

MAP 573 - Group 12

Time Series Forecast using (LSTM) RNN network

Data Visualization

Energy Forecast Data Visualisation

Time Series

3D Time Series

Statistical Analysis

Time Series Select

- ☒ Zone 1 Load ☐ Zone 2 Load ☐ Zone 3 Load
☐ Zone 4 Load ☐ Zone 5 Load ☐ Zone 6 Load
☐ Zone 7 Load ☐ Zone 8 Load ☐ Zone 9 Load
☐ Zone 10 Load ☐ Zone 11 Load ☐ Zone 12 Load
☐ Zone 13 Load ☐ Zone 14 Load ☐ Zone 15 Load
☐ Zone 16 Load ☐ Zone 17 Load ☒ Zone 18 Load
☐ Zone 19 Load ☐ Zone 20 Load ☒ St. 1 Temp.
☐ St. 2 Temp. ☐ St. 3 Temp. ☐ St. 4 Temp.
☐ St. 5 Temp. ☐ St. 6 Temp. ☐ St. 7 Temp.
☐ St. 8 Temp. ☐ St. 9 Temp. ☐ St. 10 Temp.
☐ St. 11 Temp.

Forecast Data

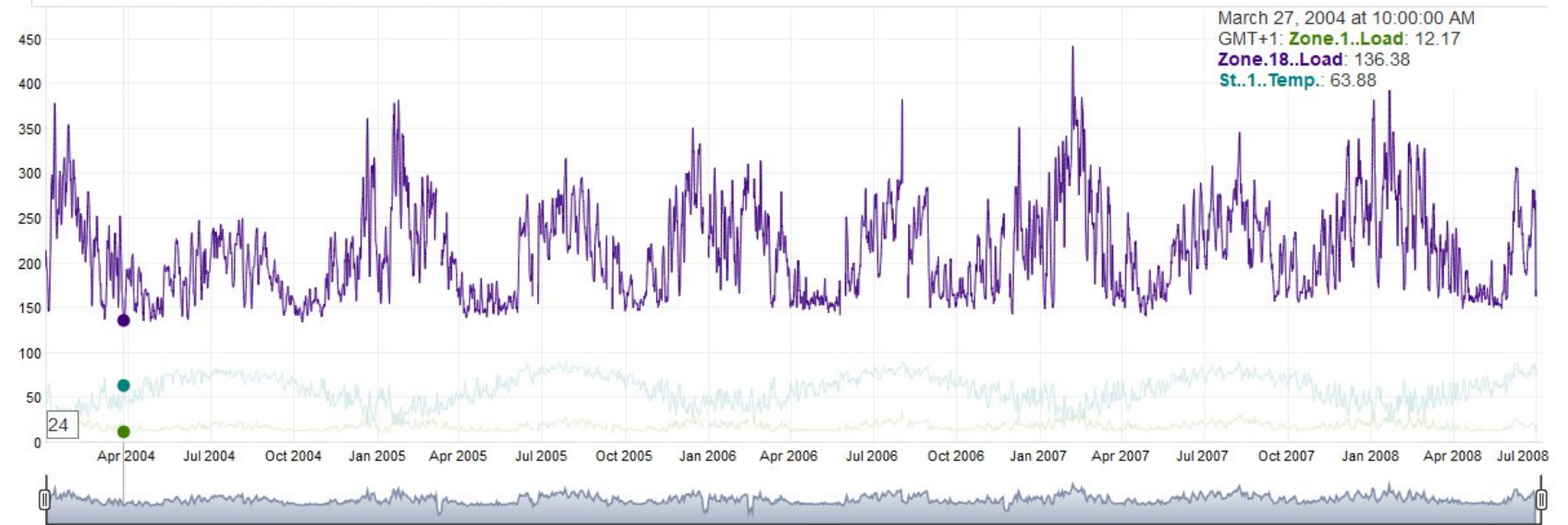
- ☐ Zone 1 Forecast ☐ Zone 2 Forecast ☐ Zone 3 Forecast
☐ Zone 4 Forecast ☐ Zone 5 Forecast
☐ Zone 6 Forecast ☐ Zone 7 Forecast
☐ Zone 8 Forecast ☐ Zone 9 Forecast
☐ Zone 10 Forecast ☐ Zone 11 Forecast
☐ Zone 12 Forecast ☐ Zone 13 Forecast
☐ Zone 14 Forecast ☐ Zone 15 Forecast
☐ Zone 16 Forecast ☐ Zone 17 Forecast
☐ Zone 18 Forecast ☐ Zone 19 Forecast
☐ Zone 20 Forecast

Plot

Summary Plot

Summary Load

Summary Temp



Data Visualization

Energy Forecast Data Visualisation

Time Series

3D Time Series

Statistical Analysis

Date range

2004-01-01

to

2008-06-30

Data source (Zones,stations)

Zone 1 Load

Periodicity

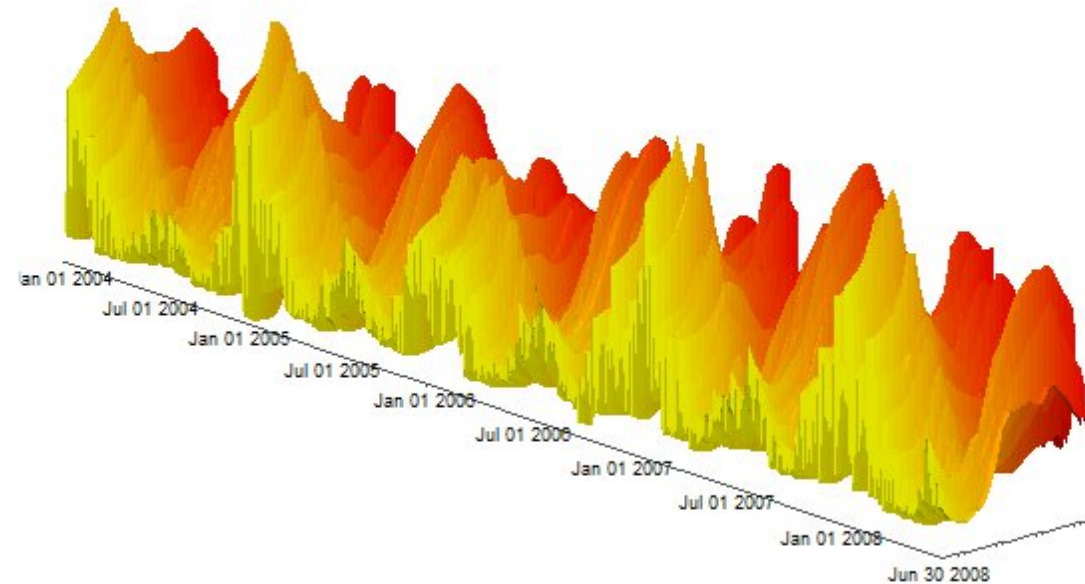
24

☐ Include Forecast Data (if it exists)

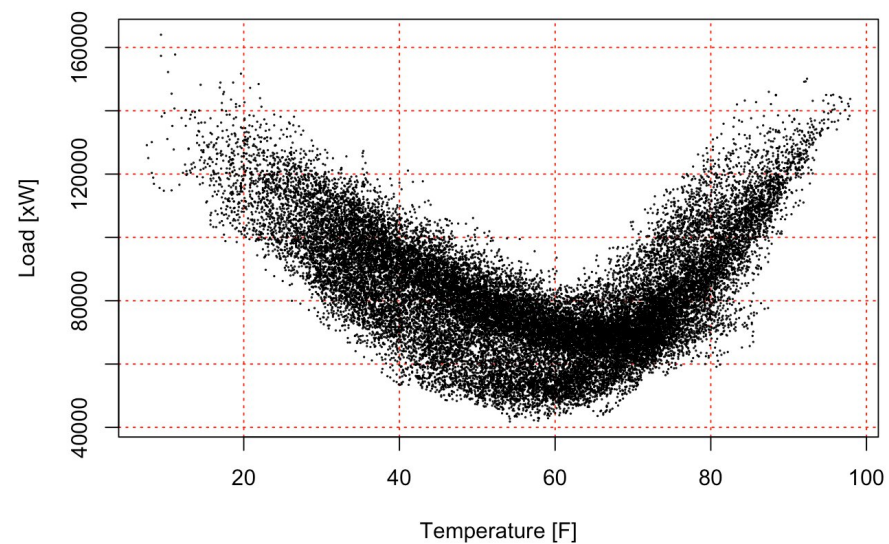
Plot

Summary

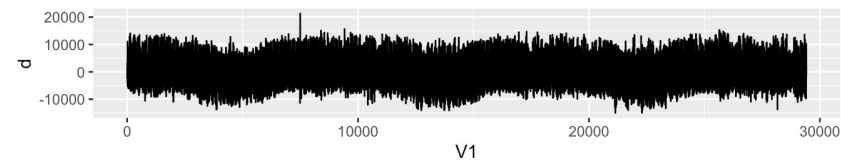
Table



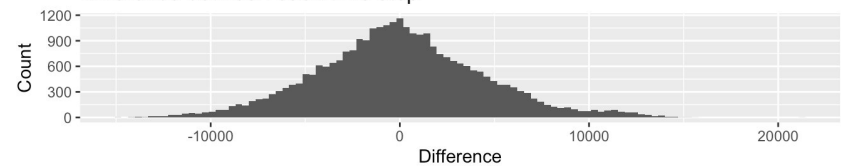
Relationship between energy load and temperature



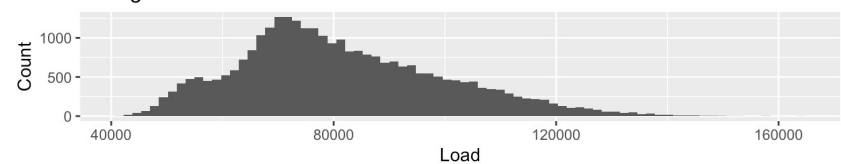
Difference in load between time steps



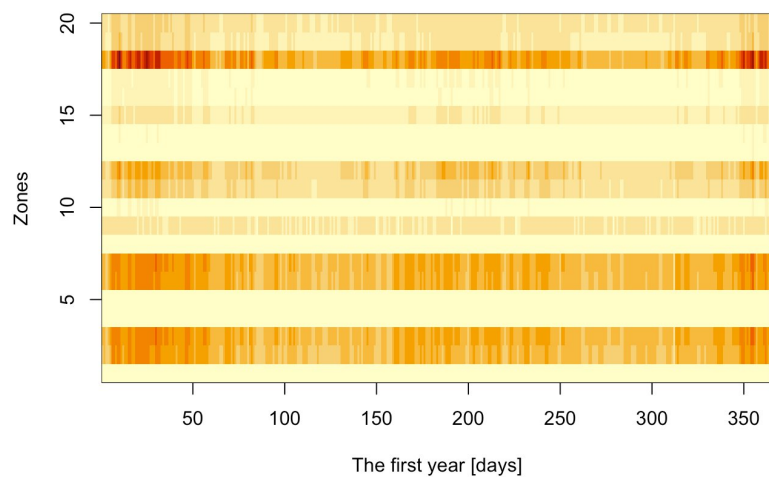
Difference between each time step



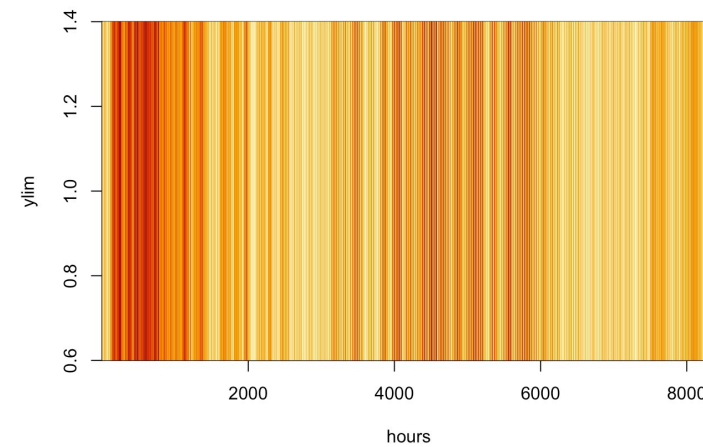
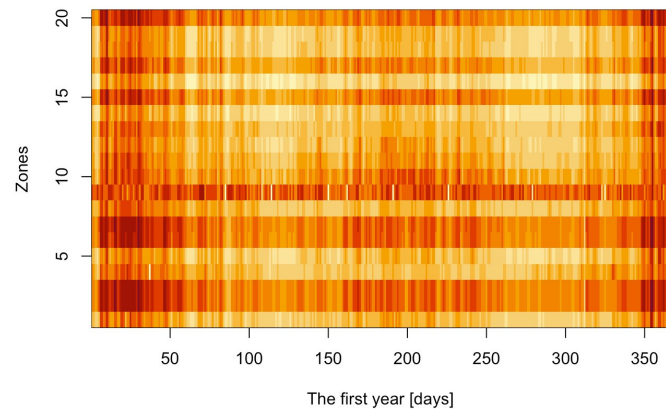
Histogram of load

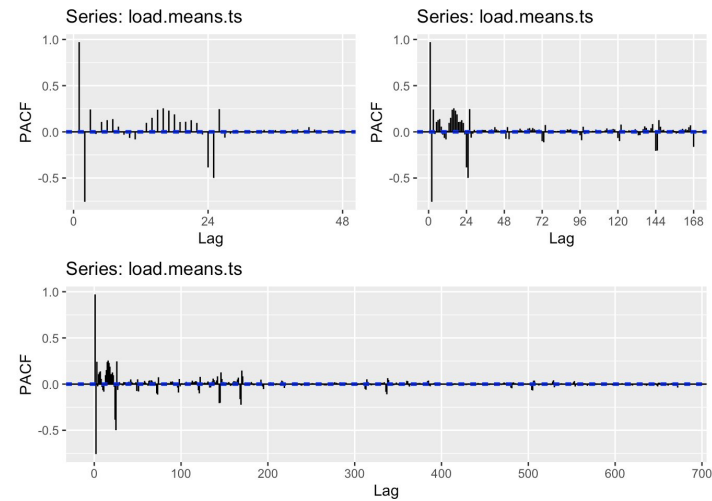
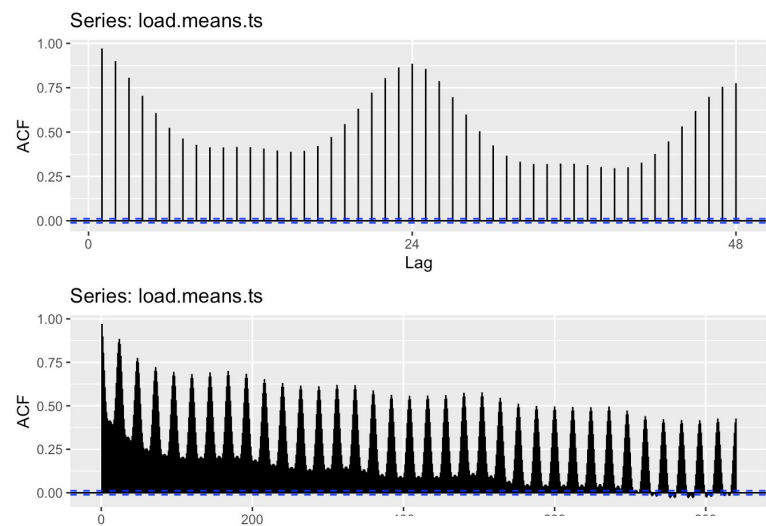


'Heatmap' of zones for the first year

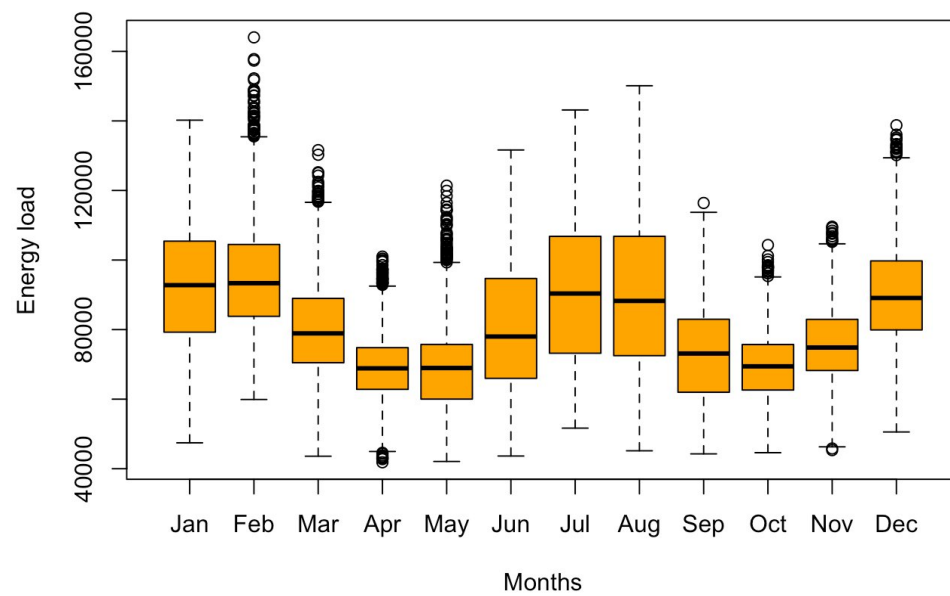


Standardized 'Heatmap' of zones for the first year





Average energy load over the zones vs months

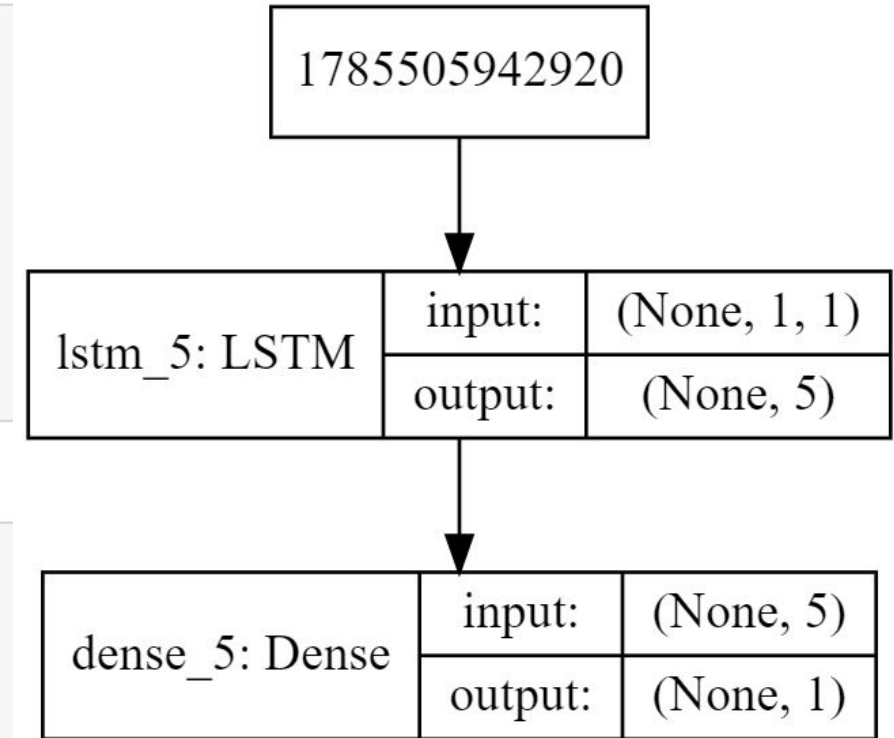


LSTM Network with Python

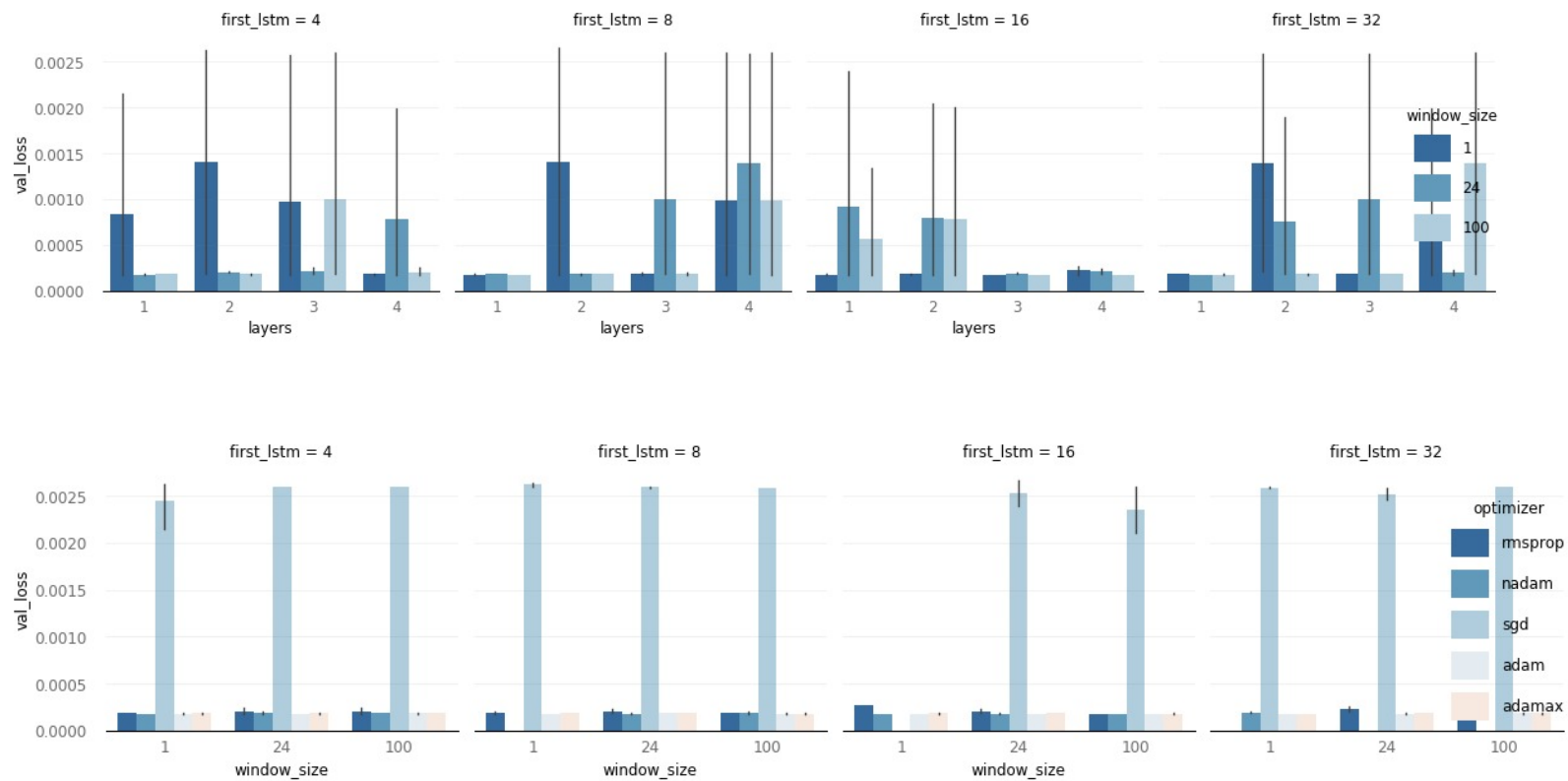
```
In [19]: 1 def get_predict_and_score(model, X, Y):
2         # transform the prediction to the original scale.
3         pred = normalizer.inverse_transform(model.predict(X))
4         # transform also the label to the original scale for interpretability.
5         orig_data = normalizer.inverse_transform([Y])
6         # calculate RMSE.
7         score = math.sqrt(mean_squared_error(orig_data[0], pred[:, 0]))
8         return(score, pred)
9
10 mse_train, train_predict = get_predict_and_score(vanilla_rnn, train_X, train_Y)
11 mse_test, test_predict = get_predict_and_score(vanilla_rnn, test_X, test_Y)
12
13 print("Training data error: %.2f MSE" % mse_train)
14 print("Test data error: %.2f MSE" % mse_test)
```

Training data error: 1437.70 MSE
Test data error: 1494.71 MSE

```
In [20]: 1 # Training predictions.
2 train_predictions = np.empty_like(dataset)
3 train_predictions[:, :] = np.nan
4 train_predictions[window_size:len(train_predict) + window_size, :] = train_predict
5
6 # Test predictions.
7 test_predictions = np.empty_like(dataset)
8 test_predictions[:, :] = np.nan
9 test_predictions[len(train_predict) + (window_size * 2) + 1:len(dataset) - 1, :] = test_predict
10
11 # Create the plot.
12 plt.figure(figsize = (15, 5))
13 plt.plot(normalizer.inverse_transform(dataset), label = "True Value")
14 plt.plot(train_predictions, label = "Training Predictions")
15 plt.plot(test_predictions, label = "Test Predictions")
16 plt.xlabel("Hours")
17 plt.ylabel("Load (MWh)")
18 plt.title("Zone 1 Comparison True vs. Predicted in Training and Testing")
19 plt.legend()
20 plt.show()
```



Results



Results

