



**National University**  
of computer and emerging sciences

## Pcn Report

CS422 Human Computer Interaction

Semester Project

name

Section

*Submitted to:*

Department of Computer Science BS(CS)

FAST-NUCES Islamabad

---

## Task 2: Experiment with parallel execution and stateful computations

We are setting up an ML model by downloading the necessary libraries and running it on our systems.

We installed the following libraries to set up the model.

- pandas numpy scikit-learn tensorflow

To execute model

### Issues:

We had issues like python version clashes. To install numpy tensorflow and ray, different versions of python were needed. Despite setting the virtual environment I was having issues.

Created the following requirement file where i mentioned the version of pandas numpy scikit-learn tensorflow, ray compatible with python 3.9.0

**pandas: 2.2.2   numpy: 1.23.5   keras: 2.12.0   scikit-learn: 1.5.0   tensorflow: 2.12.0**

It was showing some unhashable list error in the code however after installing the tensorflow version **keras: 2.12.0**. I previously had version 2.4 of keras.

### Comparative Analysis:

I ran the simple model without RAY and the same model with RAY.

### Ray:

Note: Then I installed the ray tune and serve libraries to avoid any further issues.

I first initialized the ray using, `.init()`. This way it would create a cluster.

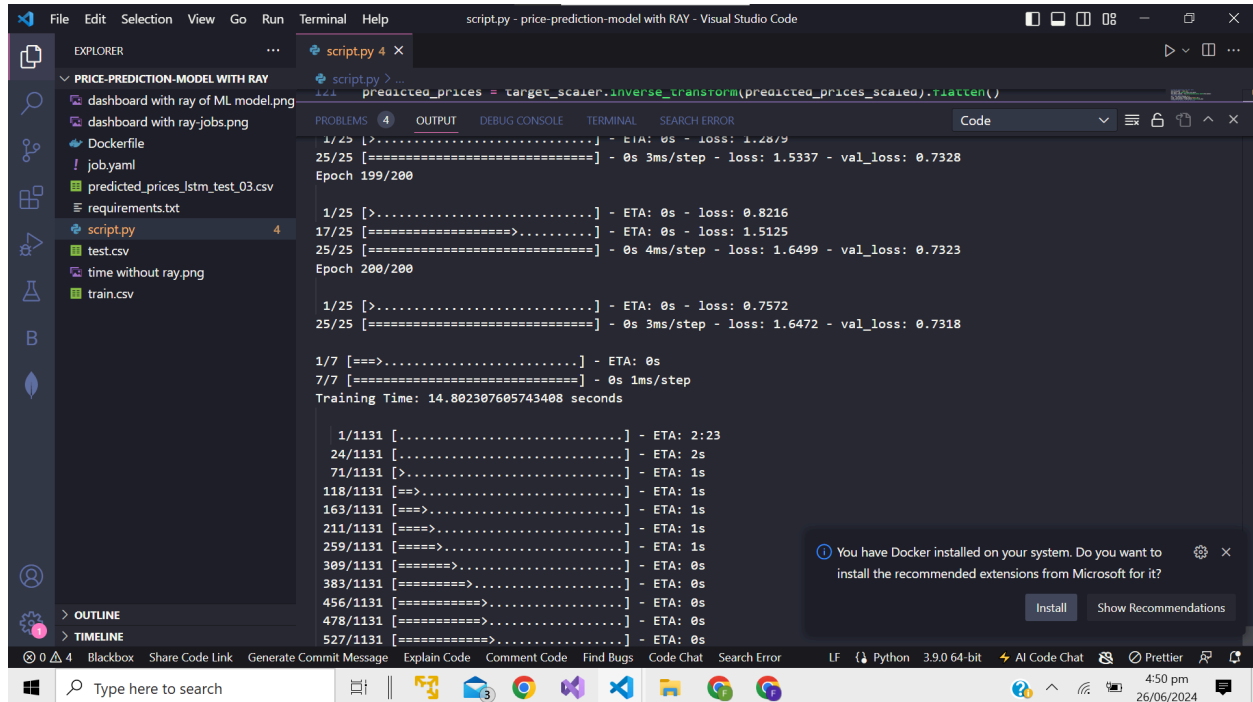
Then I used the remote decorator `@ray.remote` on the `data_processing` function to enable that function to be executed as a task on a separate worker process which means that the function will not run on the main Python process but will be offloaded to another process managed by the Ray framework.

```
train_data_cleaned = ray.get(preprocess_data.remote(train_data))
```

`reprocess_data.remote(train_data)` schedules the execution of the `preprocess_data` function remotely on a Ray worker process.

**Result Retrieval:** `ray.get(...)` is a blocking call that waits for the completion of the remote task (`preprocess_data`) and retrieves its result.

Below is the comparative analysis of the model with and without Ray.



```
script.py - price-prediction-model with RAY - Visual Studio Code
predicted_prices = target_scaler.inverse_transform(predicted_prices_scaled).ravel()

1/25 [>.....] - ETA: 0s - loss: 1.2879
25/25 [=====] - 0s 3ms/step - loss: 1.5337 - val_loss: 0.7328
Epoch 199/200

1/25 [>.....] - ETA: 0s - loss: 0.8216
17/25 [=====] - ETA: 0s - loss: 1.5125
25/25 [=====] - 0s 4ms/step - loss: 1.6499 - val_loss: 0.7323
Epoch 200/200

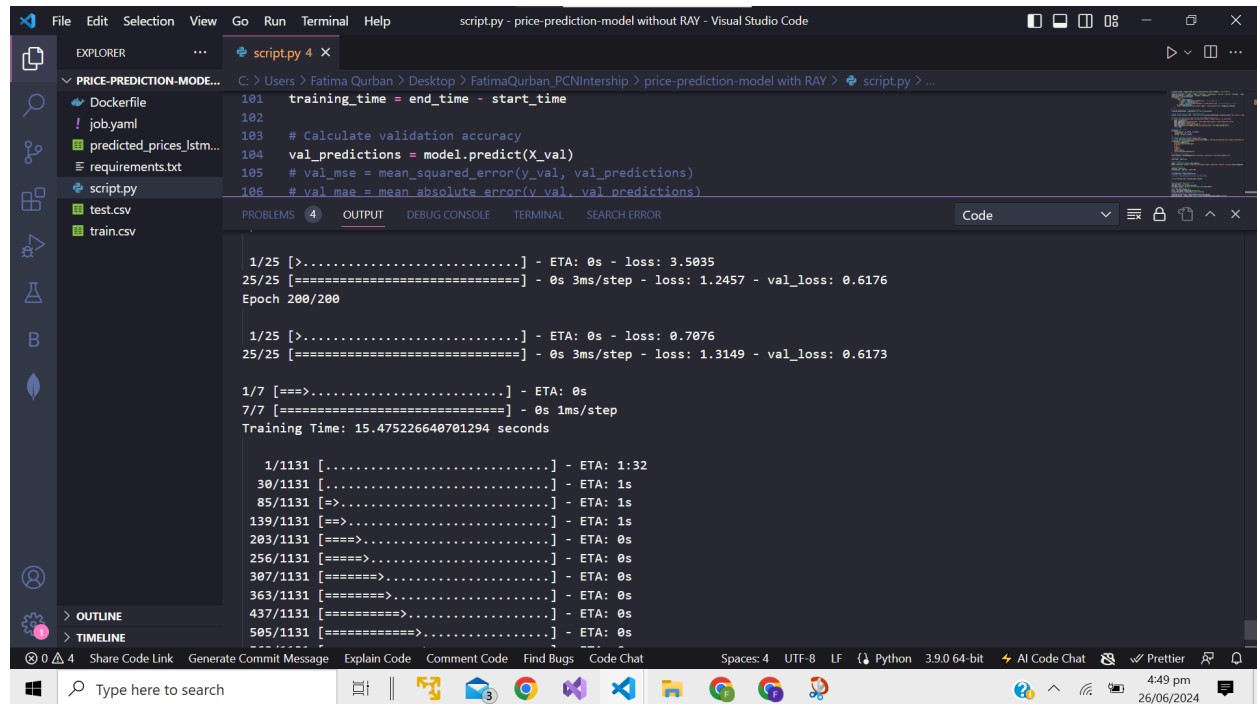
1/25 [>.....] - ETA: 0s - loss: 0.7572
25/25 [=====] - 0s 3ms/step - loss: 1.6472 - val_loss: 0.7318

1/7 [====>.....] - ETA: 0s
7/7 [=====] - 0s 1ms/step
Training Time: 14.802307605743408 seconds

1/1131 [.....] - ETA: 2:23
24/1131 [.....] - ETA: 2s
71/1131 [>.....] - ETA: 1s
118/1131 [==>.....] - ETA: 1s
163/1131 [====>.....] - ETA: 1s
211/1131 [====>.....] - ETA: 1s
259/1131 [====>.....] - ETA: 1s
309/1131 [====>.....] - ETA: 0s
383/1131 [====>.....] - ETA: 0s
456/1131 [====>.....] - ETA: 0s
478/1131 [====>.....] - ETA: 0s
527/1131 [====>.....] - ETA: 0s
```

You have Docker installed on your system. Do you want to install the recommended extensions from Microsoft for it?

Install Show Recommendations



```
101 training_time = end_time - start_time
102
103 # Calculate validation accuracy
104 val_predictions = model.predict(X_val)
105 # val_mse = mean_squared_error(y_val, val_predictions)
106 # val_mae = mean_absolute_error(y_val, val_predictions)
```

1/25 [>.....] - ETA: 0s - loss: 3.5035  
25/25 [=====] - 0s 3ms/step - loss: 1.2457 - val\_loss: 0.6176  
Epoch 200/200

1/25 [>.....] - ETA: 0s - loss: 0.7076  
25/25 [=====] - 0s 3ms/step - loss: 1.3149 - val\_loss: 0.6173

1/7 [==>.....] - ETA: 0s  
7/7 [=====] - 0s 1ms/step  
Training Time: 15.475226640701294 seconds

1/1131 [.....] - ETA: 1:32  
30/1131 [.....] - ETA: 1s  
85/1131 [=>.....] - ETA: 1s  
139/1131 [==>.....] - ETA: 1s  
203/1131 [====>.....] - ETA: 0s  
256/1131 [=====>.....] - ETA: 0s  
307/1131 [=====>.....] - ETA: 0s  
363/1131 [=====.....] - ETA: 0s  
437/1131 [=====.....] - ETA: 0s  
505/1131 [=====.....] - ETA: 0s

With Ray we have less time. Although it should have been less. But we notice a difference of 1 sec.