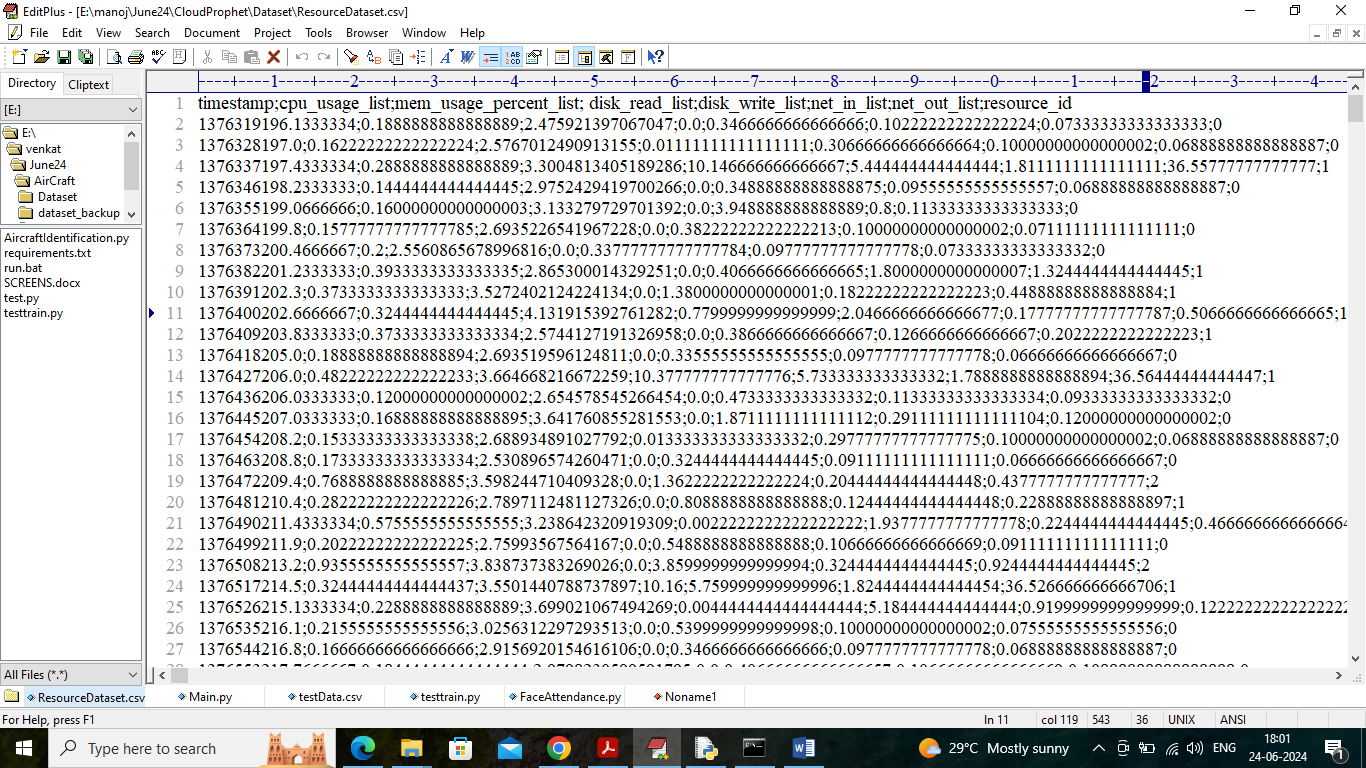
CloudProphet: A Machine Learning-Based Performance Prediction for Public Clouds

Now-a-days cloud services are gaining lots of popularity because of providing heavy computation and storage at cheaper cost. Across world many peoples will access cloud services and it’s mandatory to utilize all cloud resources efficiently to manage all requests from all clients. All existing cloud services are utilizing VM scheduling algorithms to allocate resources efficiently but all those scheduling algorithms required accurate performance prediction of cloud resources. So author of this paper introducing novel concept called ‘CloudProphet’ which can achieve high-accuracy predictions for black-box VMs, the proposed method first identifies the running application inside the virtual machine. It then selects highly correlated runtime metrics as the input of the machine learning approach to accurately predict the performance level of the cloud application. Experimental results with state-of-the-art cloud benchmarks demonstrate that our proposed method outperforms existing prediction methods in terms of the worst prediction error compare to existing algorithm. In addition, successfully tackle the challenge of performance prediction for applications with variable workloads by introducing the performance degradation index, which other comparison methods fail to consider. The workflow versatility of the proposed approach has been verified with different modern servers and VM configurations.

In propose algorithm author has generated his own dataset with two different servers and multiple VM’s with different CPU and memory sizes but that dataset was not publish on internet so we utilize VM resource processing dataset from GITHUB. In below screen showing dataset details



In above dataset screen first row contains dataset column names and remaining rows contains dataset values.

To predict VM best or performance degradation author has used 3 different modules

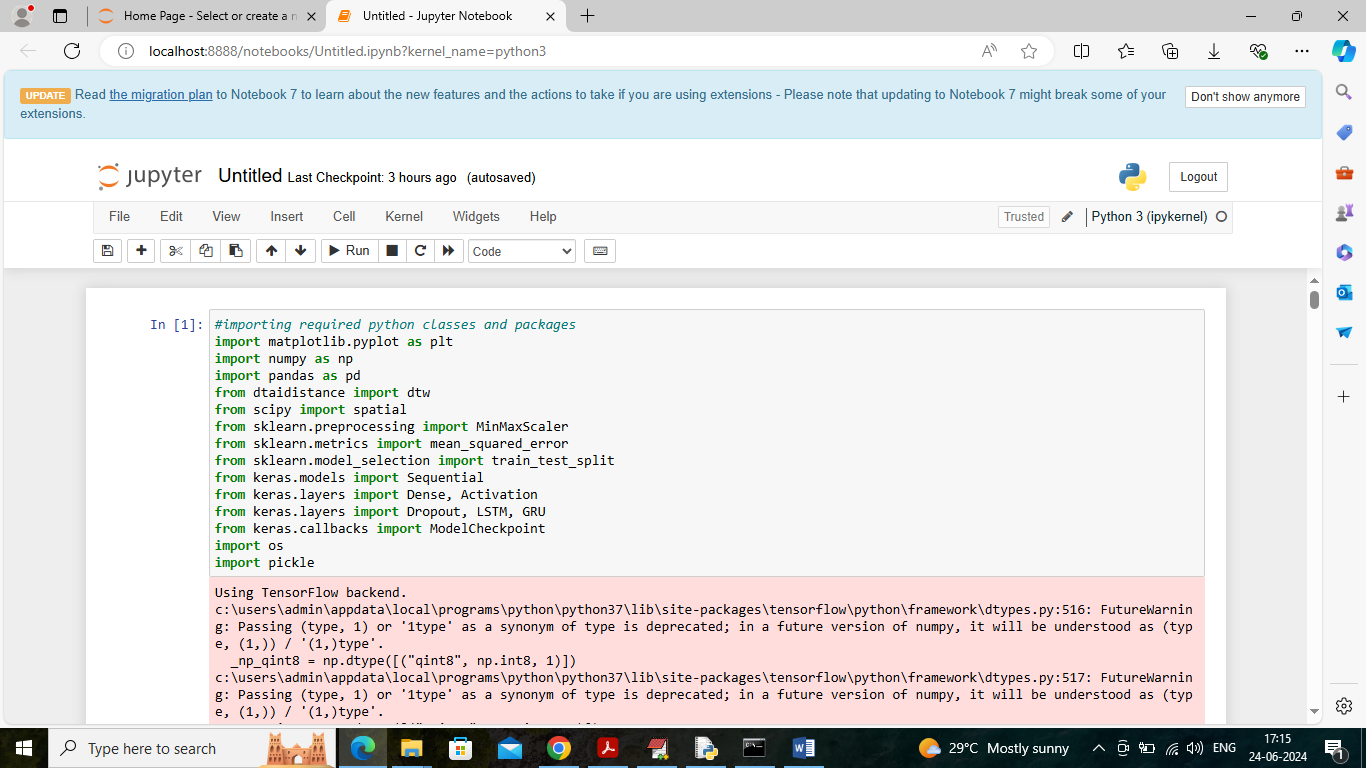
1. Application Type Identification: different requests required different resources so author taking reference dataset and test data to apply Dynamic Time Wrapping (DTW) algorithm which removes temporal or time based mismatch values from reference and test data to select close matching values and then employ Euclidean Distance function between reference and test data to match reference application type to test data. Using application type we can identify or select suitable VM for processing.
2. Highly Correlated Metrics: PEARSON correlation formula will be applied on reference dataset to select all attributes or metrics which are highly correlated or similar. Highly correlated metrics will show us which CPU usage matching with which RAM size so based on this dependency Neural Network can predict Cloud VM performance.
3. Neural Network: selected highly correlated metrics will be input to neural networks algorithm to predict VM performance. If predicted variable work load greater than 100% then VM required another processor otherwise performance will be degraded and if predicted work load less than 100% then VM is in best performance.

Extension Concept1: in propose work author using traditional algorithms like CNN, ANN and LSTM to predict cloud performance so as extension we have used lightweight and more sophisticated algorithm like GRU (gated recurrent unit) which is giving better accuracy compare to propose algorithms.

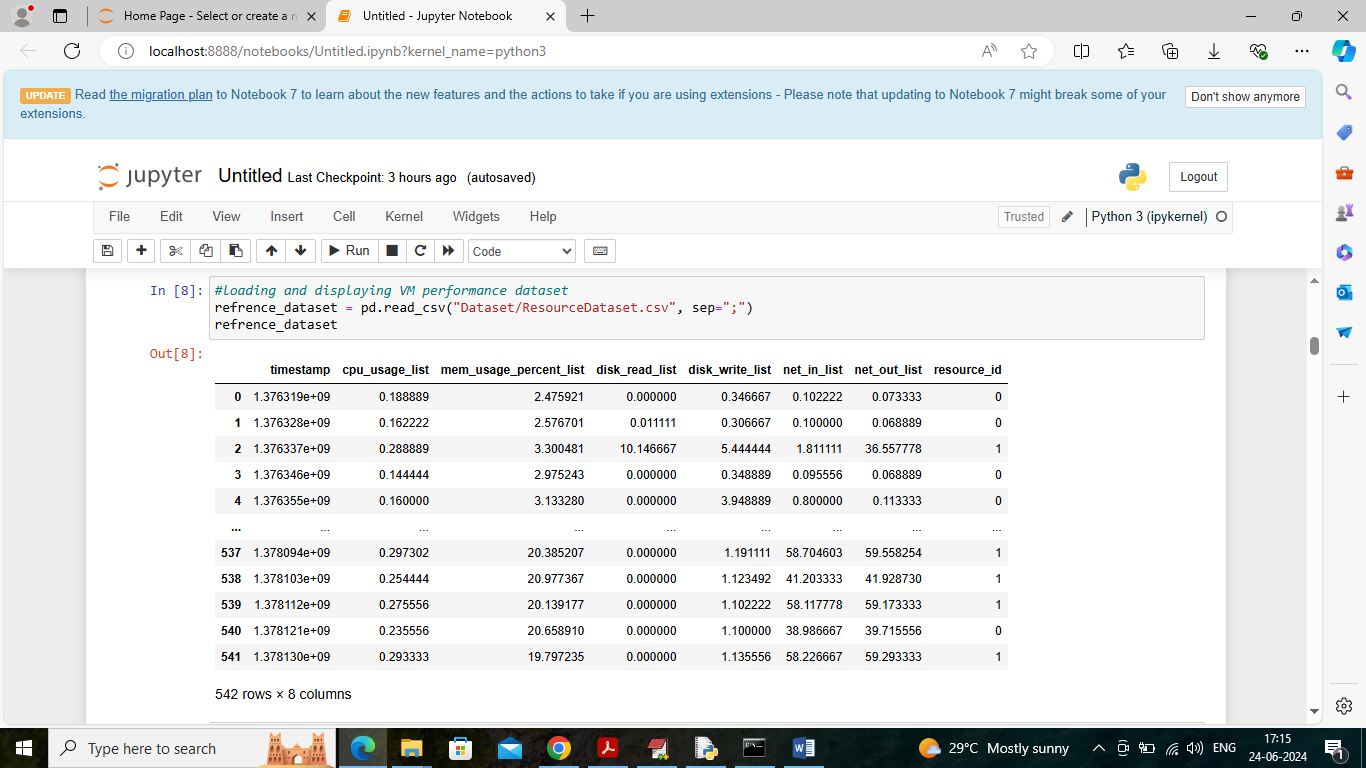
Extension Concept2: in propose paper author is just giving performance evaluation without any live dataset testing to predict APPLICATION Type and VM performance so as extension 2 we have taken live dataset and then applied DTW algorithm, highly correlated metric selection and then employed extension GRU algorithm on that live test data to predict APPLICATION Type and performance type as BEST or Degradation.

SCREEN SHOTS

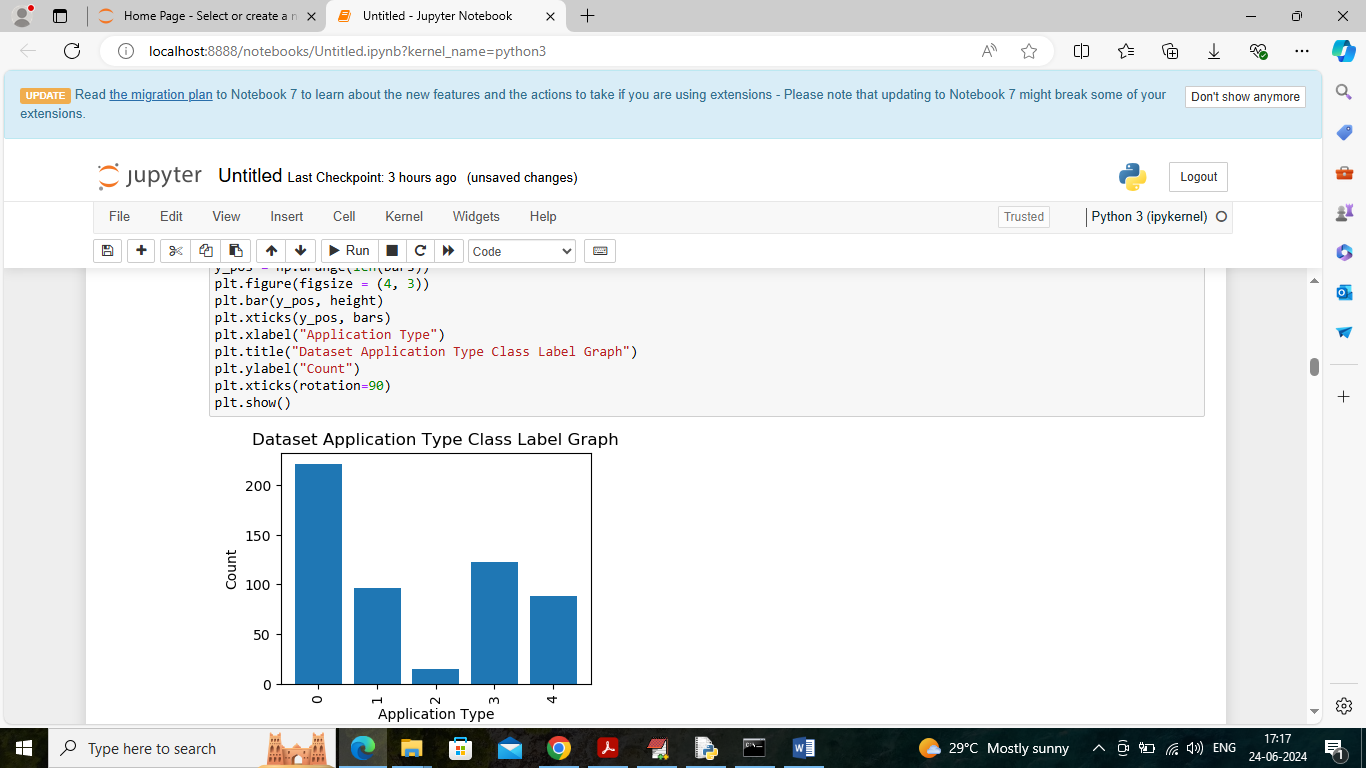
This application using Cloud dataset with Deep learning algorithms so for coding we have used JUPYTER notebook and below are the code and output screens with blue colour comments.



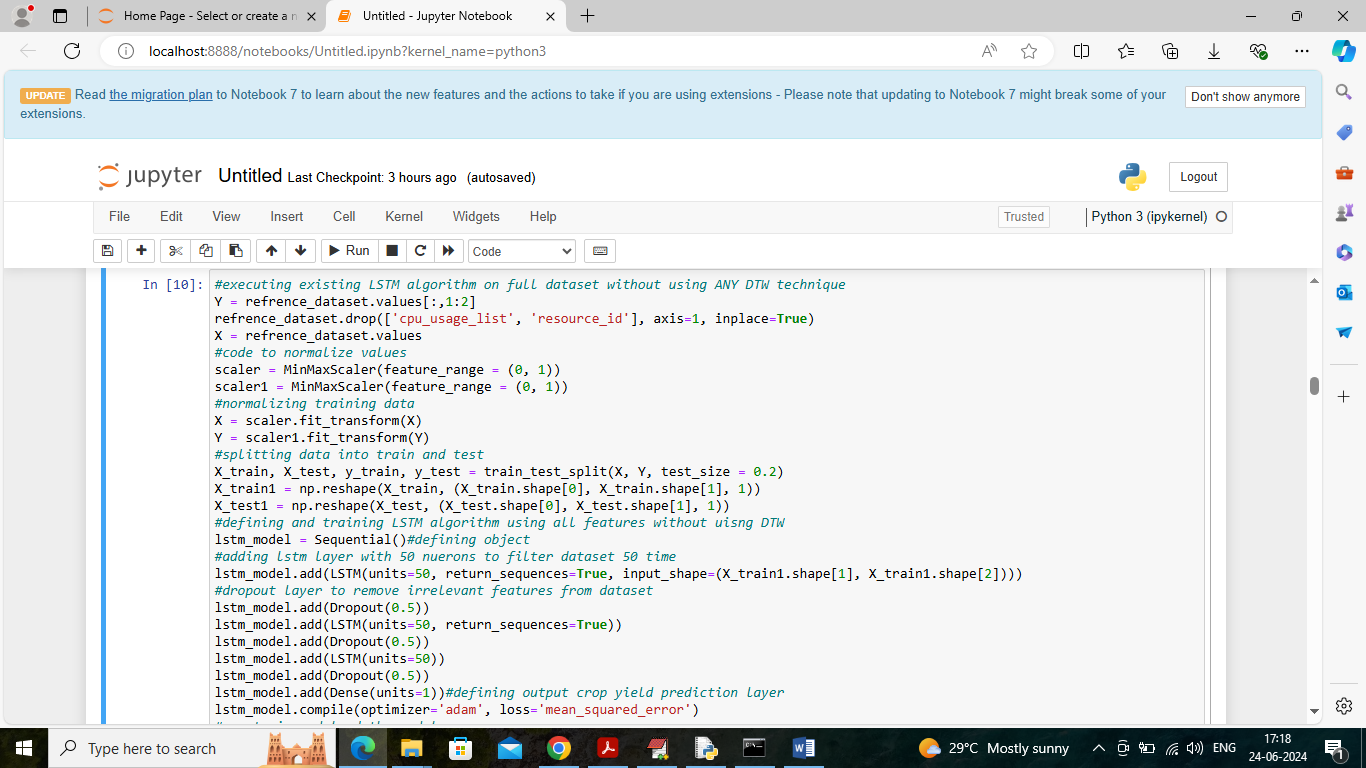
In above screen importing required python classes and packages



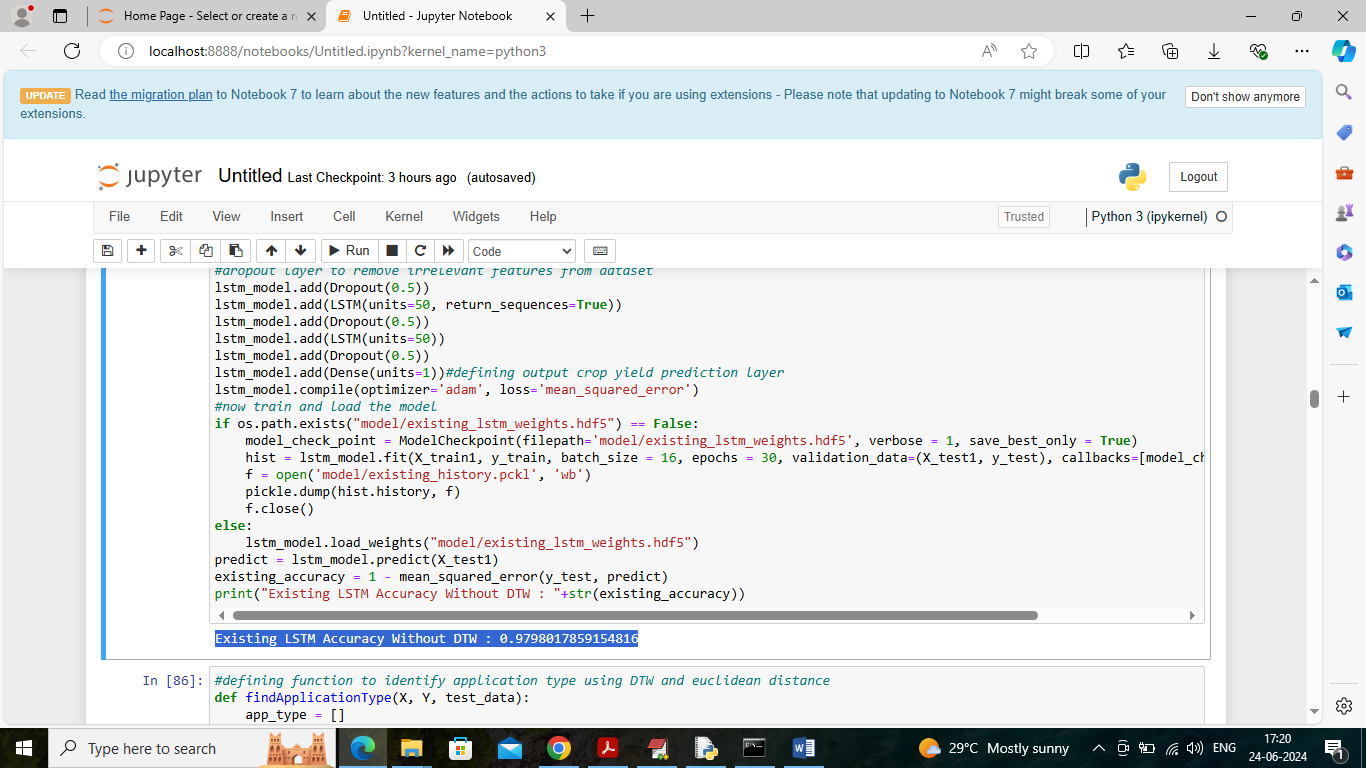
In above screen loading and displaying refrence dataset values



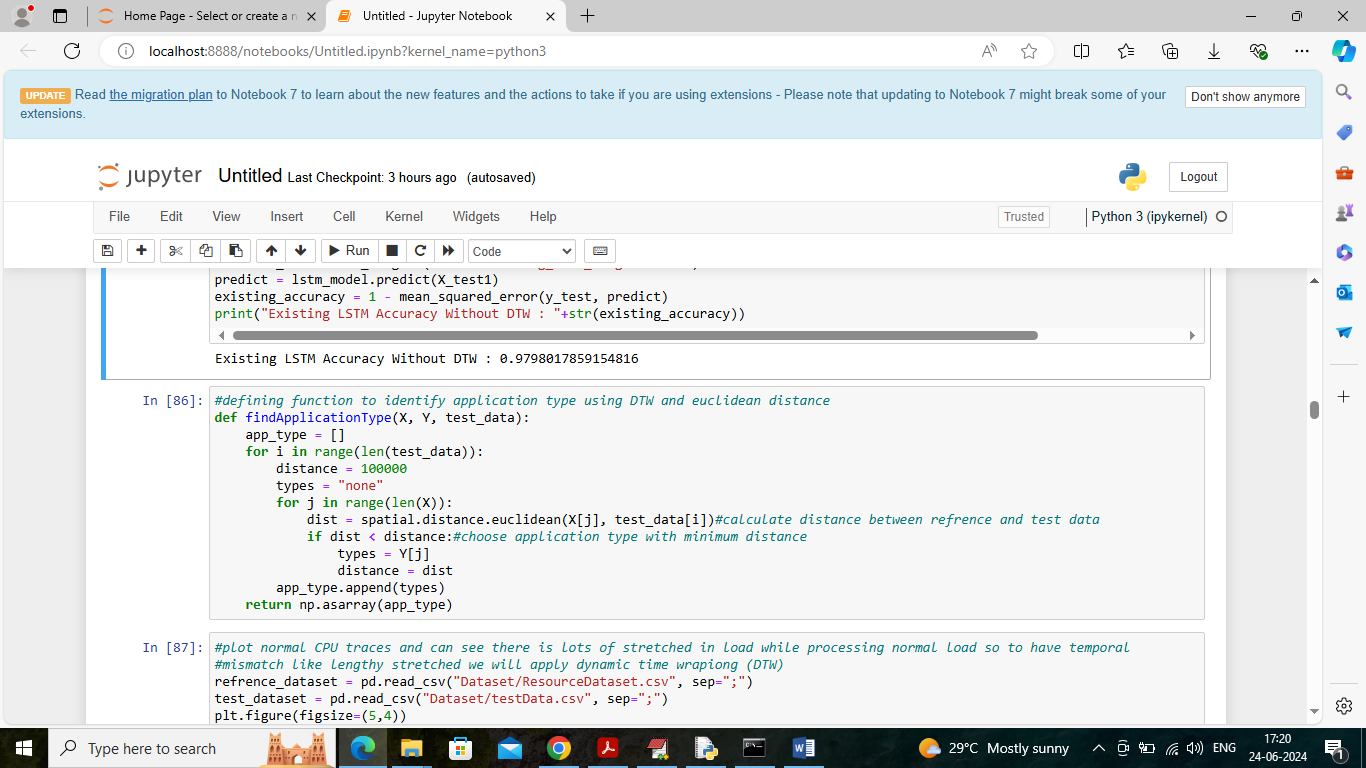
In above screen visualizing graph of different applications types where x-axis represents ‘type of application’ and y-axis represents number of instances



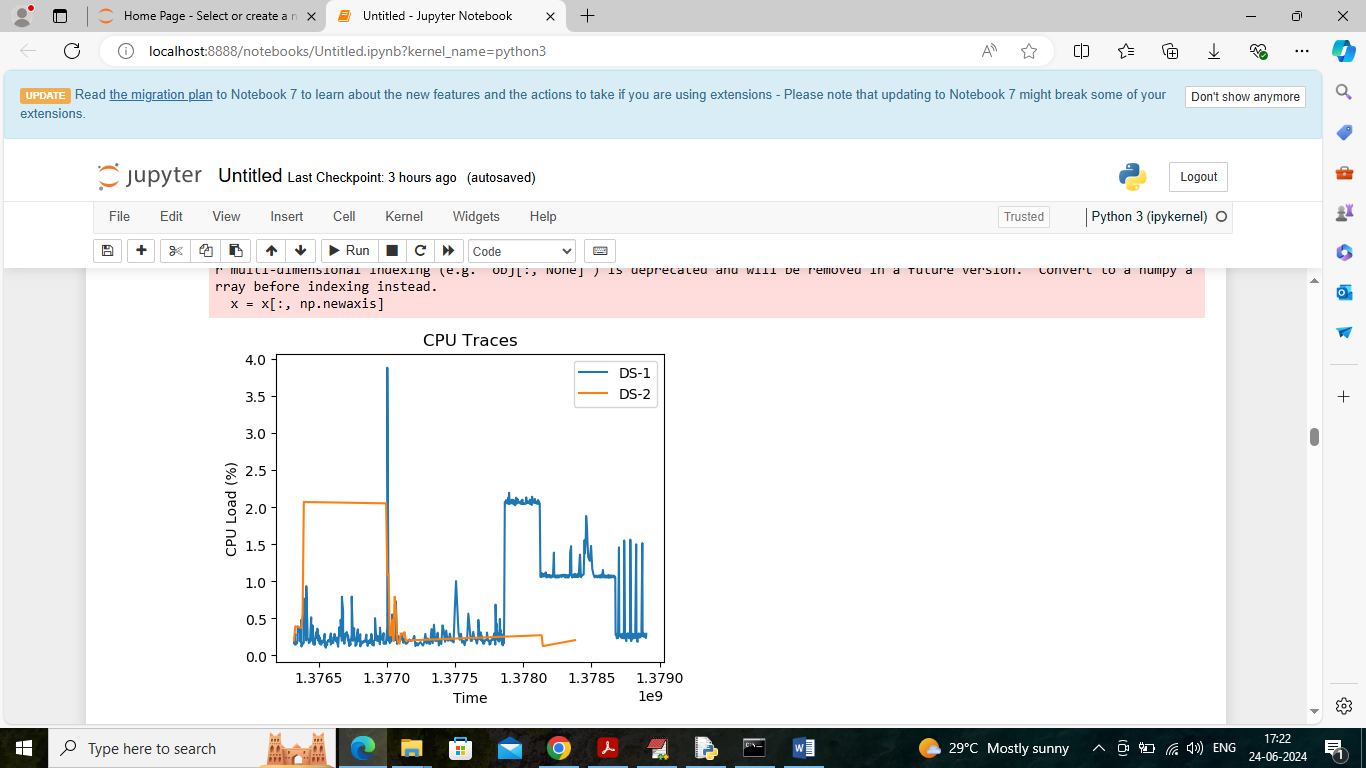
In above screen training existing LSTM algorithm without using any DTW and highly correlated metrics and after training this algorithm will get below accuracy



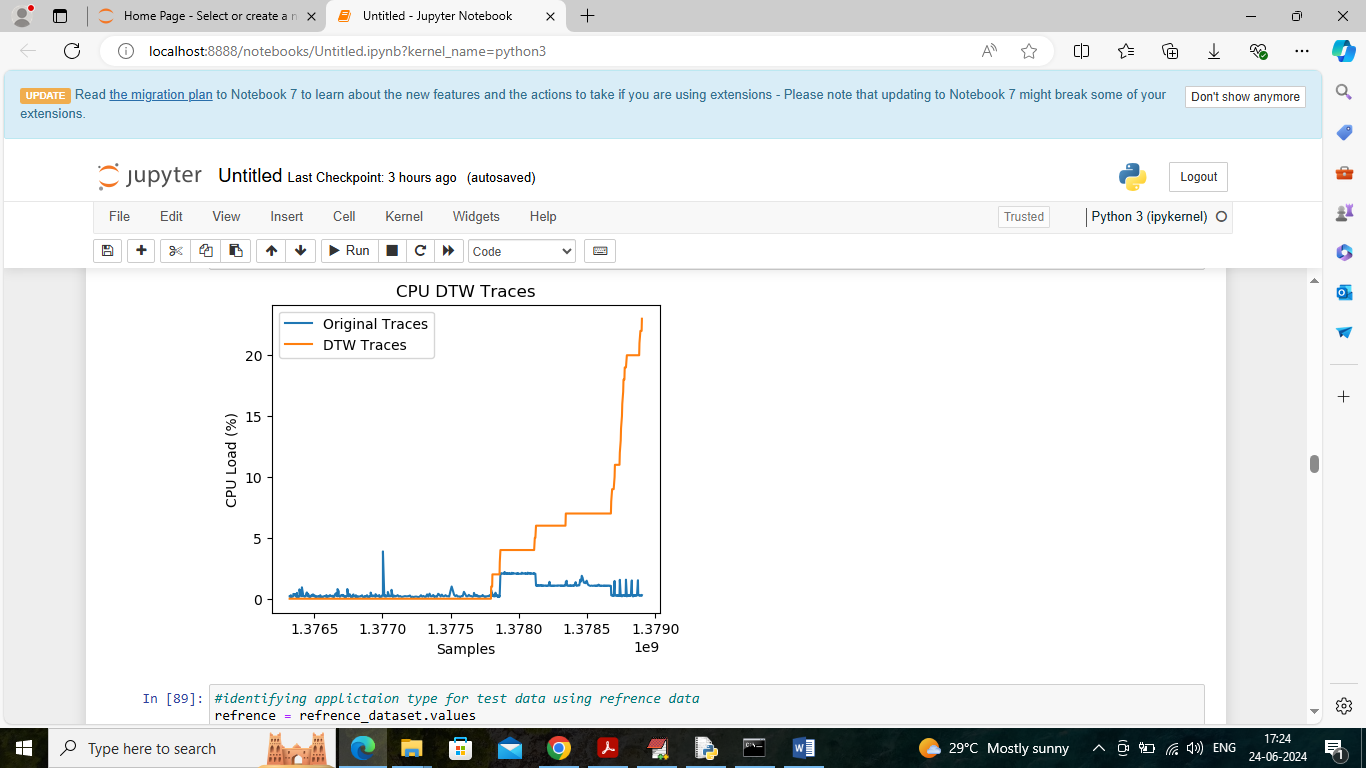
In above screen in blue colour text can see LSTM with existing technique got 97% accuracy



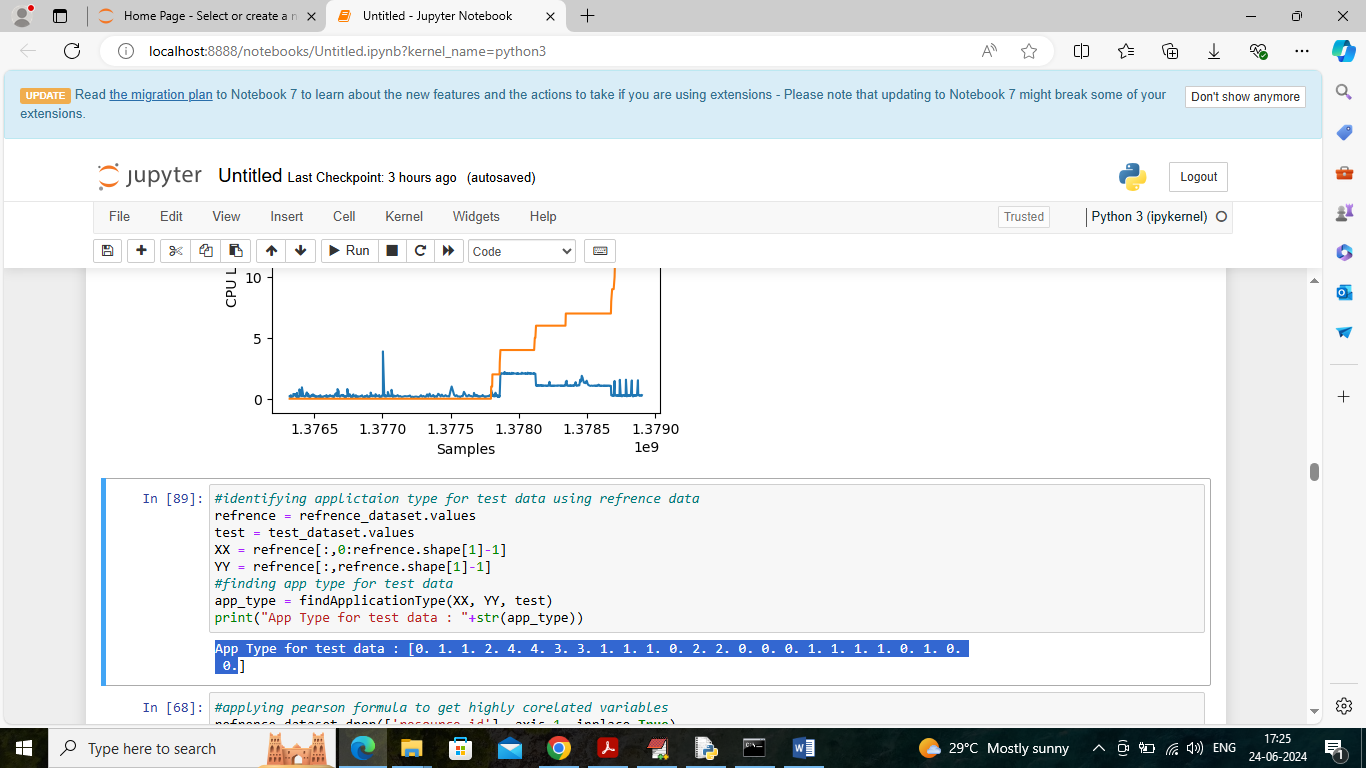
In above screen defining function to calculate application type using Euclidean distance function on reference and test dataset



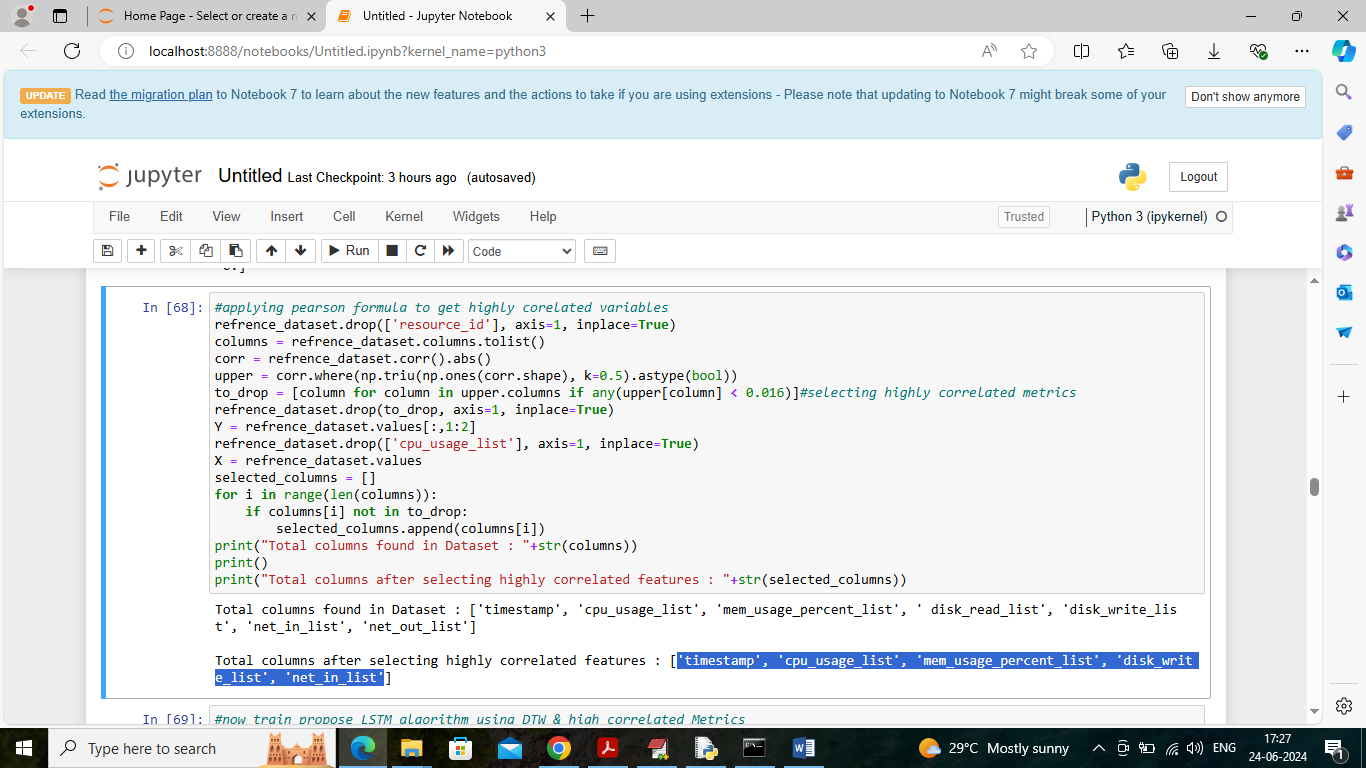
In above screen plotting graph of Data Serving (DS) for same application where DS1 is for one request serving and other for second request serving. In above graph two different lines like orange and blue refers to two data serving and can see major scatter between two data serving (difference in values at different times) so we will apply DTW technique to remove temporal mismatch and select only close values and in below graph showing close values selection using DTW



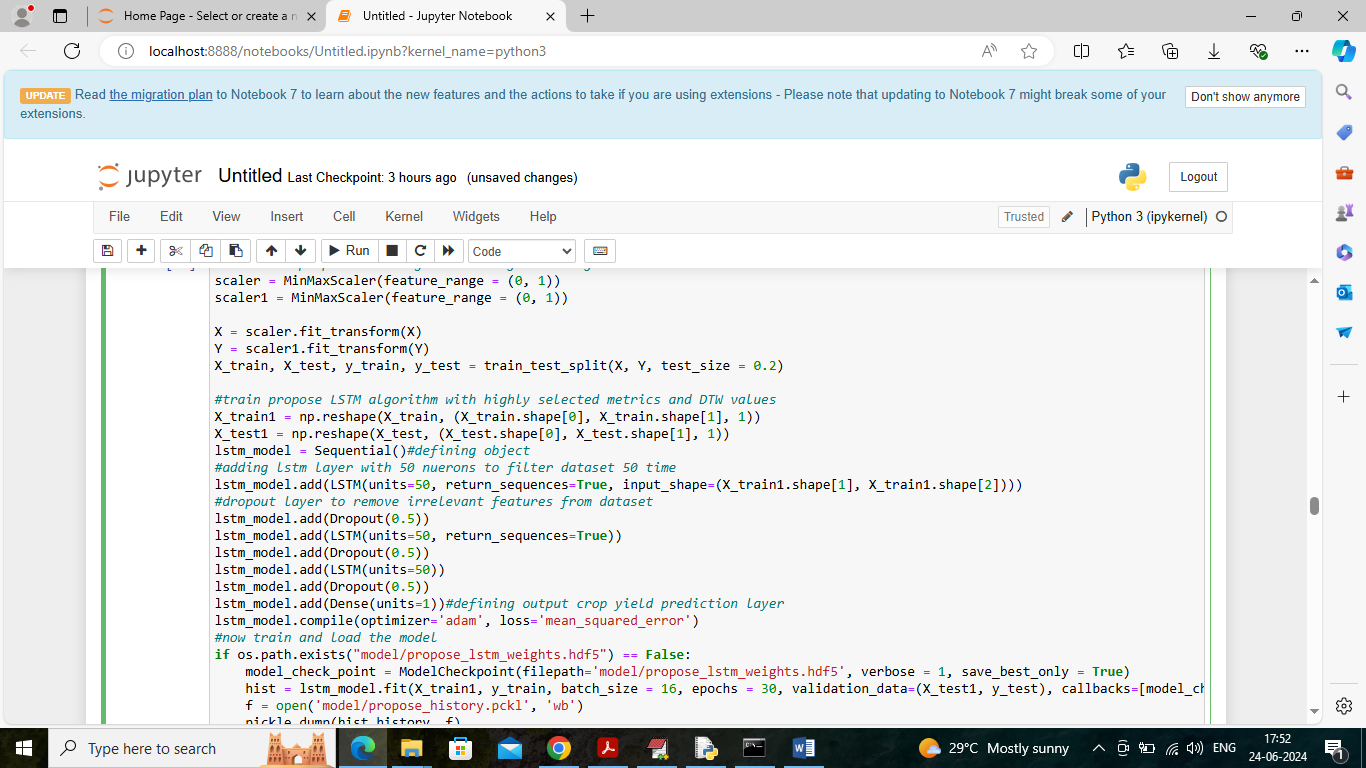
In above DTW graph can see both DS1 and DS2 traces are closed as both lines are fully overlapping initially.



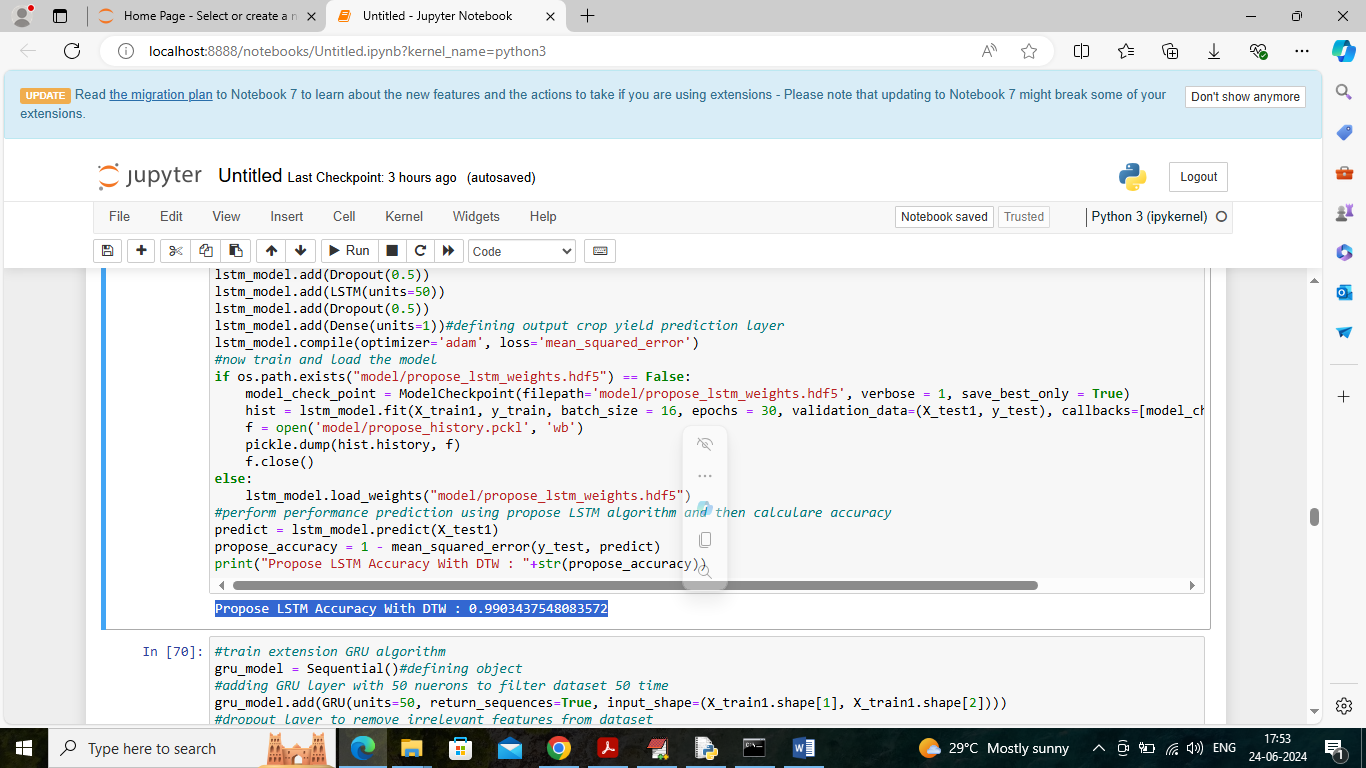
In above screen for each test data we are identifying ‘Application Type’ and in blue colour text can see application identity values like 0, 1, 2 etc. All above test data values are loading from ‘Dataset/testData.csv’ file



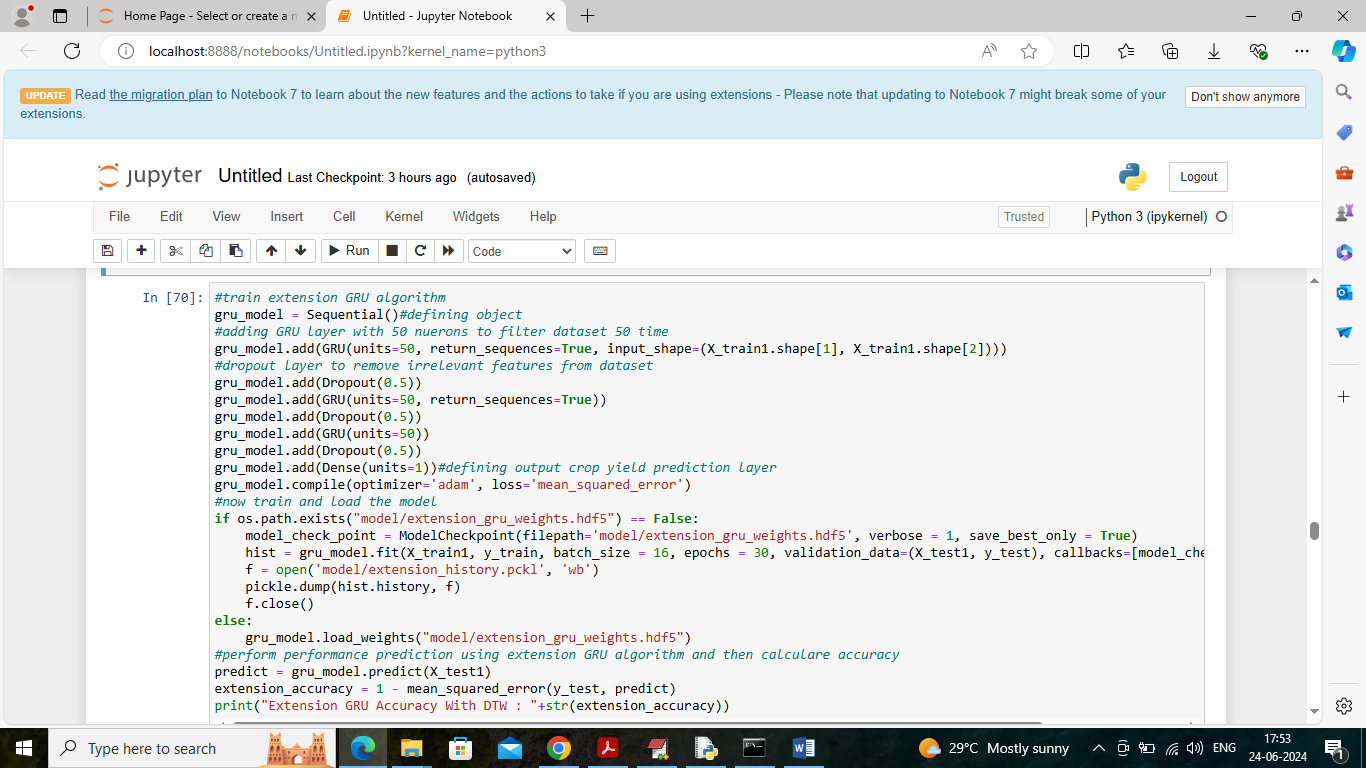
In above screen applying “pearson correlation’ algorithm to select highly correlated metrics and in first line can see all attributes or metrics available in dataset and then printing ‘selected highly correlated metrics’.



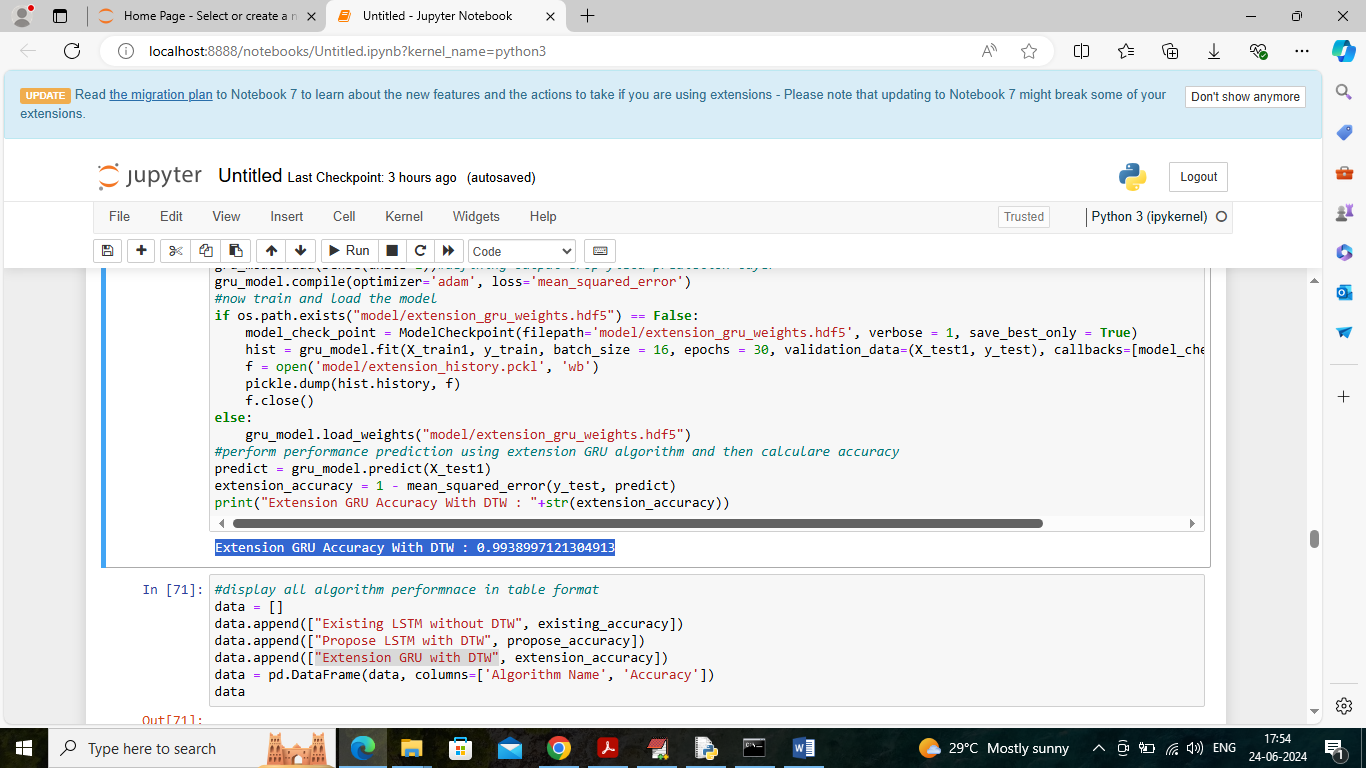
In above screen training propose LSTM with highly selected and DTW metrics and after training will get below accuracy



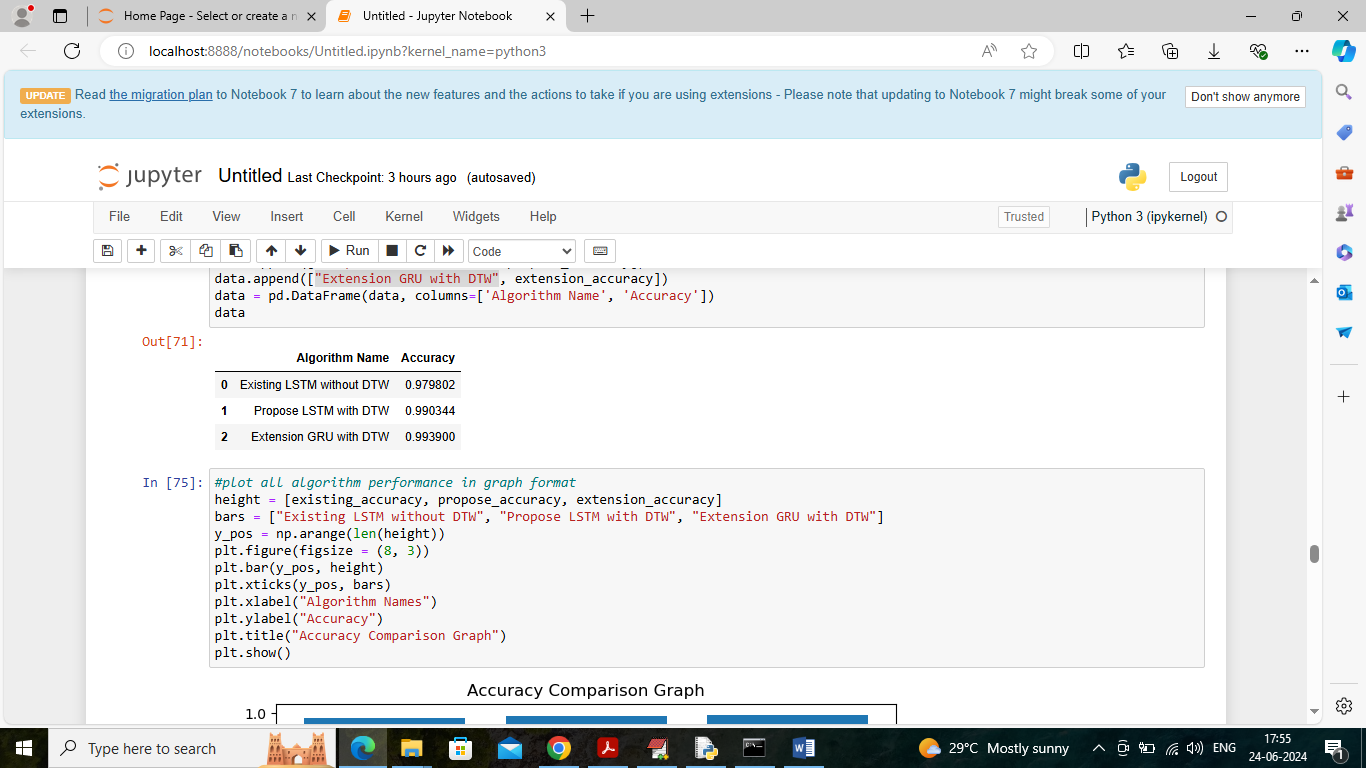
In above screen in blue colour text can see propose LSTM got 99% accuyracy



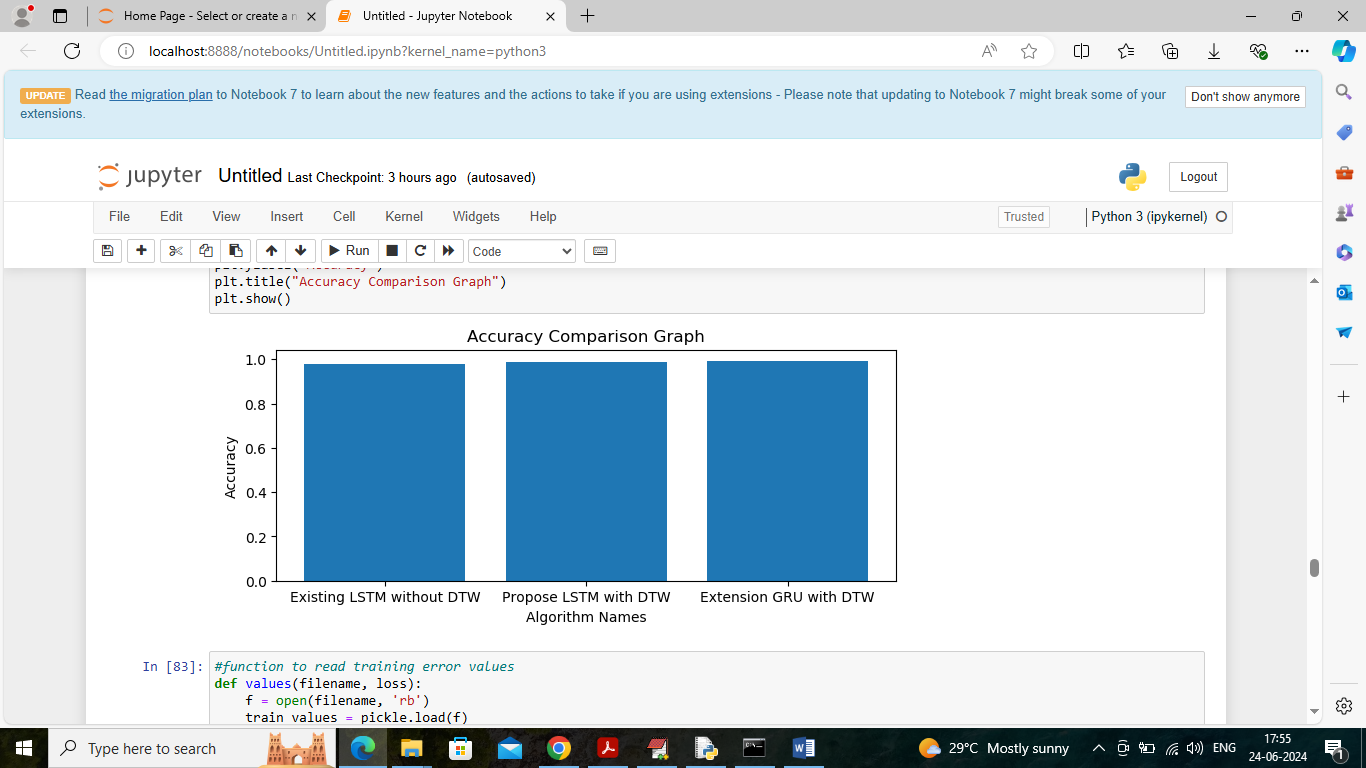
In above screen training extension GRU algorithm with DTW and highly correlated metrics and after training will get below output



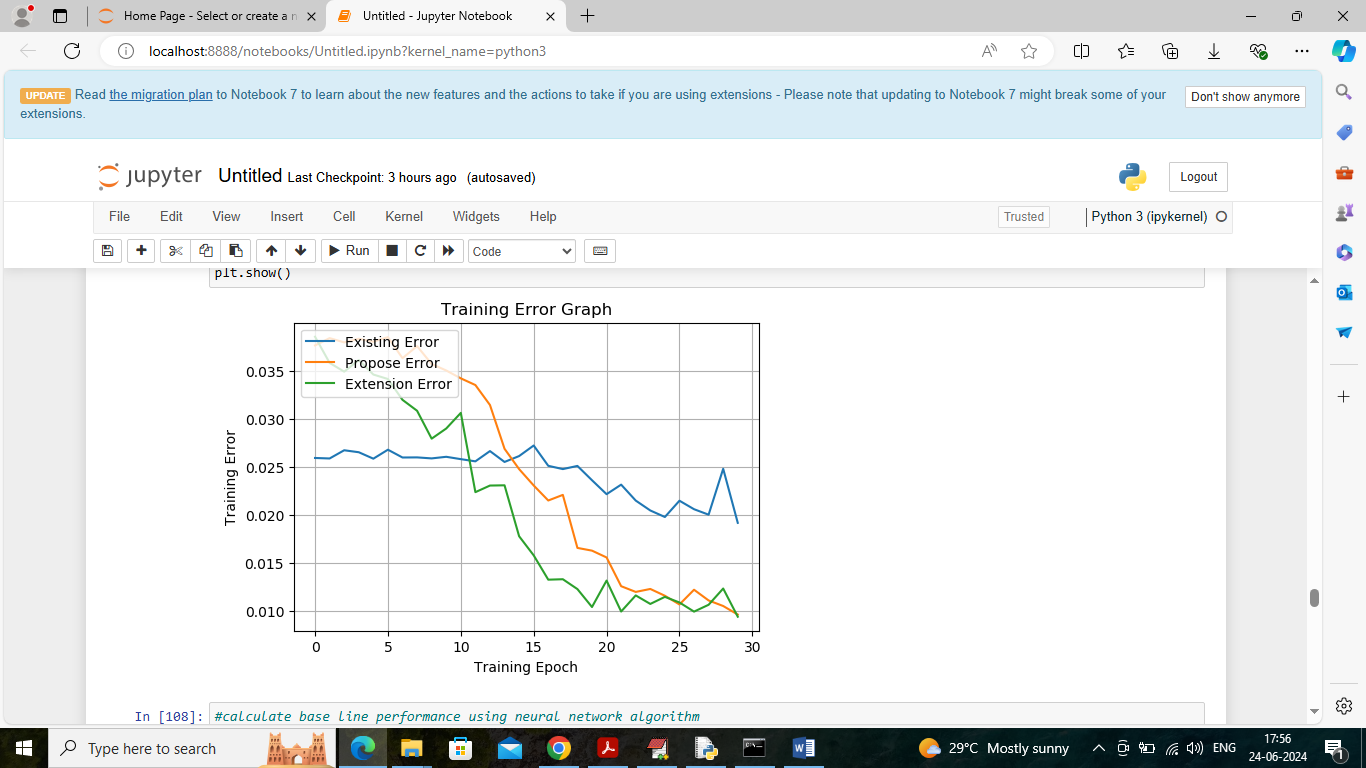
In above screen extension got 0.9938% accuracy which is higher than propose algorithm



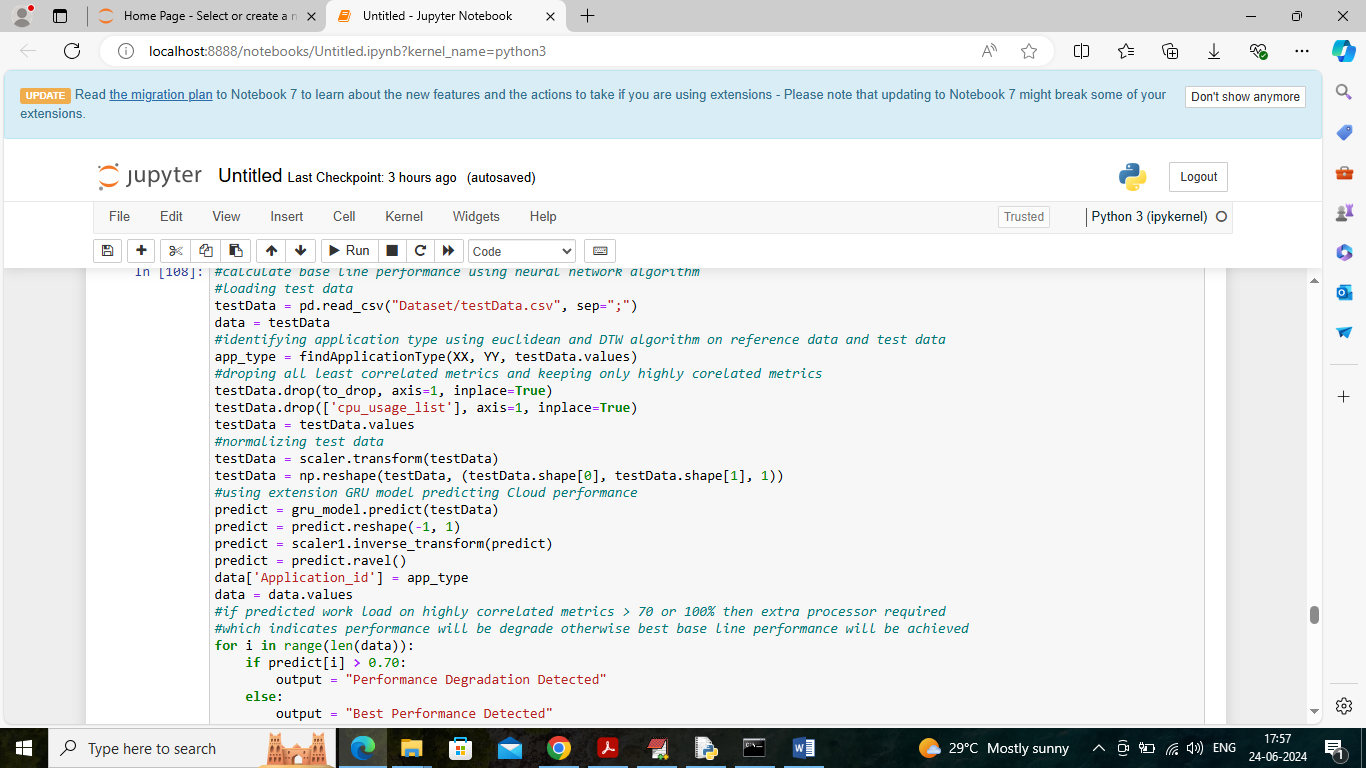
In above screen displaying all algorithm performance in tabular format



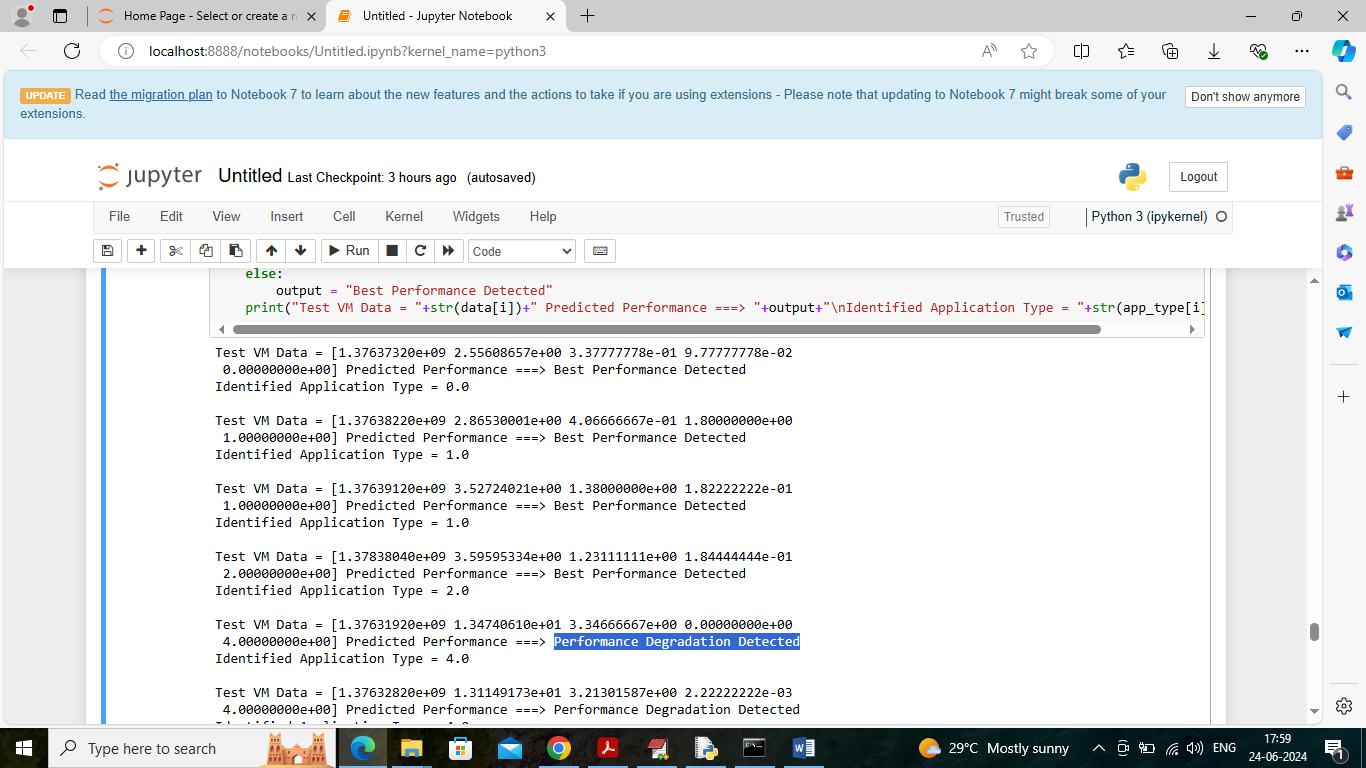
In above screen displaying all algorithm performance in graph format where x-axis represents algorithm names and y-axis represents accuracy



In above graph displaying training error graph for all algorithms where x-axis represents training epochs and y-axis represents ERROR and different line represents different algorithms and green line for extension got less error



In above screen as extension 2 reading live test data and then applying all algorithms to predict cloud performance and in above code read blue colour comments to know about each processing and after predicting on above test data will get below output



In above screen in square bracket can see LIVE test data values and then after =🡺 arrow symbol can see predicted performance and then in next line can see ‘Identified application type. We will see predicted performance for all test data values