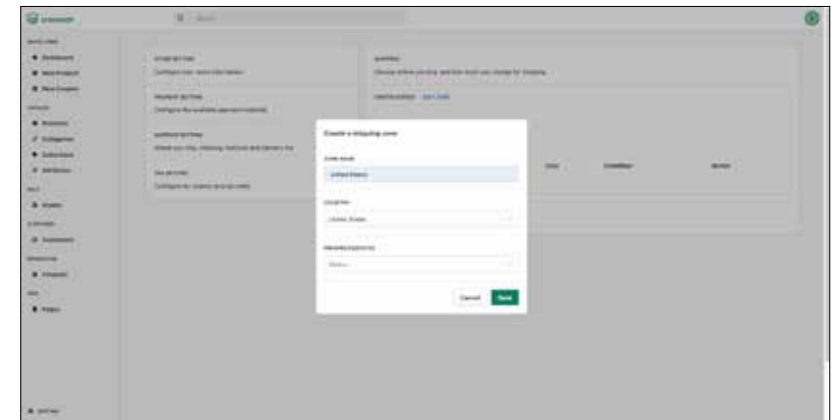
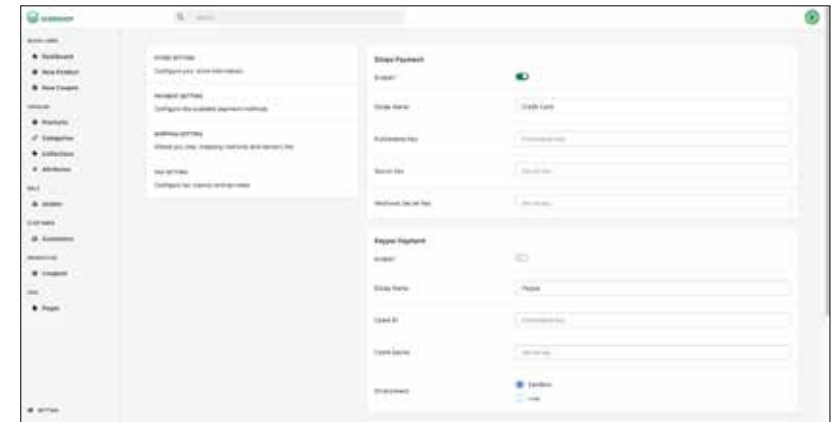
1. Enter URL in web browser using app url  
(e.g. localhost)
2. Click store setting

## Configure payment settings

1. Click setting on the bottom left
2. Click payment setting
3. Enable stripe payment
4. Click save

## Configure shipping settings

1. Add shipping zone
  - Click setting on the bottom left
  - Click shipping setting
  - Click create new shipping zone
  - Add shipping details
  - Click save



## ✓ 1. AI model training

Install and deploy sample  
ecommerce application

[Configure sample  
ecommerce application](#)

Access admin panel from  
web browser

Add products

Configure store settings

Configure payment  
settings

[Configure shipping  
settings](#)

[Access sampe ecommerce  
application](#)

Use fraud detection AI model  
with EverShop Storefront

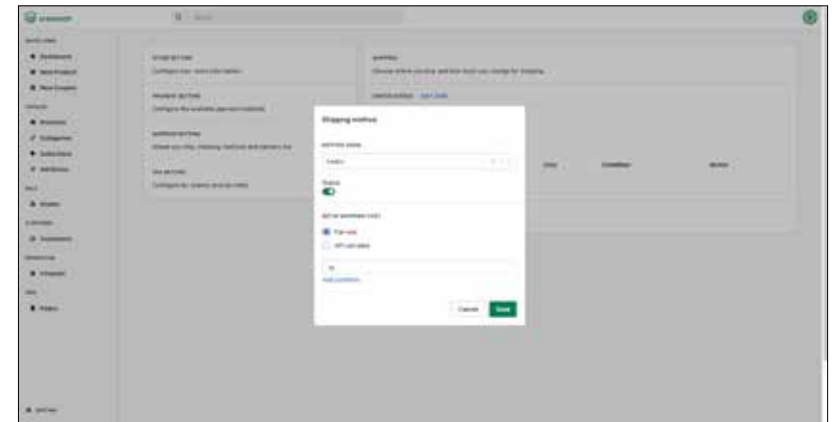
## ✓ 2. AI model deployment

### 2. Add payment method

- Click add method
- Add method name (e.g. FedEx)
- Enable status
- Add flat rate (e.g. 10)
- Click save

## ✓ 3. AI model analysis

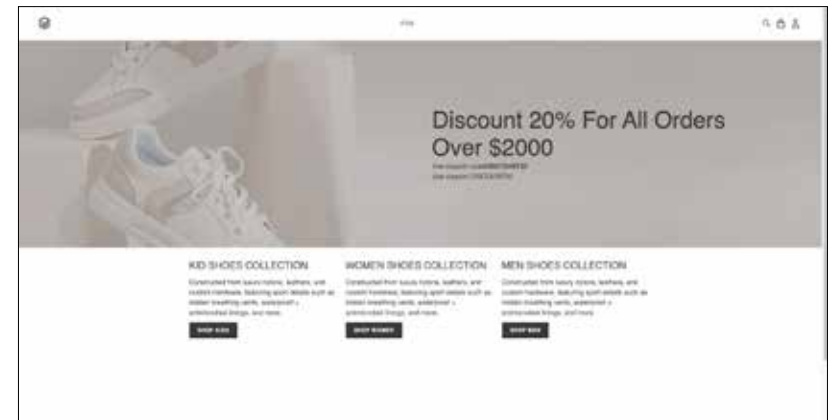
## 4. AI model integration



## Access sampe ecommerce application

1. Enter URL in web browser using app url  
(e.g. localhost)

`http://localhost:3000`



[Install and deploy sample  
ecommerce application](#)

[Configure sample  
ecommerce application](#)

[Access admin panel from  
web browser](#)

[Add products](#)

[Configure store settings](#)

[Configure payment  
settings](#)

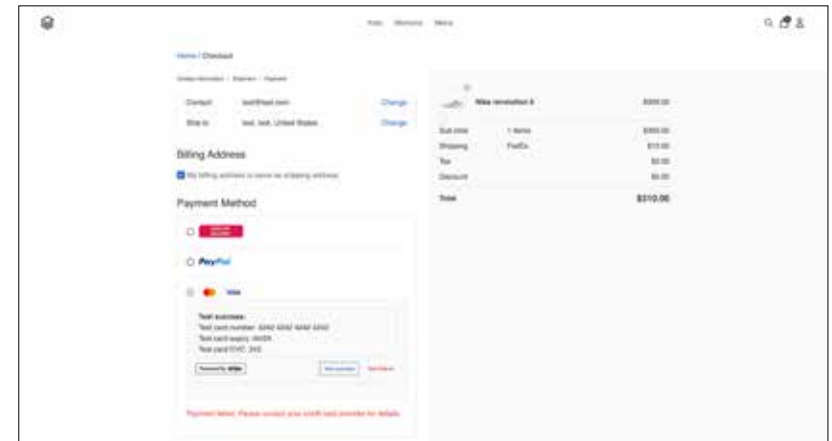
[Configure shipping  
settings](#)

[Access sampe ecommerce  
application](#)

[Use fraud detection AI model  
with EverShop Storefront](#)

## Use fraud detection AI model with EverShop Storefront

1. Make sure you have AI on Linux on Z Sample Fraud Detection Dashboard deployed for inferencing and analysis on the same local system
2. AI on Linux on Z Sample Fraud Detection Dashboard is configured to invoke TIS AI model
3. Add items to cart
4. Place order
  - Choose test failure as payment method for fraud transaction example
  - Choose test success as payment method for non fraud transaction example



---

✓ 1. AI model training

✓ 2. AI model deployment

✓ 3. AI model analysis

✓ 4. AI model integration

✓ AI model integration complete