**Error 1:**

**Error**: 'StepArtifact' object is not subscriptable

**Solution**: debug the code line by line. If there Is any bug in previous line, this issue will come. So debug each line separately to find the error then this error will not occur

**Error 2:**

**Error:** Error occured when running the main pipeline

Full Exception Traceback:

Traceback (most recent call last):

File "C:\python\_VSC\GUVI\_Projects\Project\_Clickstream\_Conversion\main\_pipeline.py", line 30, in run\_main\_pipeline

best\_regression\_model, regression\_metrics = regression\_pipeline(train\_data, test\_data)

^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

ValueError: too many values to unpack (expected 2)

**Solution**:

**Issue:**

Your pipeline function regression\_pipeline(train\_data\_path, test\_data\_path) **is a ZenML pipeline decorator function**, which means **it does not return values like a normal function** when called. Instead, it **creates a pipeline run object**.

So, when you do:

“best\_regression\_model, regression\_metrics = regression\_pipeline(train\_data, test\_data)”

you are expecting two return values, but regression\_pipeline(train\_data, test\_data) **actually returns a ZenML pipeline run execution** rather than your dummy return values.

**Solution:**

To properly retrieve outputs from a ZenML pipeline, **you need to access artifacts produced by steps** rather than expecting a return value.

**✅ Fix 1: Remove unpacking from regression\_pipeline**

Instead of trying to unpack the result directly, just call:

python

CopyEdit

result = regression\_pipeline(train\_data, test\_data)

print(result) # This prints the pipeline execution details, not actual model values.

Then, check the ZenML artifact store for actual output values.

**✅ Fix 2: Use ZenML's Step Caching to Retrieve Output**

To retrieve outputs from the regression\_pipeline, you need to fetch the step artifacts explicitly using ZenML's client API. You can do something like:

python

CopyEdit

from zenml.client import Client

# After pipeline runs

pipeline\_run = Client().get\_pipeline("regression\_pipeline").last\_successful\_run

if pipeline\_run:

best\_model\_output = pipeline\_run.get\_step("select\_best\_regression\_model").outputs

print(best\_model\_output)

This fetches the output from the select\_best\_regression\_model step, which is where your best model is selected.

**✅ Fix 3: Modify regression\_pipeline to Return Outputs as Artifacts**

ZenML does not allow direct return values, but you can log the outputs using step decorators: Modify select\_best\_regression\_model to save the best model:

python

CopyEdit

from zenml.steps import step

@step

def select\_best\_regression\_model(regression\_eval\_df):

best\_regression\_model = "DummyModel"

regression\_metrics = {"metric1": 0.95, "metric2": 0.80}

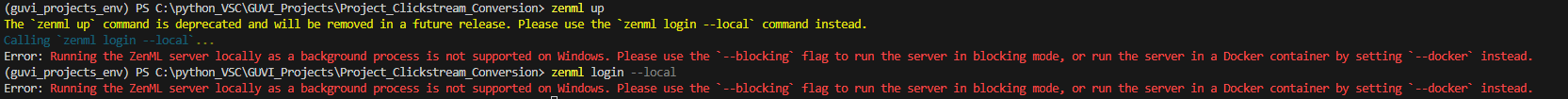
return best\_regression\_model, regression\_metrics # ZenML stores this in artifacts

Then retrieve it in main\_pipeline.py as shown in Fix 2.

**Summary:**

* **ZenML pipeline functions do not return values directly** like normal Python functions.
* You **cannot unpack return values** from regression\_pipeline(train\_data, test\_data).
* Instead, **access artifacts** using ZenML's Client API.

**Error 3:**



(guvi\_projects\_env) PS C:\python\_VSC\GUVI\_Projects\Project\_Clickstream\_Conversion> zenml up

The `zenml up` command is deprecated and will be removed in a future release. Please use the `zenml login --local` command instead.

Calling `zenml login --local`...

Error: Running the ZenML server locally as a background process is not supported on Windows. Please use the `--blocking` flag to run the server in blocking mode, or run the server in a Docker container by setting `--docker` instead.

(guvi\_projects\_env) PS C:\python\_VSC\GUVI\_Projects\Project\_Clickstream\_Conversion> zenml login --local

Error: Running the ZenML server locally as a background process is not supported on Windows. Please use the `--blocking` flag to run the server in blocking mode, or run the server in a Docker container by setting `--docker` instead.

**Solution:**

**Fixing ZenML Server Startup on Windows**

Since **zenml up is deprecated**, and **local mode doesn’t work on Windows**, you have two options:

**Option 1: Run ZenML Server in Blocking Mode**

Try running the following command:

zenml login --local --blocking

This will start the ZenML server **in blocking mode**, meaning it will keep running in the terminal.  
If this works, **open a new terminal** and run:

zenml run train\_pipeline --data\_path="data/Train\_data.csv"

Zenml Login credentials –

“zenml login --local –blocking” this command provides a server link, click it a web page will open to login use the below mentioned password

*username – default*

*email – default@gmail.com*