NBEATS_RevIN_pdf

September 27, 2022

1 RevIN

RevIN is a flexible Reversible normalizing technique. Flexible meaning this method can be appled to any chosen layer (might perform differently in different instances). Revin the normalisation and denormalisation are performed in a way that it removes non stationary information (like mean, variance which changng throughout the time) and restores the same information in another layer. hence model forecast without the non stationary information.

(Normalisationa nd Denormalisation are done at the symmetric positions)

Revin helps to sort distribution shift problems due to non stationary nature of time series.

2 Hyper Parameters of NBEATS

As when I got this assignment NBEATS and Time Series forecasting was something new to me. For the sake of the assignment the hyperparameters used for taken directly from the N-BEATS papers to produce optimal result as I was learning and replicating N-BEATS paper results helped me understand the conceppt better.

2.1 Values from N-BEATS paper Figure 1 and Table 18/Appendix D

N_EPOCHS = 5000 # called "Iterations" in Table 18 N_NEURONS = 512 # called "Width" in Table 18 N_LAYERS = 4 N_STACKS = 30

INPUT_SIZE = WINDOW_SIZE * HORIZON # called "Lookback" in Table 18 THETA_SIZE = INPUT_SIZE + HORIZON

INPUT_SIZE, THETA_SIZE

3 What works and what doesn't.

The algorithm works fine for almost all the dataset I used from dataset50. Due my machine's GPU and RAM problems I could not produce the result for all the datasets. However the algorithm should work fine for almost all the datasets with little to no tweaking.

As proposed in the paper the result would have been better if we loop through multiple window sizes rangng from 2*(Horizon) to 7*(Horizon) or if a better sequence made more sense. I chose a single Window size due to GPU restriction. However I can build the code to loop through different window sizes if required.

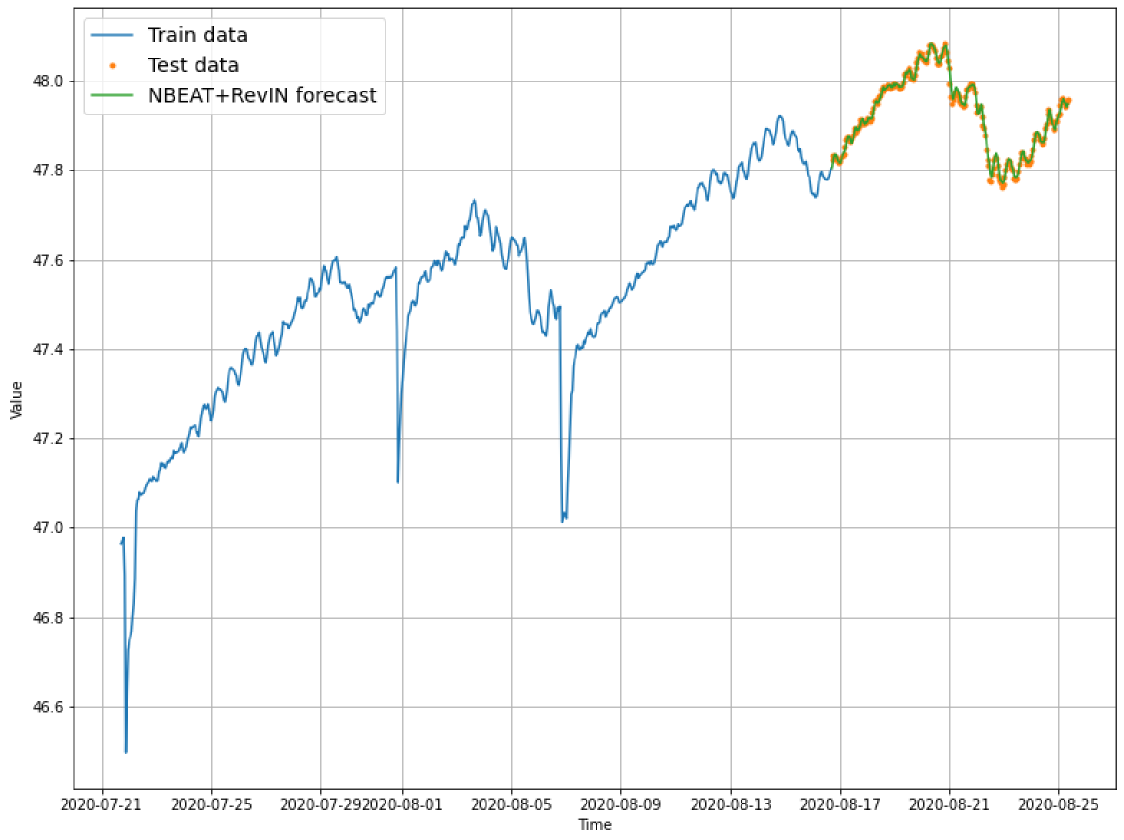
Do Go through the notebook and steps are simple execute each cell in order one by one and to replicate results for a different dataset follow the same steps but add your datasets wherever required.

```
[1]: #CSV - 0
```

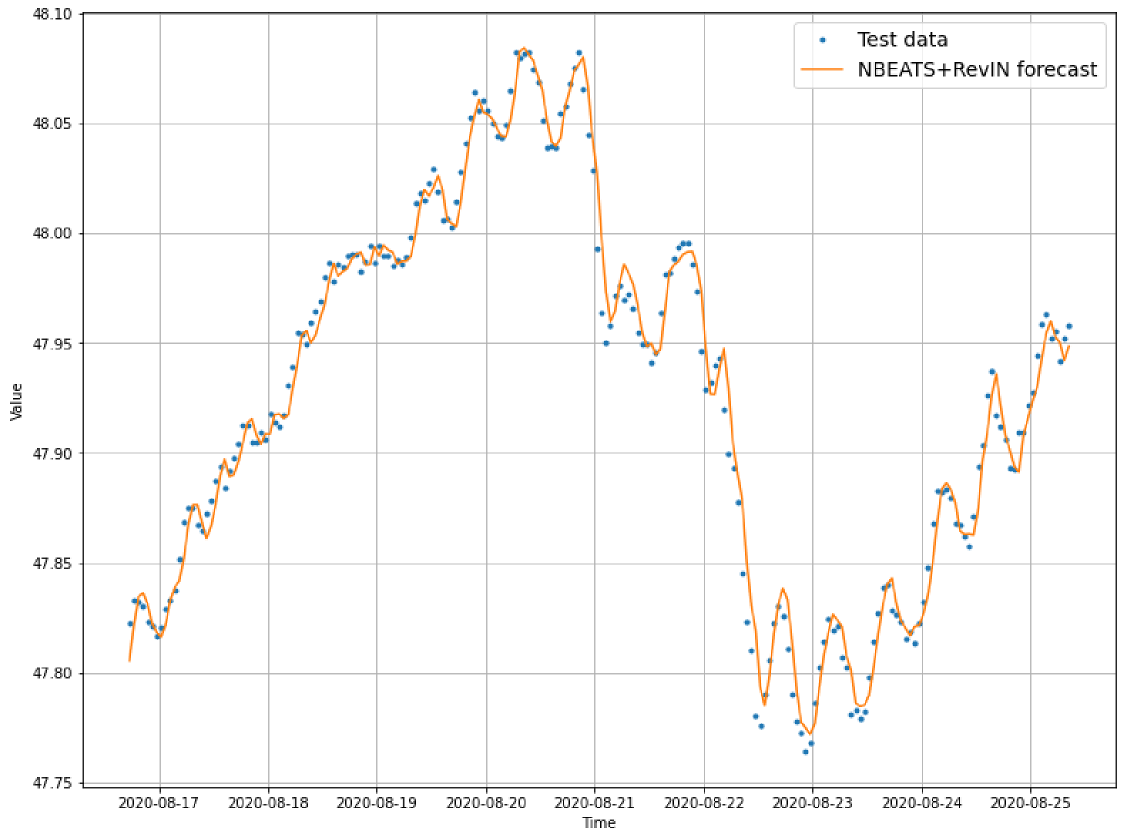
```
[33]: import numpy as np
      # Evaluate ensemble model(s) predictions
      ensemble_results = evaluate_preds(y_true=y_test,
                                        y_pred=np.median(ensemble_preds, axis=0)) #␣
       ↪take the median across all ensemble predictions
      ensemble_results
```

```
[33]: {'mae': 0.00859631,
       'mse': 0.00012589259,
       'rmse': 0.011220187,
       'mape': 0.017939832,
       'mase': 0.97986066}
```

```
[34]: import matplotlib.pyplot as plt
      plt.figure(figsize=(13, 10))
      ensemble_median = np.median(ensemble_preds, axis=0)
      offset = 0 # offset the values by 300 timesteps
      plot_time_series(timesteps=X_train.index[offset:], values=y_train[offset:],␣
       ↪label="Train data")
      plot_time_series(timesteps=X_test.index[offset:], values=y_test[offset:],␣
       ↪format = '.', start=offset, label="Test data")
      plot_time_series(timesteps=X_test.index[offset:], values=ensemble_median[offset:
       ↪], start=offset, label="NBEAT+RevIN forecast");
```



[35]:
```
plt.figure(figsize=(13, 10))
offset = 0 # offset the values by 300 timesteps
plot_time_series(timesteps=X_test.index[offset:], values=y_test[offset:],␣
 ↪format = ".", start=offset, label="Test data")
plot_time_series(timesteps=X_test.index[offset:], values=ensemble_median[offset:
 ↪], format="-", start=offset, label="NBEATS+RevIN forecast");
```



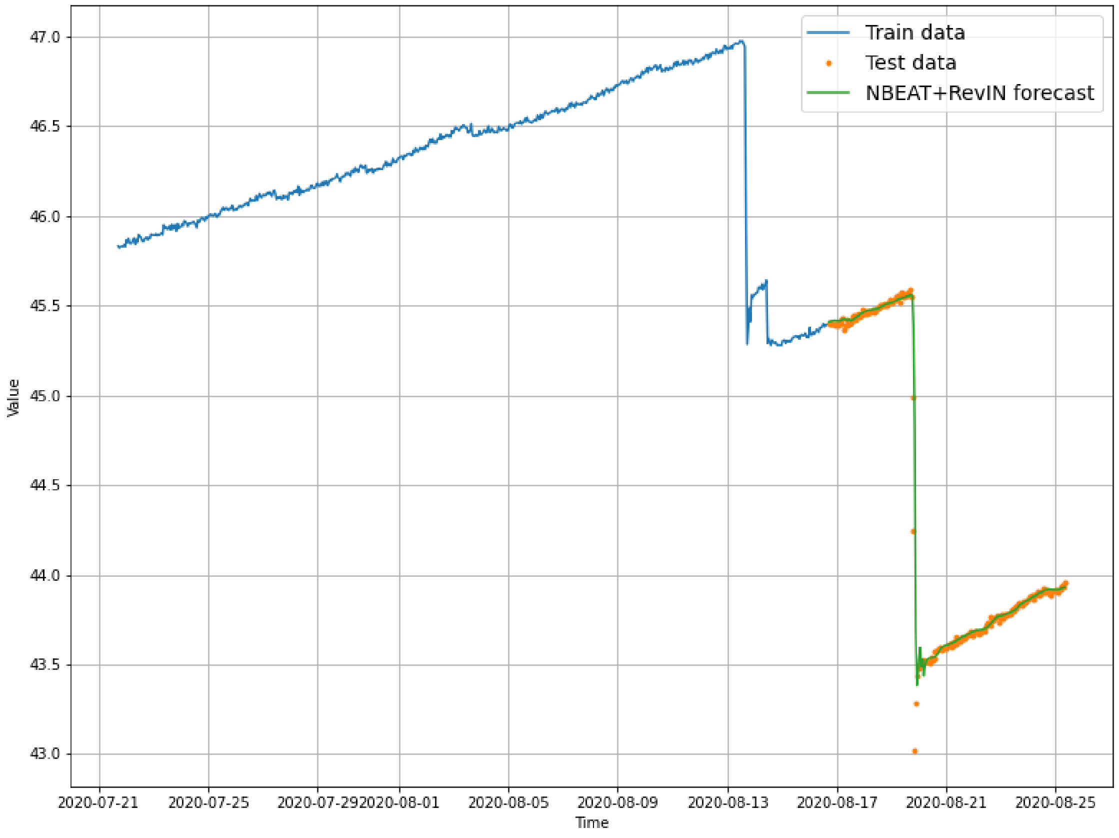
```
[2]: #CSV - 1
```

```
[ ]: # Evaluate ensemble model(s) predictions
     ensemble_results = evaluate_preds(y_true=y_test,
                                       y_pred=np.median(ensemble_preds, axis=0)) #␣
      ↪take the median across all ensemble predictions
     ensemble_results
```

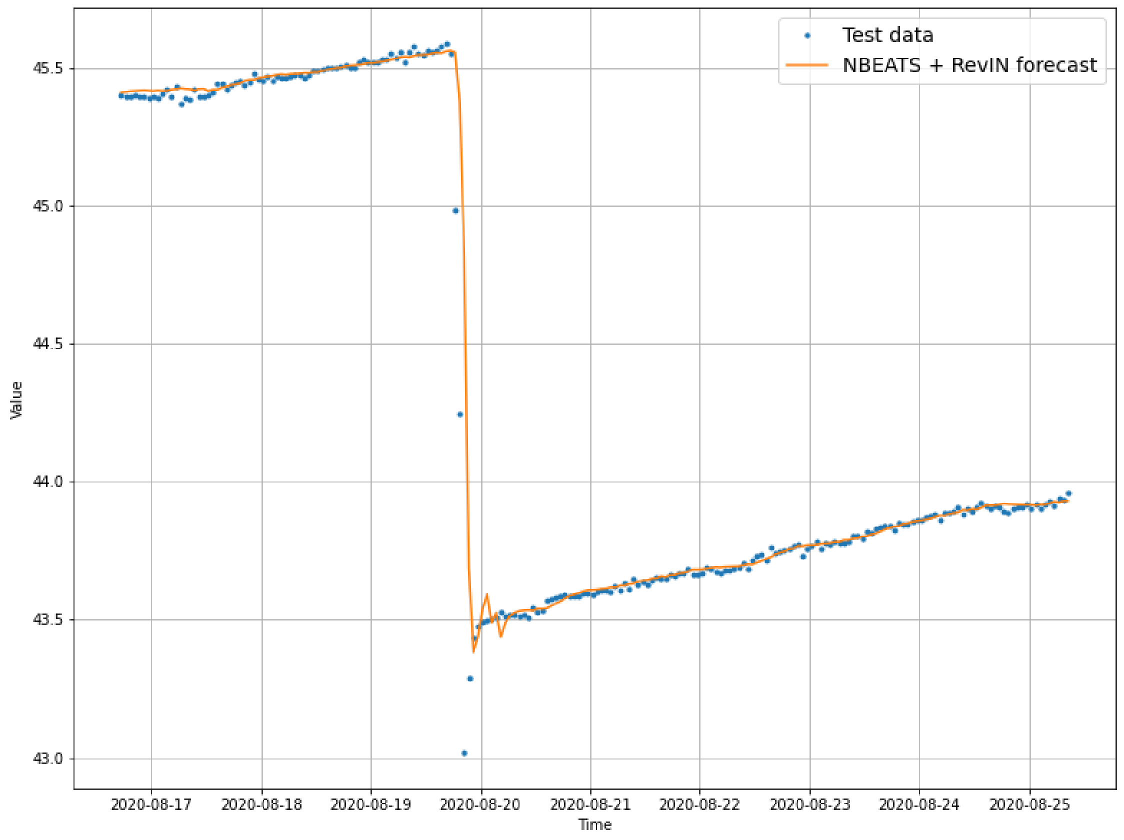
```
[ ]: {'mae': 0.03162775,
      'mse': 0.0243611,
      'rmse': 0.15608042,
      'mape': 0.07199581,
      'mase': 1.2107365}
```

```
[ ]: import matplotlib.pyplot as plt
     plt.figure(figsize=(13, 10))
     ensemble_median = np.median(ensemble_preds, axis=0)
     offset = 0 # offset the values by 300 timesteps
     plot_time_series(timesteps=X_train.index[offset:], values=y_train[offset:],␣
      ↪label="Train data")
```

```
plot_time_series(timesteps=X_test.index[offset:], values=y_test[offset:],␣
 ↪format = '.', start=offset, label="Test data")
plot_time_series(timesteps=X_test.index[offset:], values=ensemble_median[offset:
 ↪], start=offset, label="NBEAT+RevIN forecast");
```



[ ]:
```
plt.figure(figsize=(13, 10))
offset = 0 # offset the values by 300 timesteps
plot_time_series(timesteps=X_test.index[offset:], values=y_test[offset:],␣
 ↪format = ".", start=offset, label="Test data")
plot_time_series(timesteps=X_test.index[offset:], values=ensemble_median[offset:
 ↪], format="-", start=offset, label="NBEATS + RevIN forecast");
```



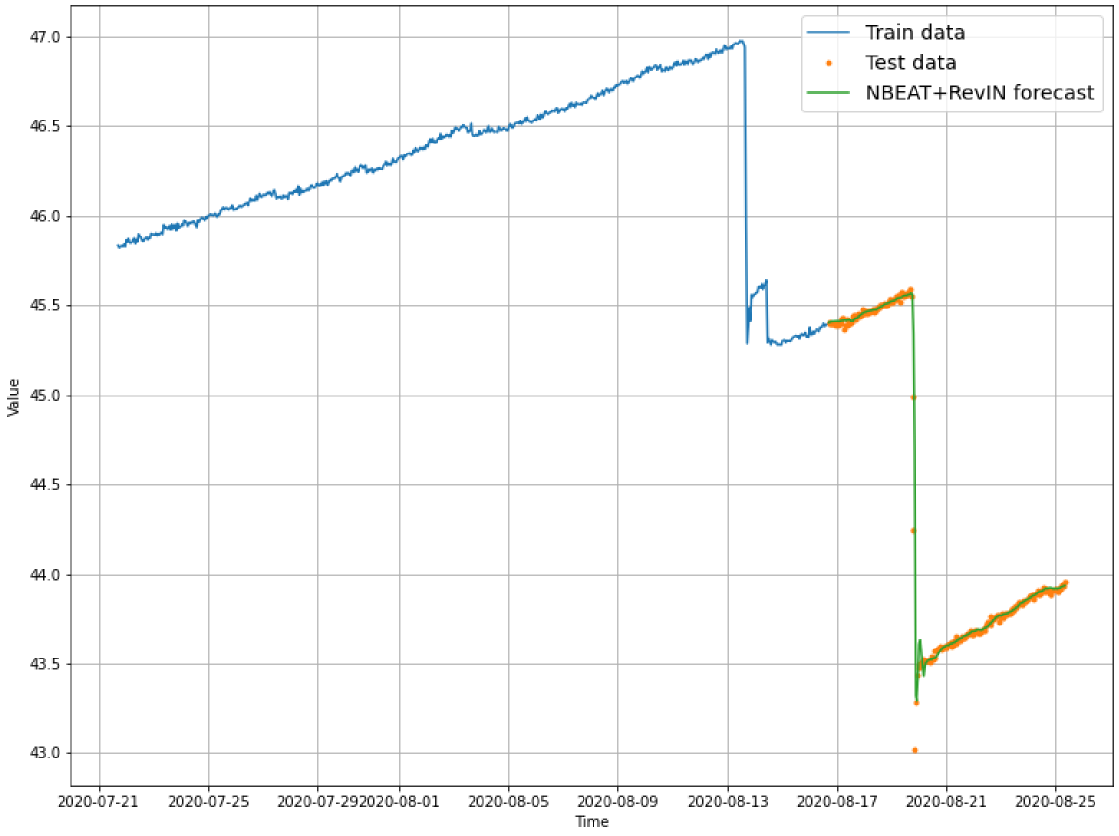
```
[3]: #CSV - 2
```

```
[ ]: import numpy as np
     # Evaluate ensemble model(s) predictions
     ensemble_results = evaluate_preds(y_true=y_test,
                                       y_pred=np.median(ensemble_preds, axis=0)) #␣
      ↪take the median across all ensemble predictions
     ensemble_results
```

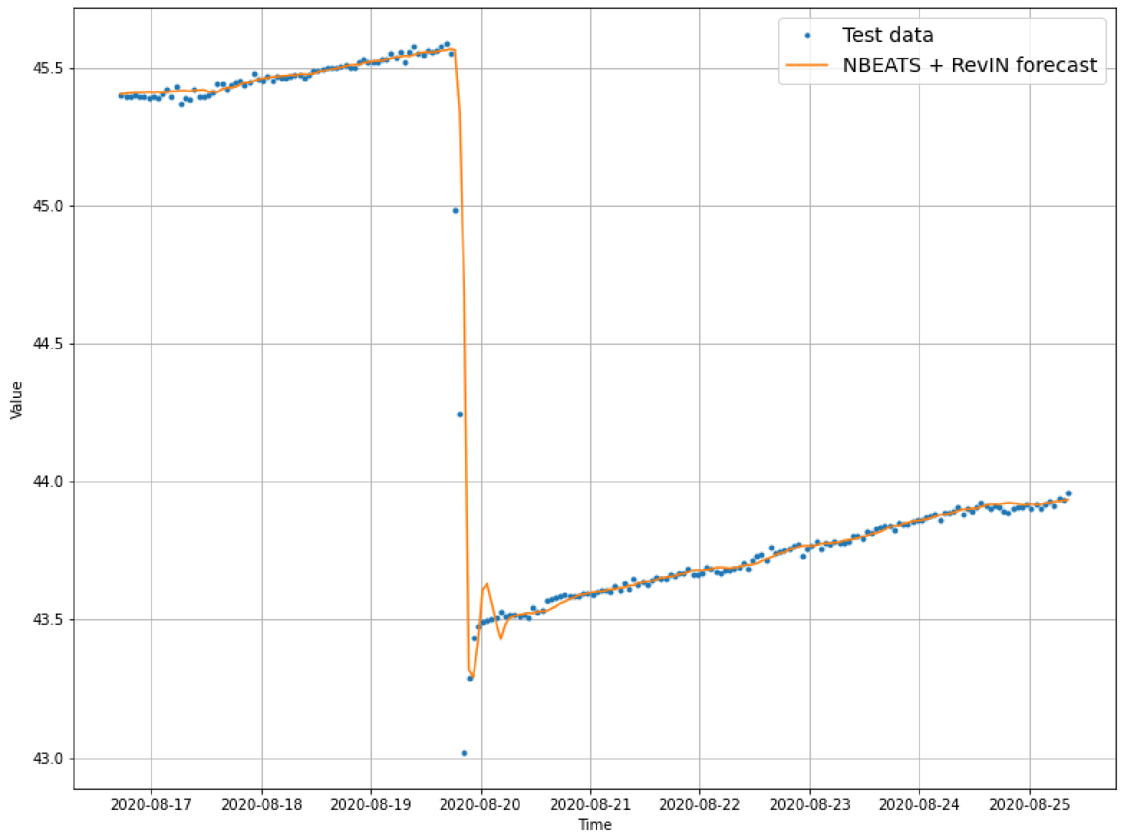
```
[ ]: {'mae': 0.029394241,
      'mse': 0.021217206,
      'rmse': 0.14566128,
      'mape': 0.06689647,
      'mase': 1.125236}
```

```
[21]: import matplotlib.pyplot as plt
      plt.figure(figsize=(13, 10))
      ensemble_median = np.median(ensemble_preds, axis=0)
      offset = 0 # offset the values by 300 timesteps
```

```
plot_time_series(timesteps=X_train.index[offset:], values=y_train[offset:],␣
↪label="Train data")
plot_time_series(timesteps=X_test.index[offset:], values=y_test[offset:],␣
↪format = '.', start=offset, label="Test data")
plot_time_series(timesteps=X_test.index[offset:], values=ensemble_median[offset:
↪], start=offset, label="NBEAT+RevIN forecast");
```



[22]:
```
plt.figure(figsize=(13, 10))
offset = 0 # offset the values by 300 timesteps
plot_time_series(timesteps=X_test.index[offset:], values=y_test[offset:],␣
↪format = ".", start=offset, label="Test data")
plot_time_series(timesteps=X_test.index[offset:], values=ensemble_median[offset:
↪], format="-", start=offset, label="NBEATS + RevIN forecast");
```



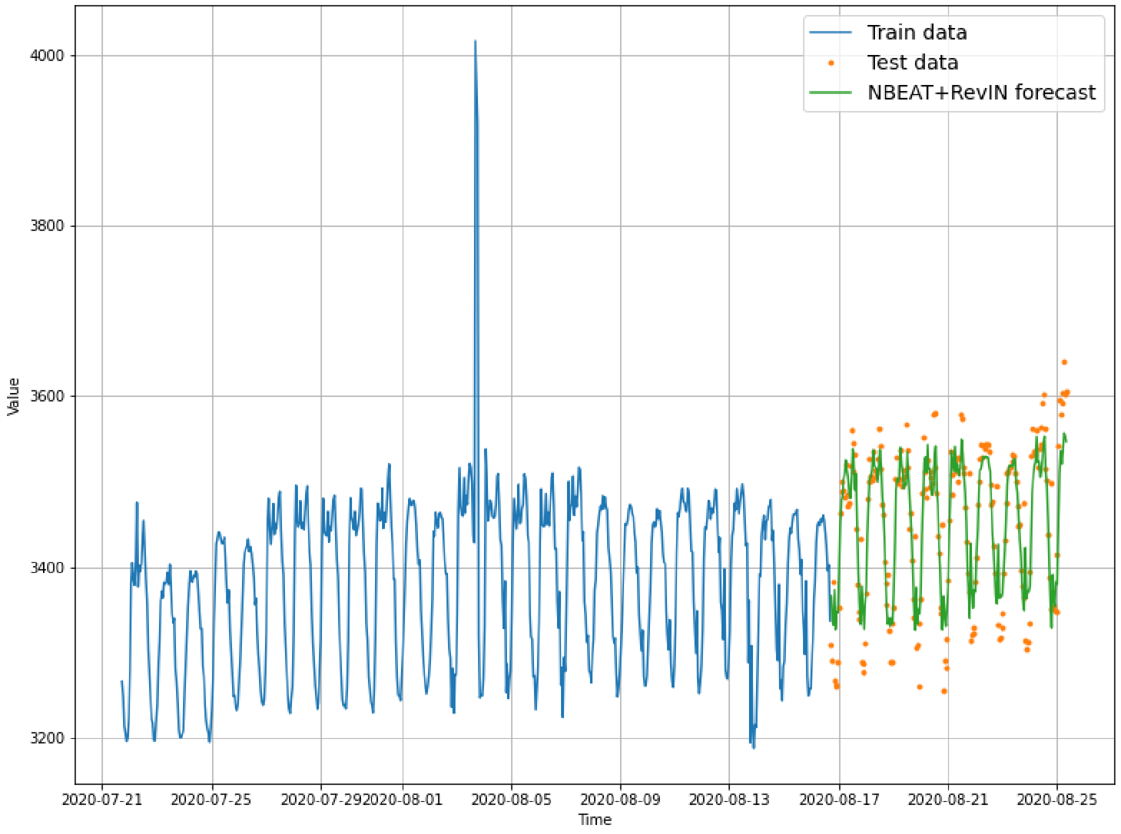
```
[4]: #CSV - 3
```

```
[19]: import numpy as np
      # Evaluate ensemble model(s) predictions
      ensemble_results = evaluate_preds(y_true=y_test,
                                        y_pred=np.median(ensemble_preds, axis=0)) #␣
       ↪take the median across all ensemble predictions
      ensemble_results
```

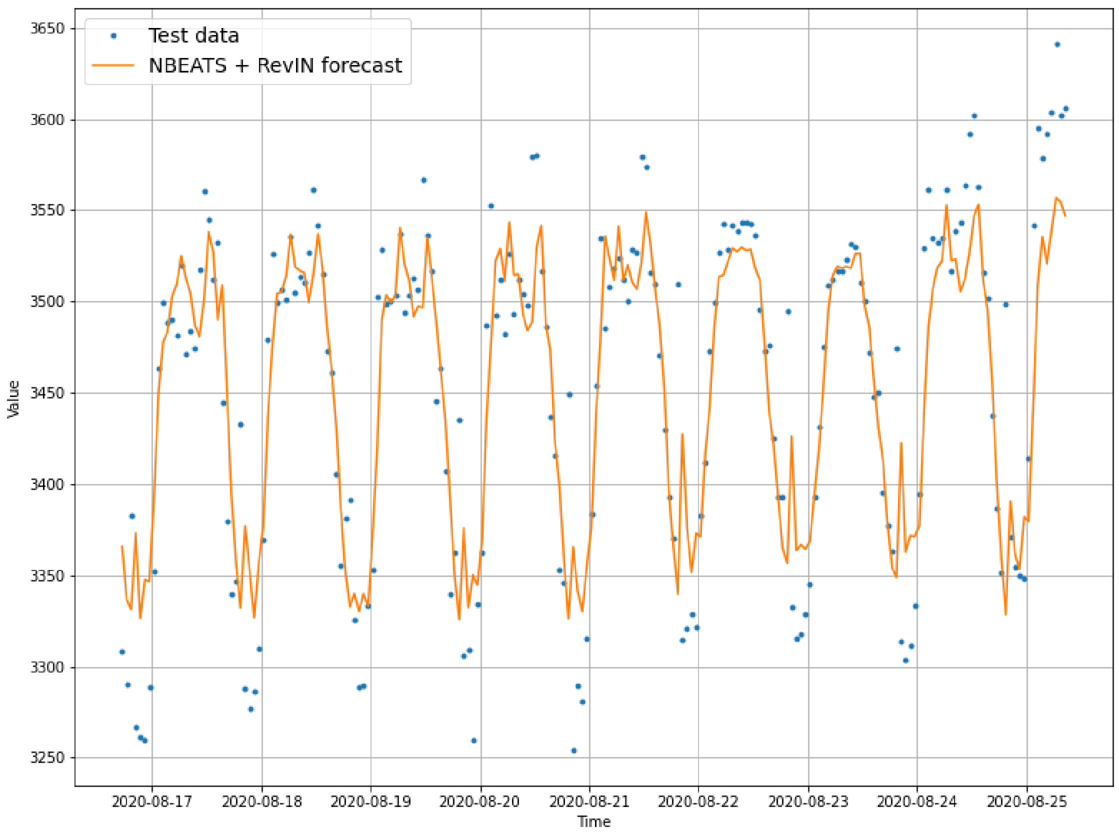
```
[19]: {'mae': 33.226906,
       'mse': 2176.4475,
       'rmse': 46.652412,
       'mape': 0.9671772,
       'mase': 0.8759156}
```

```
[21]: import matplotlib.pyplot as plt
      plt.figure(figsize=(13, 10))
      ensemble_median = np.median(ensemble_preds, axis=0)
      offset = 0 # offset the values by 300 timesteps
```

```
plot_time_series(timesteps=X_train.index[offset:], values=y_train[offset:],␣
↪label="Train data")
plot_time_series(timesteps=X_test.index[offset:], values=y_test[offset:],␣
↪format = '.', start=offset, label="Test data")
plot_time_series(timesteps=X_test.index[offset:], values=ensemble_median[offset:
↪], start=offset, label="NBEAT+RevIN forecast");
```



[22]:
```
plt.figure(figsize=(13, 10))
offset = 0 # offset the values by 300 timesteps
plot_time_series(timesteps=X_test.index[offset:], values=y_test[offset:],␣
↪format = ".", start=offset, label="Test data")
plot_time_series(timesteps=X_test.index[offset:], values=ensemble_median[offset:
↪], format="-", start=offset, label="NBEATS + RevIN forecast");
```



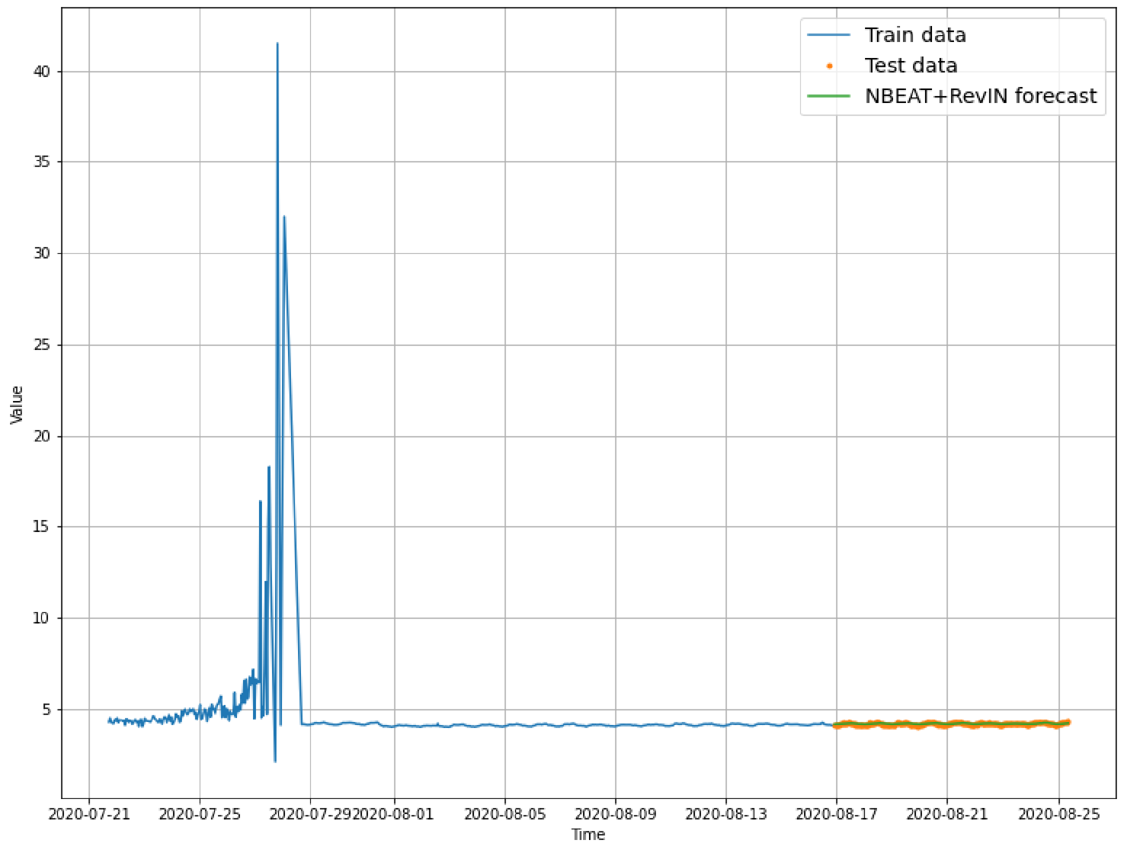
```
[5]: #CSV - 5
```

```
[34]: import numpy as np
      # Evaluate ensemble model(s) predictions
      ensemble_results = evaluate_preds(y_true=y_test,
                                        y_pred=np.median(ensemble_preds, axis=0)) #␣
       ↪take the median across all ensemble predictions
      ensemble_results
```

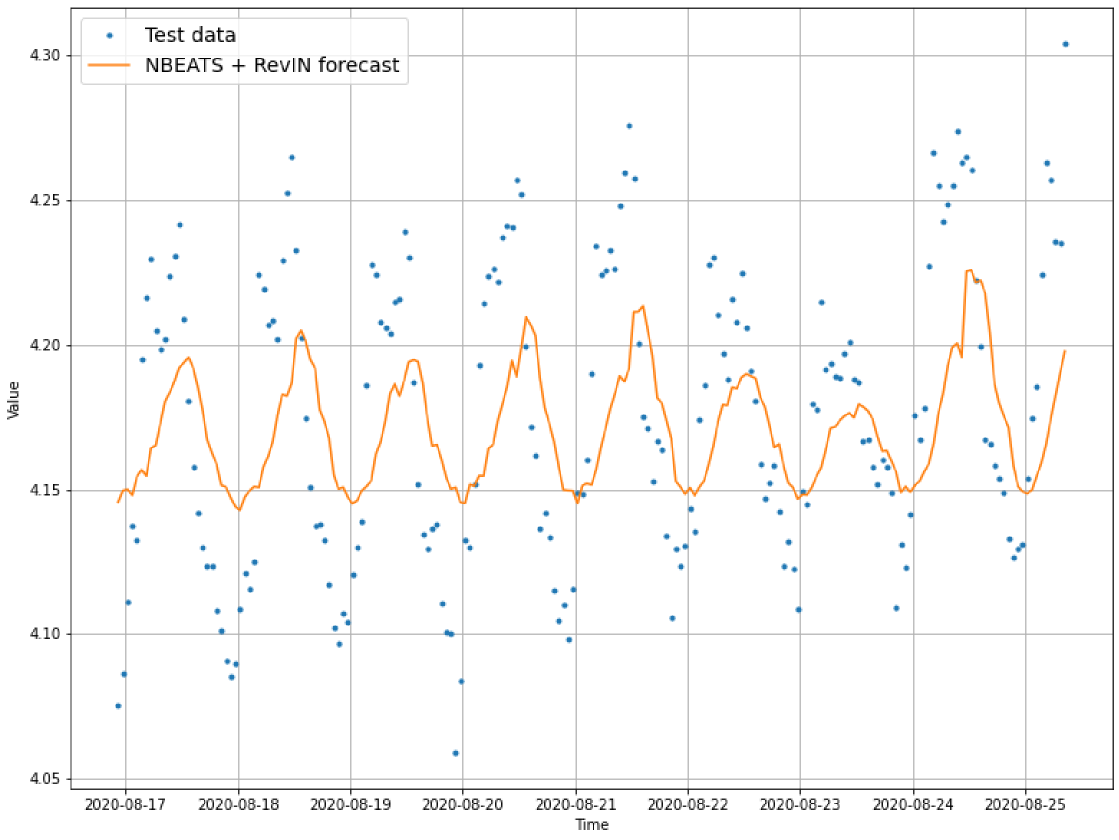
```
[34]: {'mae': 0.037641846,
       'mse': 0.001879538,
       'rmse': 0.043353636,
       'mape': 0.8998065,
       'mase': 2.2969658}
```

```
[37]: import matplotlib.pyplot as plt
      plt.figure(figsize=(13, 10))
      ensemble_median = np.median(ensemble_preds, axis=0)
      offset = 0 # offset the values by 300 timesteps
      plot_time_series(timesteps=X_train.index[offset:], values=y_train[offset:],␣
       ↪label="Train data")
```

```
plot_time_series(timesteps=X_test.index[offset:], values=y_test[offset:],␣
 ↪format = '.', start=offset, label="Test data")
plot_time_series(timesteps=X_test.index[offset:], values=ensemble_median[offset:
 ↪], start=offset, label="NBEAT+RevIN forecast");
```



[46]:
```
plt.figure(figsize=(13, 10))
offset = 0
plot_time_series(timesteps=X_test.index[offset:], values=y_test[offset:],␣
 ↪format = ".", start=offset, label="Test data")
plot_time_series(timesteps=X_test.index[offset:], values=ensemble_median[offset:
 ↪], format="-", start=offset, label="NBEATS + RevIN forecast");
```



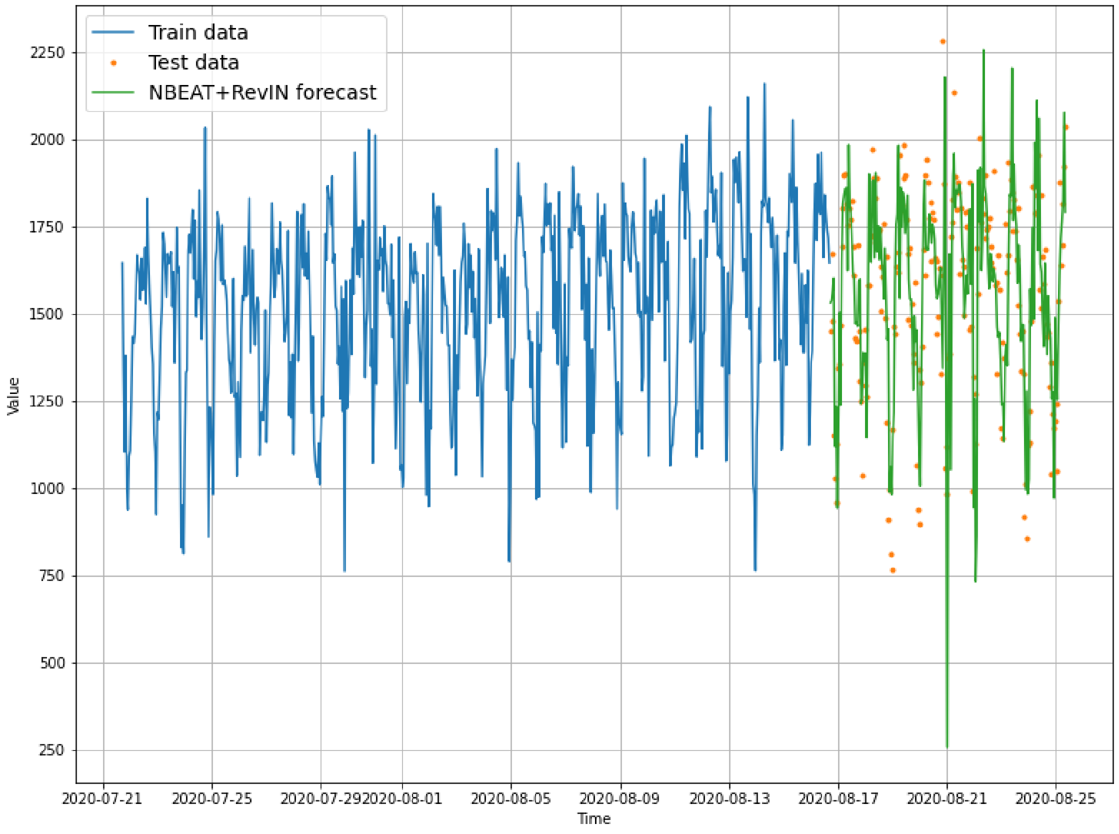
[6]:
```
#CSV - 7
```

[27]:
```
import numpy as np
# Evaluate ensemble model(s) predictions
ensemble_results = evaluate_preds(y_true=y_test,
                                  y_pred=np.median(ensemble_preds, axis=0)) #␣
 ↪take the median across all ensemble predictions
ensemble_results
```

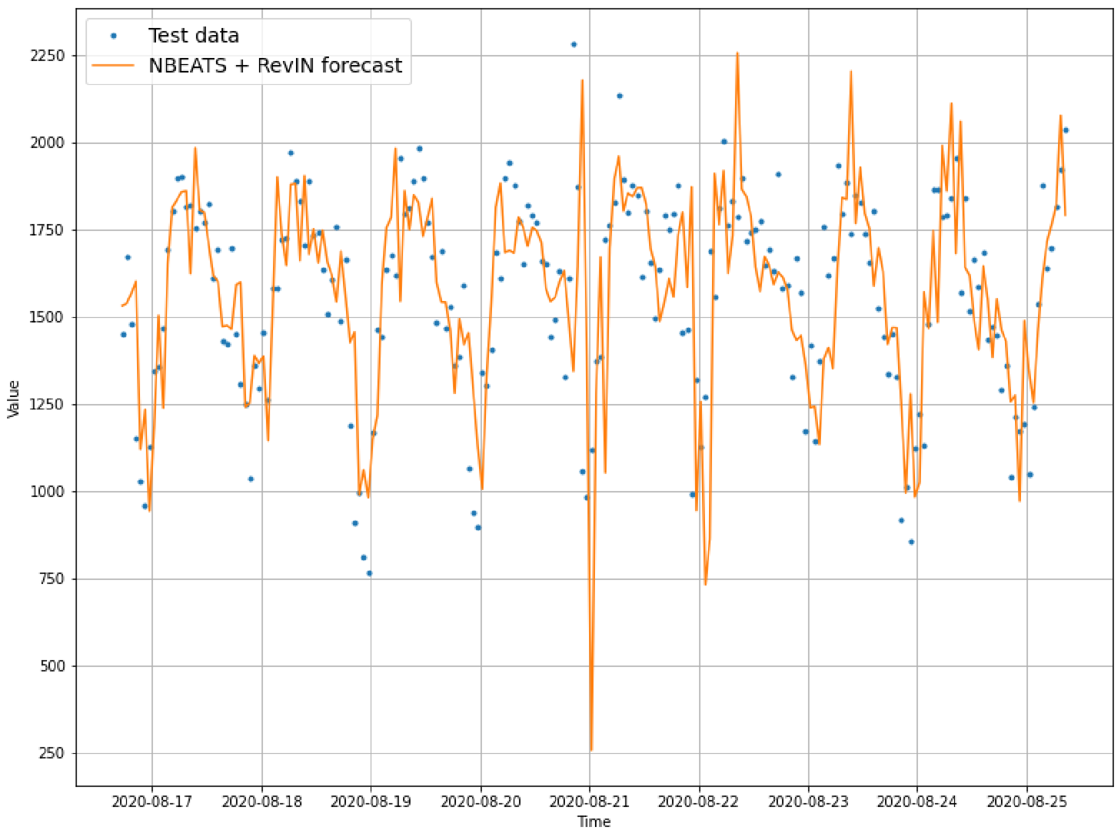
[27]:
```
{'mae': 174.61816,
 'mse': 59687.13,
 'rmse': 244.3095,
 'mape': 12.349029,
 'mase': 1.0740503}
```

[28]:
```
import matplotlib.pyplot as plt
plt.figure(figsize=(13, 10))
ensemble_median = np.median(ensemble_preds, axis=0)
offset = 0 # offset the values by 300 timesteps
```

```
plot_time_series(timesteps=X_train.index[offset:], values=y_train[offset:],␣
↪label="Train data")
plot_time_series(timesteps=X_test.index[offset:], values=y_test[offset:],␣
↪format = '.', start=offset, label="Test data")
plot_time_series(timesteps=X_test.index[offset:], values=ensemble_median[offset:
↪], start=offset, label="NBEAT+RevIN forecast");
```



[29]:
```
plt.figure(figsize=(13, 10))
offset = 0 # offset the values by 300 timesteps
plot_time_series(timesteps=X_test.index[offset:], values=y_test[offset:],␣
↪format = ".", start=offset, label="Test data")
plot_time_series(timesteps=X_test.index[offset:], values=ensemble_median[offset:
↪], format="-", start=offset, label="NBEATS + RevIN forecast");
```



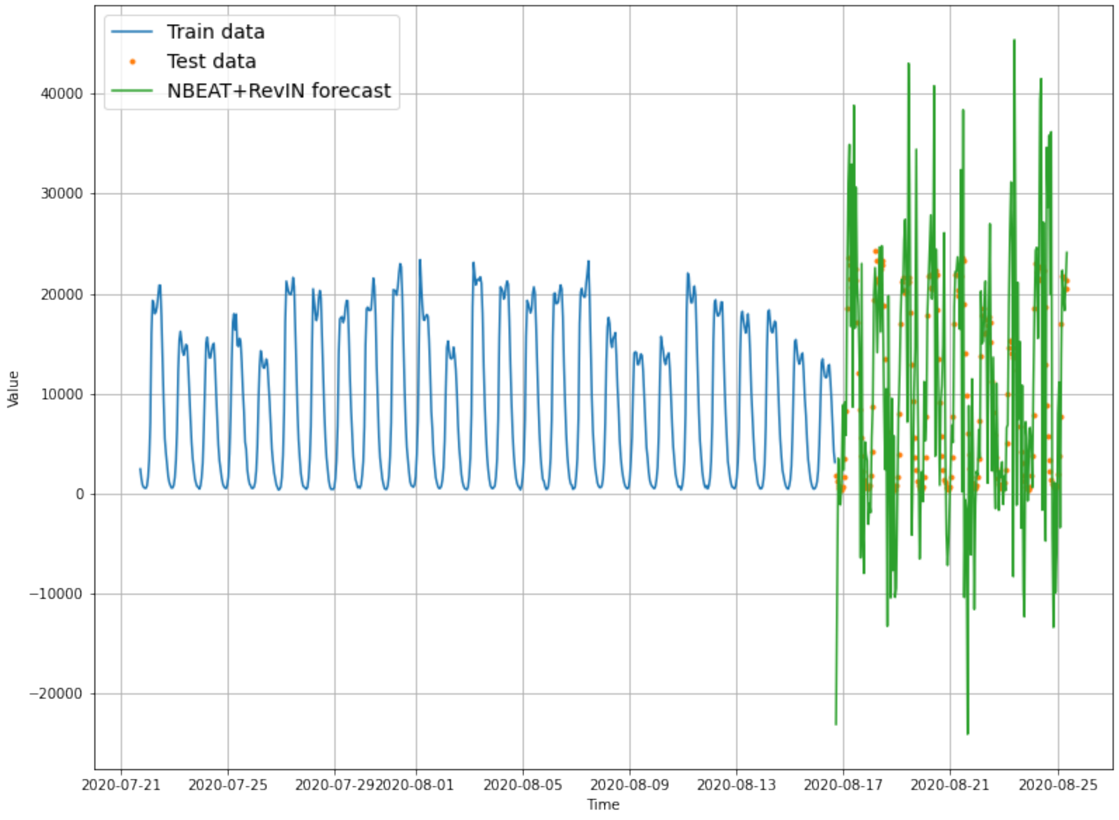
[7]:
```
#CSV - 8
```

[17]:
```
import numpy as np
# Evaluate ensemble model(s) predictions
ensemble_results = evaluate_preds(y_true=y_test,
                                  y_pred=np.median(ensemble_preds, axis=0)) #␣
 ↪take the median across all ensemble predictions
ensemble_results
```

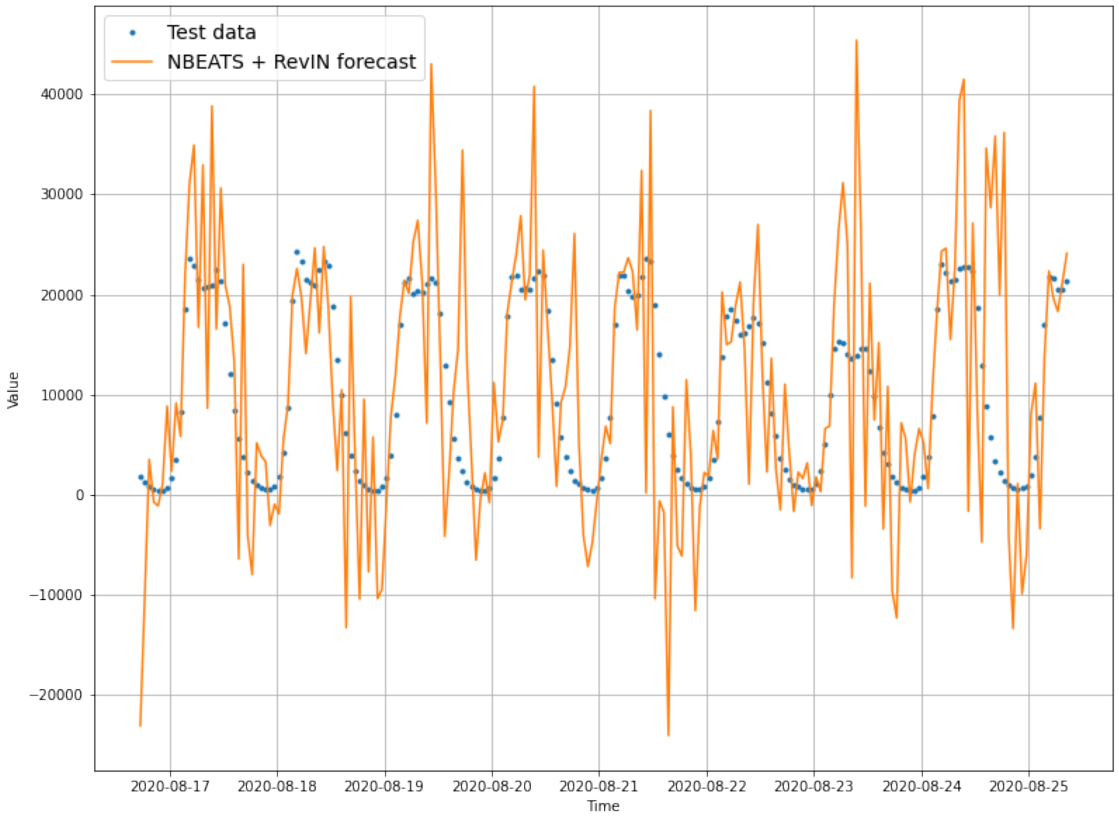
[17]:
```
{'mae': 7349.5547,
 'mse': 105993096.0,
 'rmse': 10295.295,
 'mape': 283.79337,
 'mase': 3.9054837}
```

[18]:
```
import matplotlib.pyplot as plt
plt.figure(figsize=(13, 10))
ensemble_median = np.median(ensemble_preds, axis=0)
offset = 0 # offset the values by 300 timesteps
plot_time_series(timesteps=X_train.index[offset:], values=y_train[offset:],␣
 ↪label="Train data")
```

```
plot_time_series(timesteps=X_test.index[offset:], values=y_test[offset:],␣
↪format = '.', start=offset, label="Test data")
plot_time_series(timesteps=X_test.index[offset:], values=ensemble_median[offset:
↪], start=offset, label="NBEAT+RevIN forecast");
```



[19]:
```
plt.figure(figsize=(13, 10))
offset = 0 # offset the values by 300 timesteps
plot_time_series(timesteps=X_test.index[offset:], values=y_test[offset:],␣
↪format = ".", start=offset, label="Test data")
plot_time_series(timesteps=X_test.index[offset:], values=ensemble_median[offset:
↪], format="-", start=offset, label="NBEATS + RevIN forecast");
```



[8]:
```
#CSV - 9
```

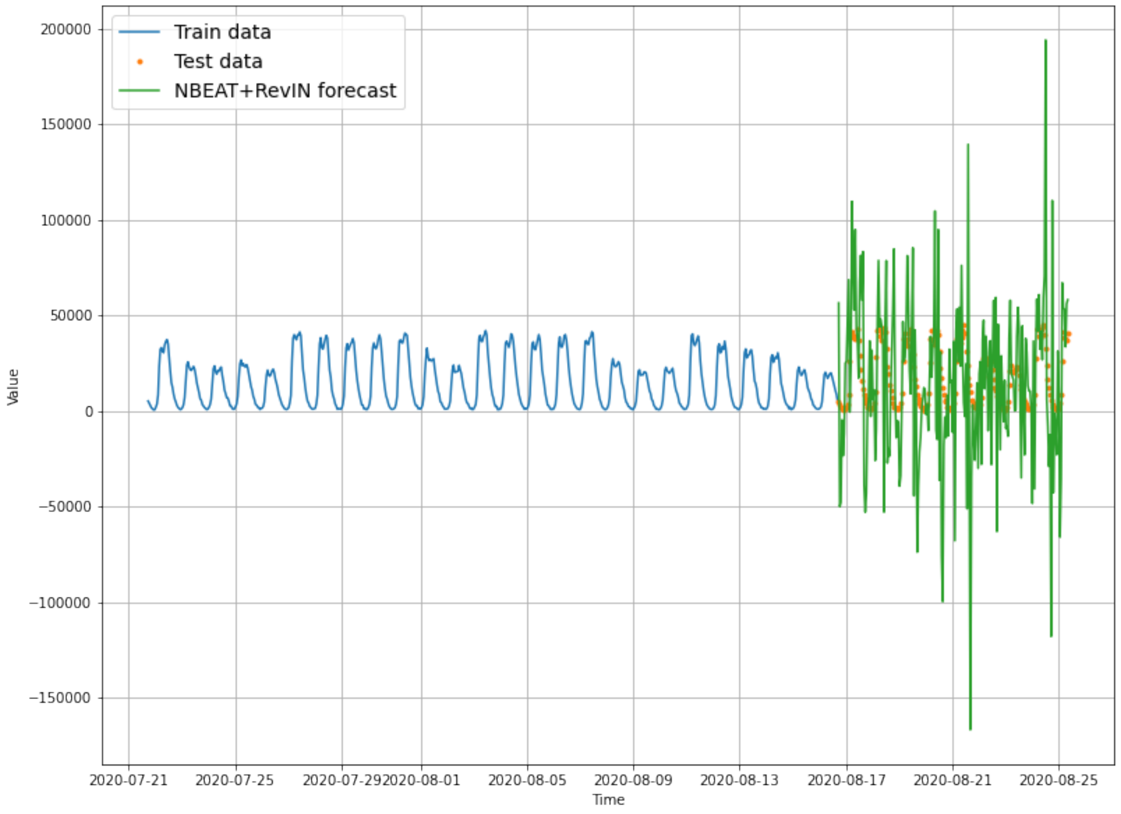
[17]:
```
import numpy as np
# Evaluate ensemble model(s) predictions
ensemble_results = evaluate_preds(y_true=y_test,
                                  y_pred=np.median(ensemble_preds, axis=0)) #␣
↪take the median across all ensemble predictions
ensemble_results
```

[17]:
```
{'mae': 28701.592,
 'mse': 1581590400.0,
 'rmse': 39769.215,
 'mape': 626.6846,
 'mase': 7.924992}
```

[18]:
```
import matplotlib.pyplot as plt
plt.figure(figsize=(13, 10))
ensemble_median = np.median(ensemble_preds, axis=0)
offset = 0 # offset the values by 300 timesteps
plot_time_series(timesteps=X_train.index[offset:], values=y_train[offset:],␣
↪label="Train data")
```

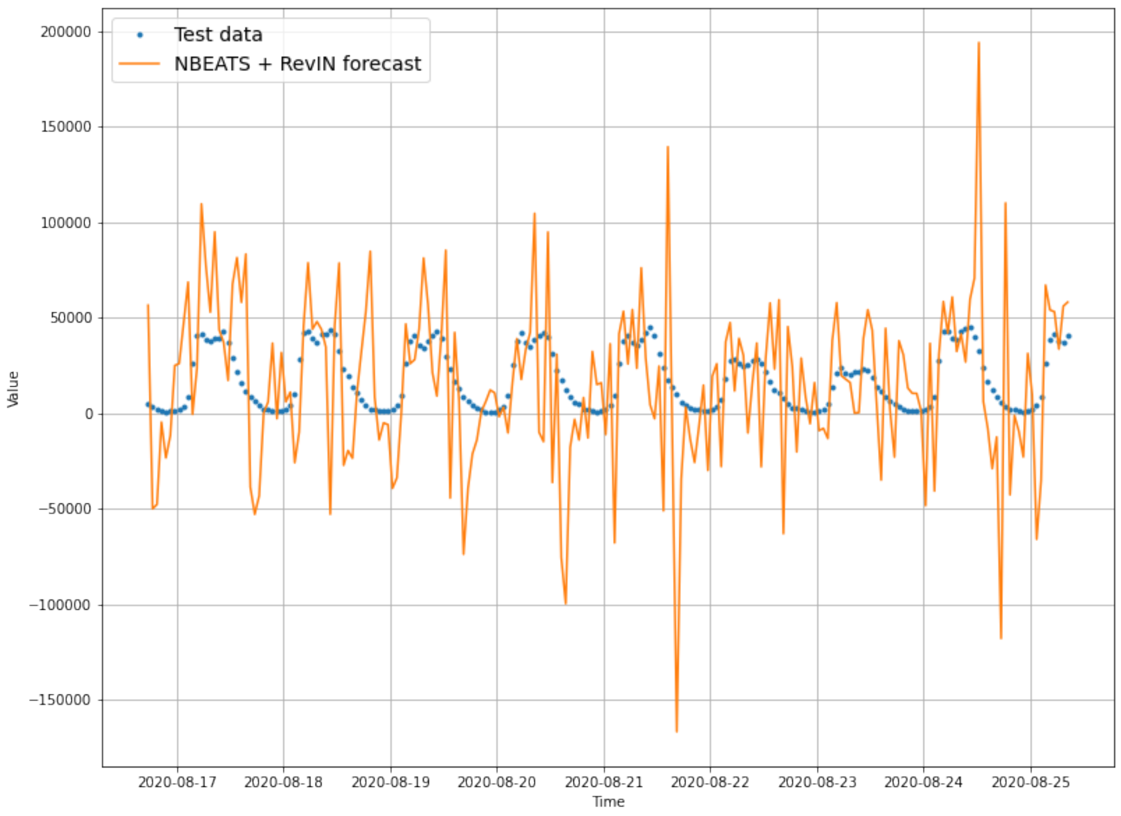
```
plot_time_series(timesteps=X_test.index[offset:], values=y_test[offset:],␣
↪format = '.', start=offset, label="Test data")
plot_time_series(timesteps=X_test.index[offset:], values=ensemble_median[offset:
↪], start=offset, label="NBEAT+RevIN forecast");
```

[18]:
```
<Figure size 936x720 with 0 Axes>
```



```
<Figure size 936x720 with 0 Axes>
```

[19]:
```
plt.figure(figsize=(13, 10))
offset = 0 # offset the values by 300 timesteps
plot_time_series(timesteps=X_test.index[offset:], values=y_test[offset:],␣
↪format = ".", start=offset, label="Test data")
plot_time_series(timesteps=X_test.index[offset:], values=ensemble_median[offset:
↪], format="-", start=offset, label="NBEATS + RevIN forecast");
```



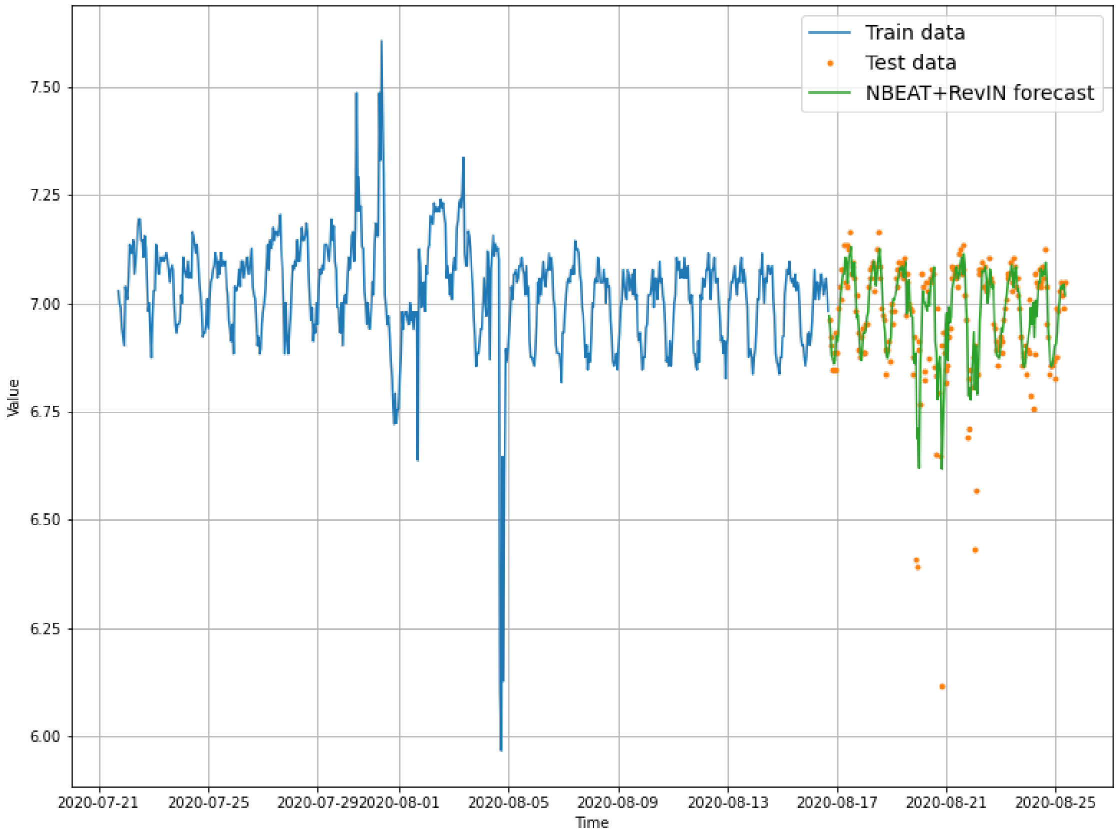
```
[9]: #CSV - 10
```

```
[17]: import numpy as np
      # Evaluate ensemble model(s) predictions
      ensemble_results = evaluate_preds(y_true=y_test,
                                        y_pred=np.median(ensemble_preds, axis=0)) #␣
       ↪take the median across all ensemble predictions
      ensemble_results
```

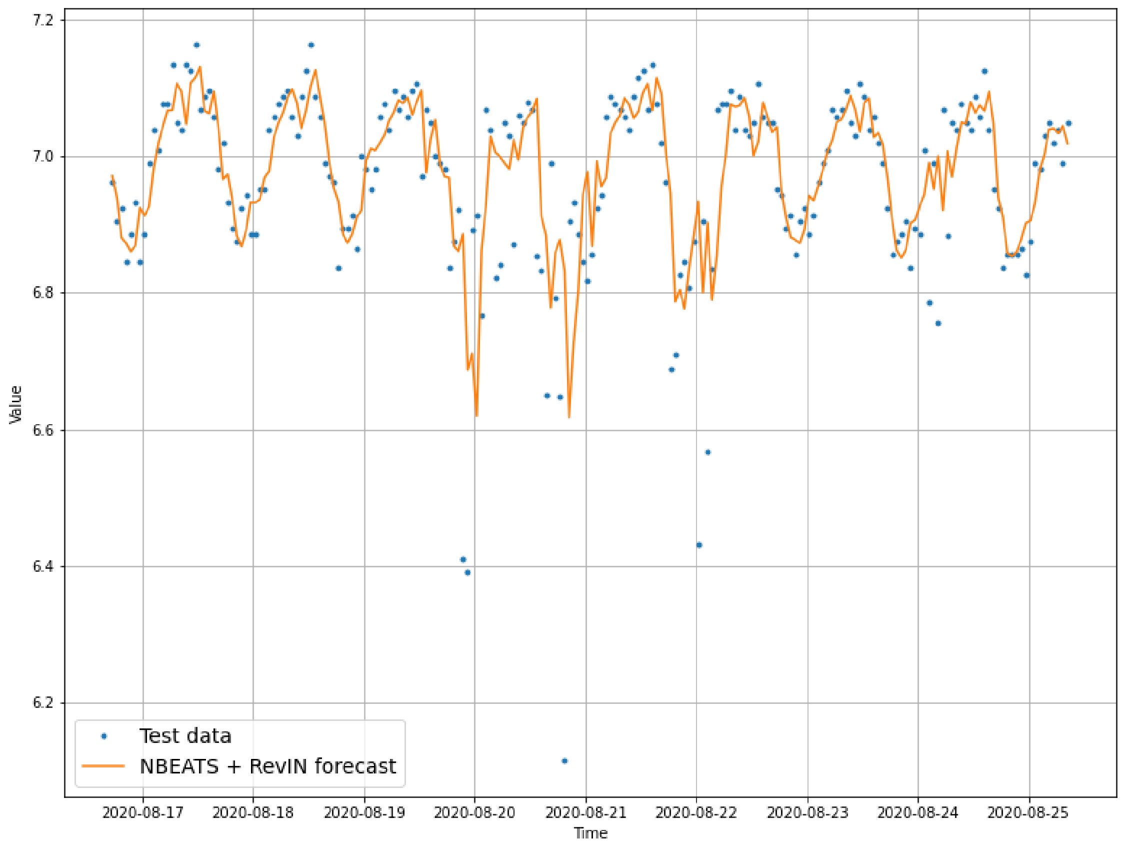
```
[17]: {'mae': 0.06465431,
       'mse': 0.011713349,
       'rmse': 0.10822823,
       'mape': 0.94714284,
       'mase': 0.87623537}
```

```
[18]: import matplotlib.pyplot as plt
      plt.figure(figsize=(13, 10))
      ensemble_median = np.median(ensemble_preds, axis=0)
      offset = 0 # offset the values by 300 timesteps
      plot_time_series(timesteps=X_train.index[offset:], values=y_train[offset:],␣
       ↪label="Train data")
```

```
plot_time_series(timesteps=X_test.index[offset:], values=y_test[offset:],␣
↪format = '.', start=offset, label="Test data")
plot_time_series(timesteps=X_test.index[offset:], values=ensemble_median[offset:
↪], start=offset, label="NBEAT+RevIN forecast");
```



[19]:
```
plt.figure(figsize=(13, 10))
offset = 0 # offset the values by 300 timesteps
plot_time_series(timesteps=X_test.index[offset:], values=y_test[offset:],␣
↪format = ".", start=offset, label="Test data")
plot_time_series(timesteps=X_test.index[offset:], values=ensemble_median[offset:
↪], format="-", start=offset, label="NBEATS + RevIN forecast");
```



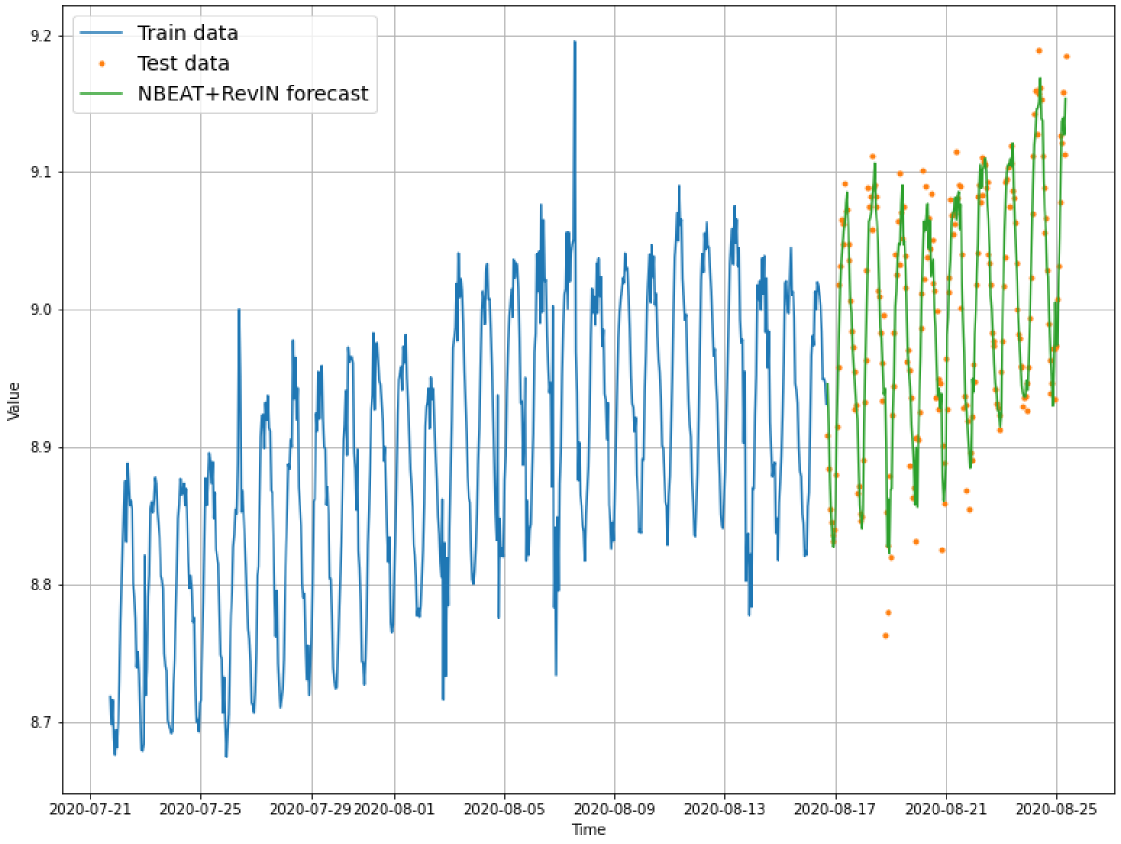
```
[10]: #CSV - 11
```

```
[17]: import numpy as np
      # Evaluate ensemble model(s) predictions
      ensemble_results = evaluate_preds(y_true=y_test,
                                        y_pred=np.median(ensemble_preds, axis=0)) #␣
       ↪take the median across all ensemble predictions
      ensemble_results
```

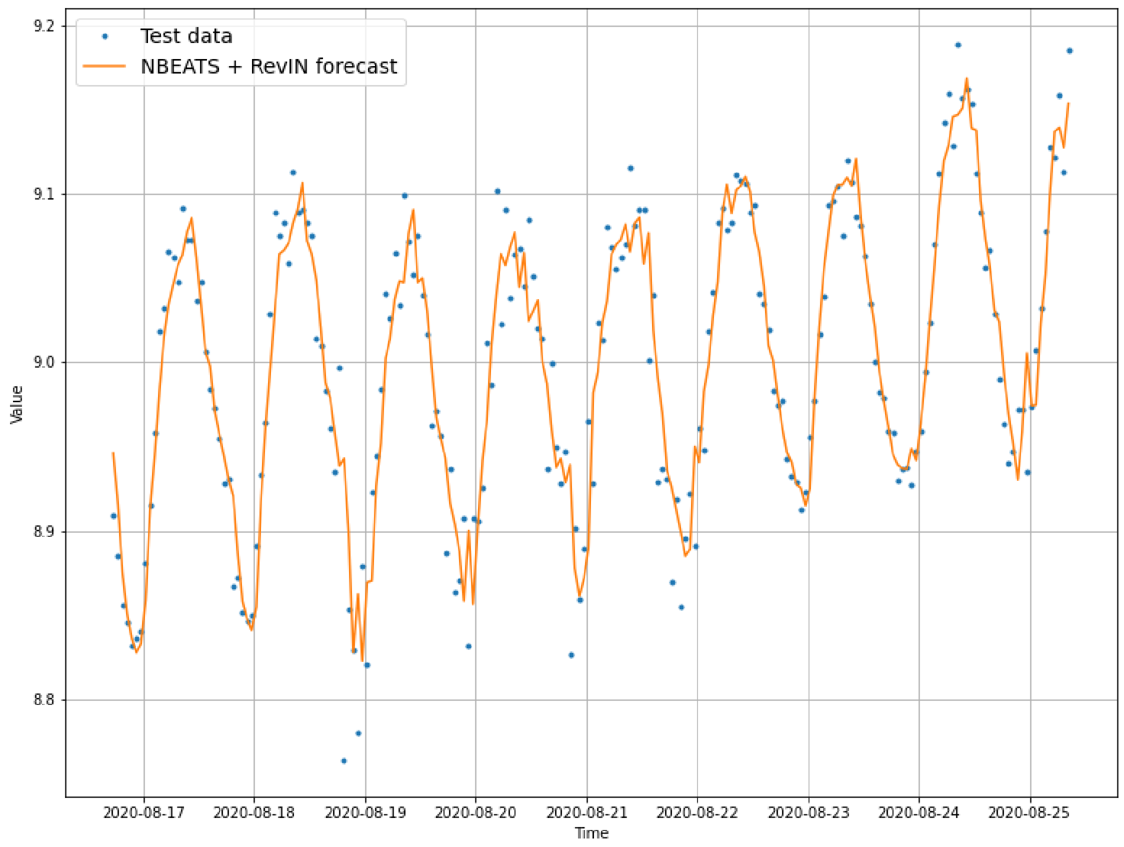
```
[17]: {'mae': 0.02240309,
       'mse': 0.0009771561,
       'rmse': 0.031259496,
       'mape': 0.24956898,
       'mase': 0.6824473}
```

```
[18]: import matplotlib.pyplot as plt
      plt.figure(figsize=(13, 10))
      ensemble_median = np.median(ensemble_preds, axis=0)
      offset = 0 # offset the values by 300 timesteps
```

```
plot_time_series(timesteps=X_train.index[offset:], values=y_train[offset:],␣
 ↪label="Train data")
plot_time_series(timesteps=X_test.index[offset:], values=y_test[offset:],␣
 ↪format = '.', start=offset, label="Test data")
plot_time_series(timesteps=X_test.index[offset:], values=ensemble_median[offset:
 ↪], start=offset, label="NBEAT+RevIN forecast");
```



[19]:
```
plt.figure(figsize=(13, 10))
offset = 0 # offset the values by 300 timesteps
plot_time_series(timesteps=X_test.index[offset:], values=y_test[offset:],␣
 ↪format = ".", start=offset, label="Test data")
plot_time_series(timesteps=X_test.index[offset:], values=ensemble_median[offset:
 ↪], format="-", start=offset, label="NBEATS + RevIN forecast");
```



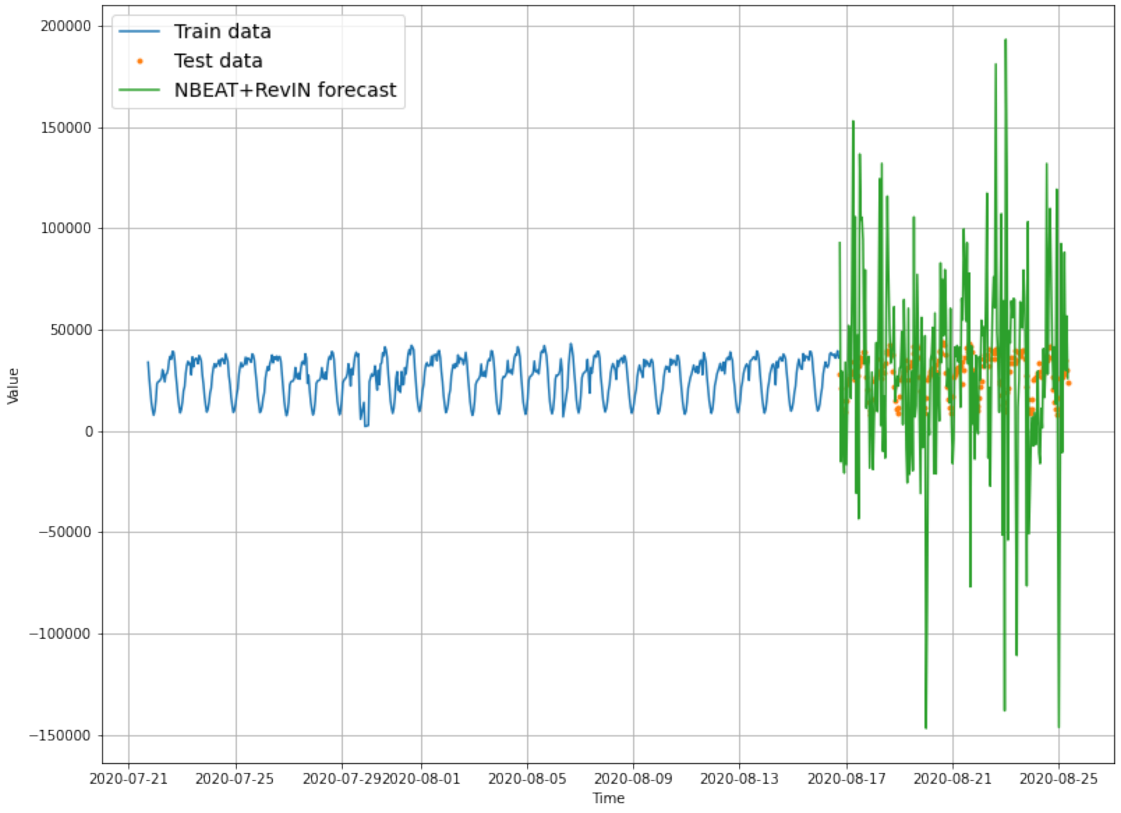
```
[11]: #CSV - 12
```

```
[17]: import numpy as np
      # Evaluate ensemble model(s) predictions
      ensemble_results = evaluate_preds(y_true=y_test,
                                        y_pred=np.median(ensemble_preds, axis=0)) #␣
       ↪take the median across all ensemble predictions
      ensemble_results
```

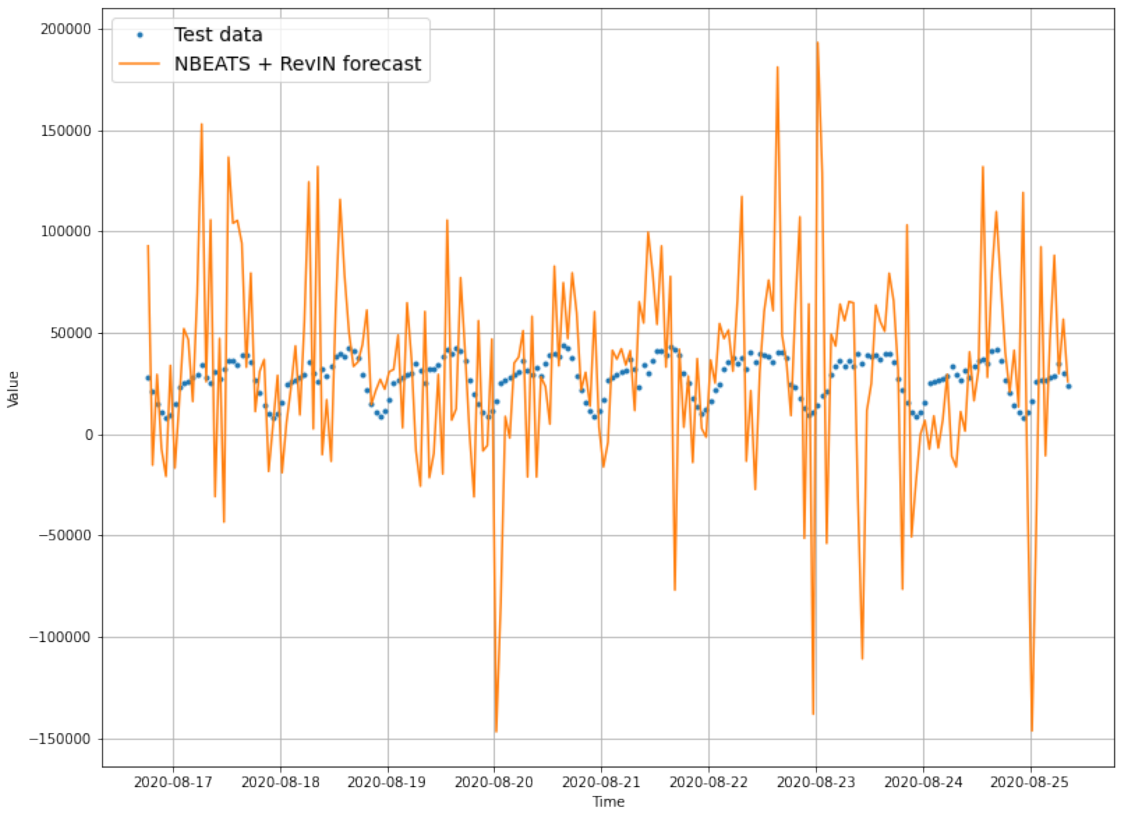
```
[17]: {'mae': 35313.027,
       'mse': 2374639000.0,
       'rmse': 48730.27,
       'mape': 156.7899,
       'mase': 8.973231}
```

```
[18]: import matplotlib.pyplot as plt
      plt.figure(figsize=(13, 10))
      ensemble_median = np.median(ensemble_preds, axis=0)
      offset = 0 # offset the values by 300 timesteps
```

```
plot_time_series(timesteps=X_train.index[offset:], values=y_train[offset:],␣
↪label="Train data")
plot_time_series(timesteps=X_test.index[offset:], values=y_test[offset:],␣
↪format = '.', start=offset, label="Test data")
plot_time_series(timesteps=X_test.index[offset:], values=ensemble_median[offset:
↪], start=offset, label="NBEAT+RevIN forecast");
```



[19]:
```
plt.figure(figsize=(13, 10))
offset = 0 # offset the values by 300 timesteps
plot_time_series(timesteps=X_test.index[offset:], values=y_test[offset:],␣
↪format = ".", start=offset, label="Test data")
plot_time_series(timesteps=X_test.index[offset:], values=ensemble_median[offset:
↪], format="-", start=offset, label="NBEATS + RevIN forecast");
```



```
[15]: #CSV - 14 - Results could have been better with different window sizes
```

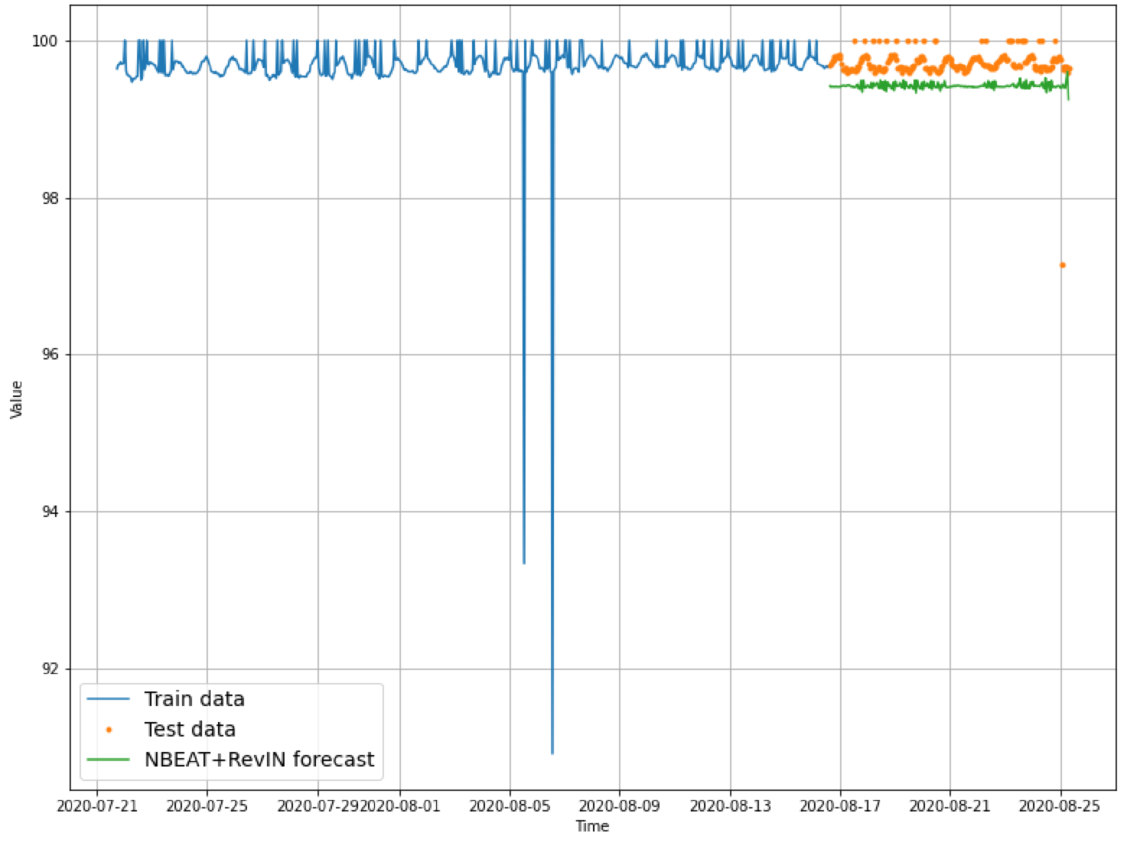
```
[22]: import numpy as np
      # Evaluate ensemble model(s) predictions
      ensemble_results = evaluate_preds(y_true=y_test,
                                        y_pred=np.median(ensemble_preds, axis=0)) #␣
       ↪take the median across all ensemble predictions
```

```
[23]: ensemble_results
```

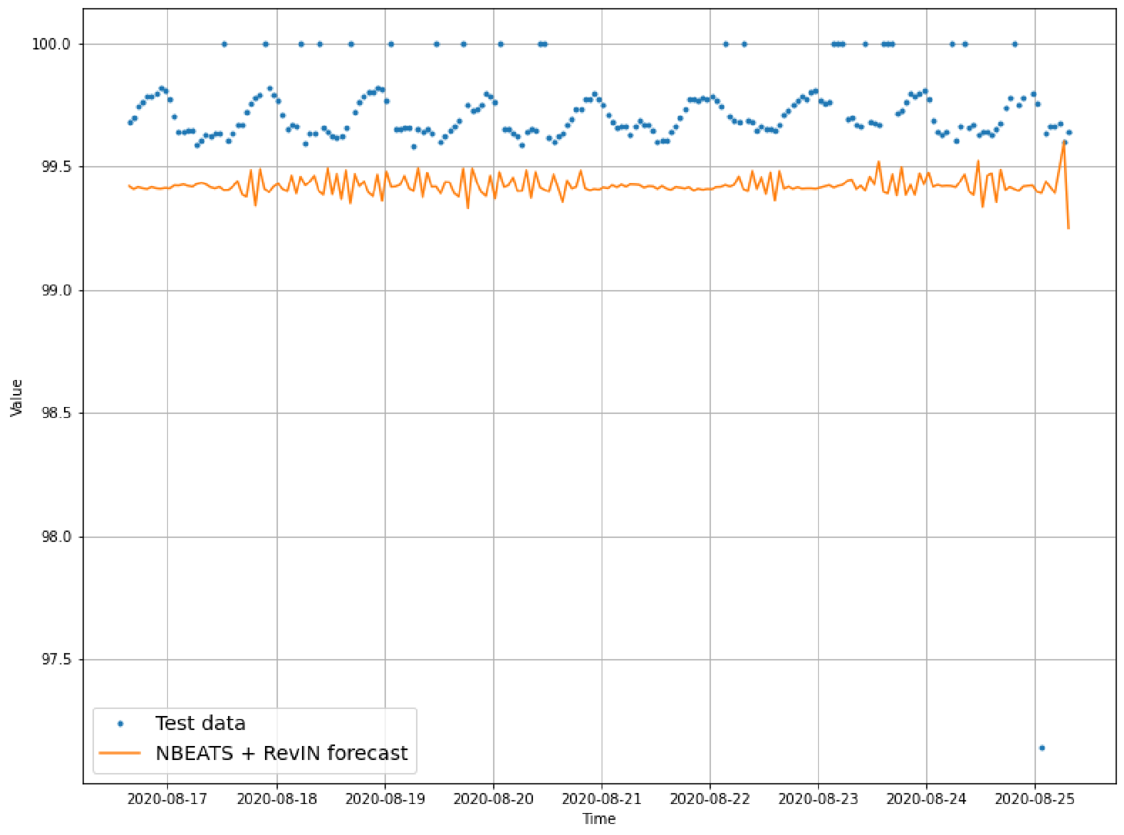
```
[23]: {'mae': 0.31690943,
       'mse': 0.13312207,
       'rmse': 0.36485896,
       'mape': 0.31792444,
       'mase': 3.2692225}
```

```
[24]: import matplotlib.pyplot as plt
      plt.figure(figsize=(13, 10))
      ensemble_median = np.median(ensemble_preds, axis=0)
      offset = 0 # offset the values by 300 timesteps
```

```
plot_time_series(timesteps=X_train.index[offset:], values=y_train[offset:],␣
 ↪label="Train data")
plot_time_series(timesteps=X_test.index[offset:], values=y_test[offset:],␣
 ↪format = '.', start=offset, label="Test data")
plot_time_series(timesteps=X_test.index[offset:], values=ensemble_median[offset:
 ↪], start=offset, label="NBEAT+RevIN forecast");
```



[25]:
```
plt.figure(figsize=(13, 10))
offset = 0 # offset the values by 300 timesteps
plot_time_series(timesteps=X_test.index[offset:], values=y_test[offset:],␣
 ↪format = ".", start=offset, label="Test data")
plot_time_series(timesteps=X_test.index[offset:], values=ensemble_median[offset:
 ↪], format="-", start=offset, label="NBEATS + RevIN forecast");
```



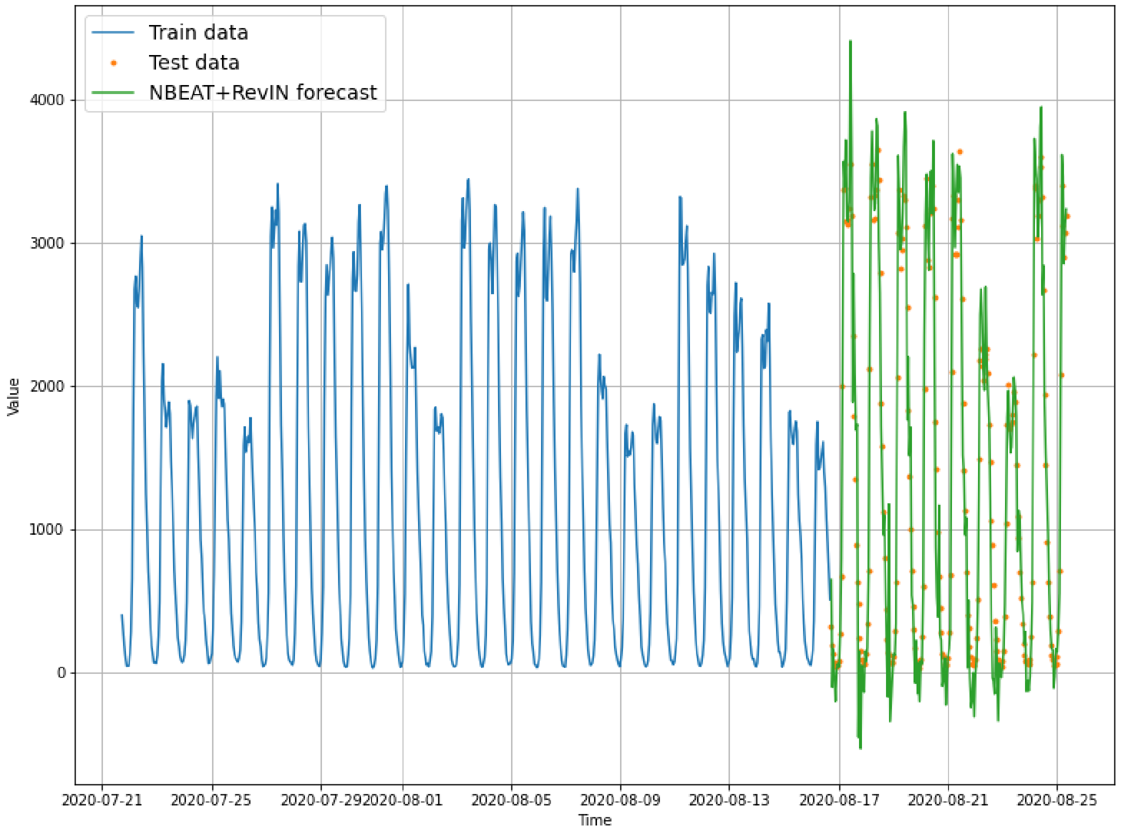
```
[13]: #CSV - 46
```

```
[29]: import numpy as np
      # Evaluate ensemble model(s) predictions
      ensemble_results = evaluate_preds(y_true=y_test,
                                        y_pred=np.median(ensemble_preds, axis=0)) #␣
       ↪take the median across all ensemble predictions
      ensemble_results
```

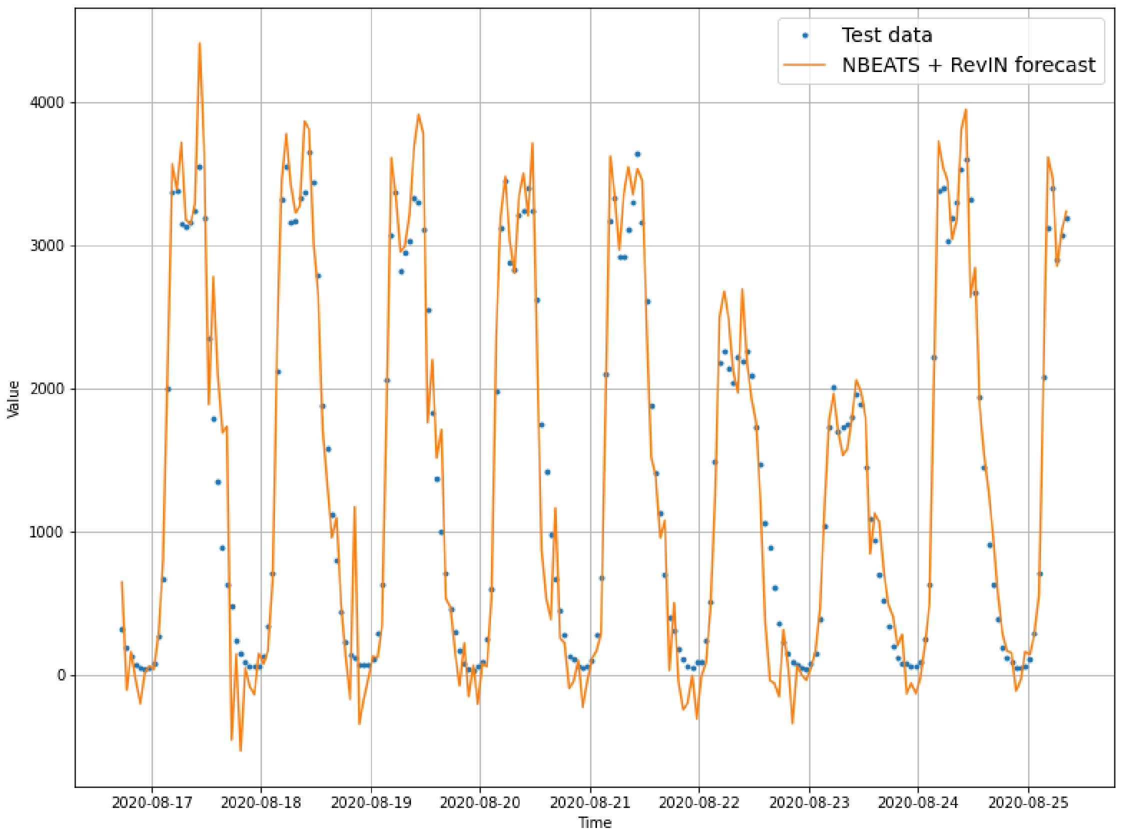
```
[29]: {'mae': 236.73495,
       'mse': 108310.95,
       'rmse': 329.1063,
       'mape': 69.8281,
       'mase': 0.8075697}
```

```
[30]: import matplotlib.pyplot as plt
      plt.figure(figsize=(13, 10))
      ensemble_median = np.median(ensemble_preds, axis=0)
      offset = 0 # offset the values by 300 timesteps
      plot_time_series(timesteps=X_train.index[offset:], values=y_train[offset:],␣
       ↪label="Train data")
```

```
plot_time_series(timesteps=X_test.index[offset:], values=y_test[offset:],␣
↪format = '.', start=offset, label="Test data")
plot_time_series(timesteps=X_test.index[offset:], values=ensemble_median[offset:
↪], start=offset, label="NBEAT+RevIN forecast");
```



[31]:
```
plt.figure(figsize=(13, 10))
offset = 0 # offset the values by 300 timesteps
plot_time_series(timesteps=X_test.index[offset:], values=y_test[offset:],␣
↪format = ".", start=offset, label="Test data")
plot_time_series(timesteps=X_test.index[offset:], values=ensemble_median[offset:
↪], format="-", start=offset, label="NBEATS + RevIN forecast");
```



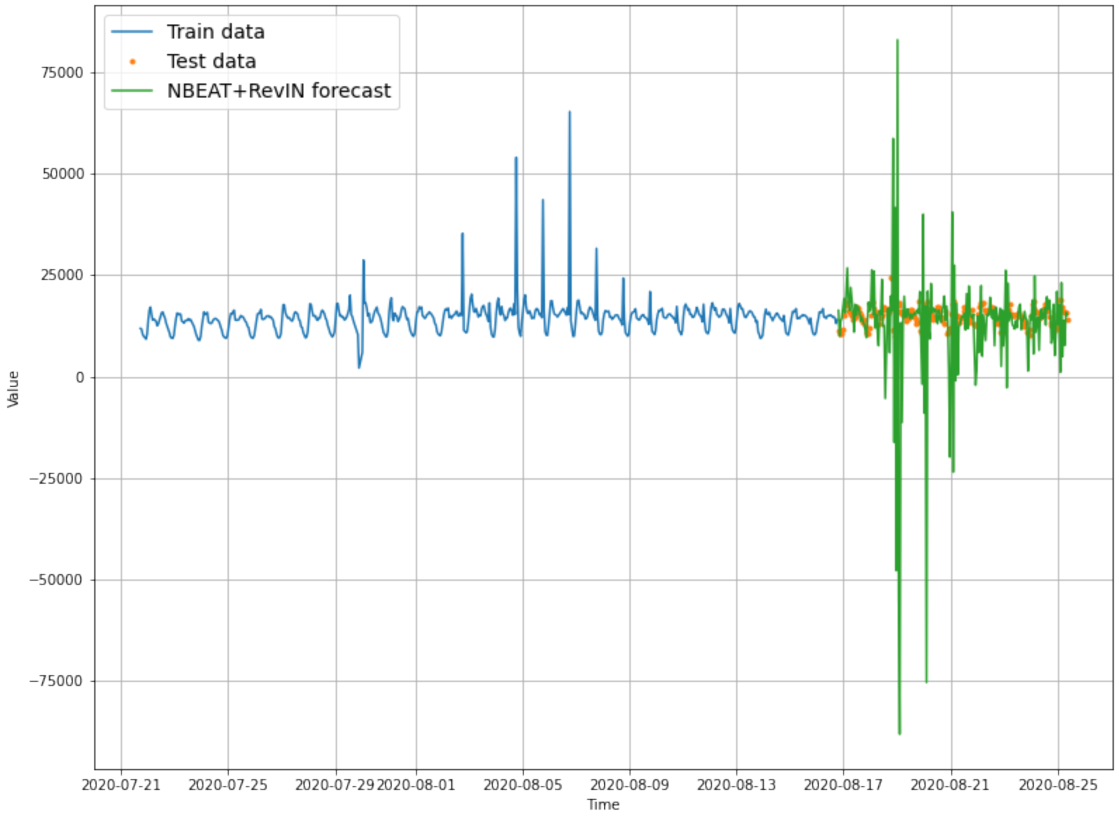
[14]:
```
#CSV - 48
```

[36]:
```
import numpy as np
# Evaluate ensemble model(s) predictions
ensemble_results = evaluate_preds(y_true=y_test,
                                  y_pred=np.median(ensemble_preds, axis=0)) #␣
 ↪take the median across all ensemble predictions
ensemble_results
```

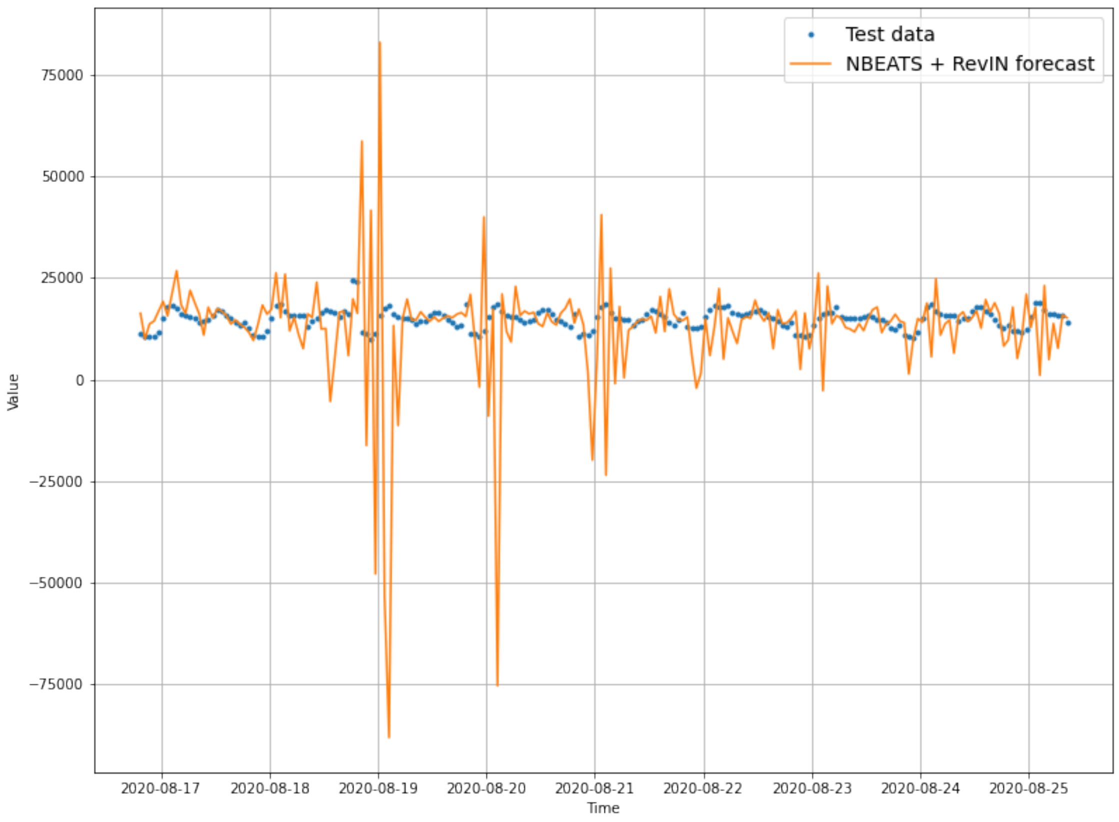
[36]:
```
{'mae': 7045.1665,
 'mse': 236219950.0,
 'rmse': 15369.448,
 'mape': 48.01327,
 'mase': 6.2121615}
```

[37]:
```
import matplotlib.pyplot as plt
plt.figure(figsize=(13, 10))
ensemble_median = np.median(ensemble_preds, axis=0)
offset = 0 # offset the values by 300 timesteps
plot_time_series(timesteps=X_train.index[offset:], values=y_train[offset:],␣
 ↪label="Train data")
```

```
plot_time_series(timesteps=X_test.index[offset:], values=y_test[offset:],␣
↪format = '.', start=offset, label="Test data")
plot_time_series(timesteps=X_test.index[offset:], values=ensemble_median[offset:
↪], start=offset, label="NBEAT+RevIN forecast");
```



[38]:
```
plt.figure(figsize=(13, 10))
offset = 0 # offset the values by 300 timesteps
plot_time_series(timesteps=X_test.index[offset:], values=y_test[offset:],␣
↪format = ".", start=offset, label="Test data")
plot_time_series(timesteps=X_test.index[offset:], values=ensemble_median[offset:
↪], format="-", start=offset, label="NBEATS + RevIN forecast");
```


Test data
NBEATS + RevIN forecast
75000
50000
25000
0
−25000
−50000
−75000
Value
2020-08-17
2020-08-18
2020-08-19
2020-08-20
2020-08-21
2020-08-22
2020-08-23
2020-08-24
2020-08-25
Time

```
[ ]:
```