In [1]:
```
%config InlineBackend.figure_format = 'retina'
```

Testing Enviornment

In [ ]:
```
import pandas as pd
import numpy as np
```

Running tests

The following section unit-tests most of the code written for the proof of concept prototype.

In [ ]:
```
%run ../test/test_algorithms.py
```

In [ ]:
```
%run ../test/test_dataloader.py
```

In [ ]:
```
%run ../test/test_predictorsI.py
```

In [ ]:
```
%run ../test/test_predictorsII.py
```

In [ ]:
```
%run ../test/test_activate.py
```

Running example of the system

In [ ]:
```
%run ../consensus/algorithms.py
```

In [ ]:
```
%run ../tools/dataloader.py
```

In [ ]:
```
test = DataLoader('aapl', '2009-01-01', '2010-02-10')
```

In [ ]:
```
prices = test.get_close()
```

In [ ]:
```
prices
```

In [ ]:
```
#prices = np.array(prices)
#Len(prices)
```

In [ ]:
```
prices
```

In [ ]:
```
%run ../tools/predictorsI.py
```

In [ ]:
```
op0 = BasicUnivariatePredictor(prices, 25, 7)
op1 = BasicUnivariatePredictor(prices, 25, 7)
op2 = BasicUnivariatePredictor(prices, 25, 7)
op3 = BasicUnivariatePredictor(prices, 25, 7)
```

In [ ]:
```
op0.create_bilstm()
```

In [ ]:
```
op0.model_blueprint()
```

In [ ]:
```
op0.fit_model(10)
```

In [ ]:
```
op0.show_performance()
```

In [ ]:
```
oyea = prices[-26:-1]
#oyea = X[-1]
#oyea
```

In [ ]:
```
nice = op0.predict(oyea)
nice
```


In [ ]:
```
op1.create_lstm()
```

In [ ]:
```
op1.model_blueprint()
```

In [ ]:
```
op1.fit_model(10)
```

In [ ]:
```
op1.show_performance()
```

In [ ]:
```
nice = op1.predict(oyea)
nice
```

In [ ]:
```
op2.create_cnn()
```

In [ ]:
```
op2.model_blueprint()
```

In [ ]:
```
op2.fit_model(10)
```

In [ ]:
```
op2.show_performance()
```

In [ ]:
```
nice = op2.predict(oyea)
nice
```

In [ ]:
```
op3.create_mlp()
```

In [ ]:
```
op3.model_blueprint()
```

In [ ]:
```
op3.fit_model(100)
```

In [ ]:
```
op3.show_performance()
```

In [ ]:
```
oyea = prices[-26:-1]
#oyea = X[-1]
#oyea
```

In [ ]:
```
nice = op3.predict(oyea)
nice
```

In [ ]:
```
%run ../tools/predictorsII.py
```

In [ ]:
```
oo = UnivariatePredictorII(prices, 7)
```

In [ ]:
```
oo.fit_neural_model(100,"D")
```

In [ ]:
```
oo.show_performance_neural()
```

In [ ]:
```
oo.predict_neural()
```

In [ ]:
```
oo.fit_prophet_model()
```

In [ ]:
```
oo.show_performance_prophet()
```

In [ ]:
```
oo.predict_prophet()
```

In [ ]:
```
%run ../tools/predictorsIII.py
```

In [ ]:
```
len(prices)
```

In [ ]:
```
op4 = HybridUnivariatePredictor(prices,2, 24, 7)
```

In [ ]:
```
op4.create_cnnlstm()
```

In [ ]:
```
op4.model_blueprint()
```

In [ ]:
```
op4.fit_model(10)
```

In [ ]:
```
op4.show_performance()
```

In [ ]:
```
oyea = prices[-25:-1]
#oyea = X[-1]
#oyea
```

In [ ]:
```
nice = op4.predict(oyea)
nice
#noice = pd.DataFrame(nice, columns=['yea'])
#noice = nice.reshape(20, 1)
#noice = pd.DataFrame(noice, columns=['yea'])
```

In [ ]:
```
nice.plot()
```

Whole system test - I am alive v.2

In [2]:
```
%run ../tools/dataloader.py
%run ../system/activate.py
```

In [3]:
```
training = DataLoader('aapl', '2009-01-01', '2010-05-01')
```

In [4]:
```
training = training.get_close()
```

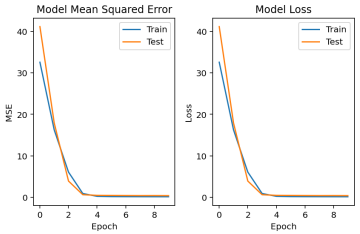
In [5]:
```
predict = DataLoader('aapl', '2010-06-01', '2010-09-01')
```

In [6]:
```
predict = predict.get_close()
```

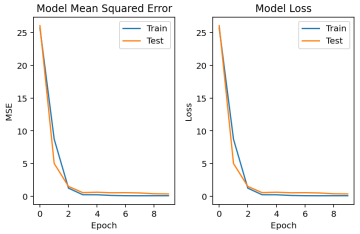
In [7]:
```
predict_req, real = data_prep(predict, 24, 30)
```


In [8]:
```
final_df1 = individual_predictors1(training, predict_req, 30)
```

```
Epoch 1/10
23/23 [==============================] - 0s 20ms/step - loss: 32.5842 - mean_squared_error: 32.5842 - val_loss: 41.1832 - val_mean_squared_error: 41.1832
Epoch 2/10
23/23 [==============================] - 0s 5ms/step - loss: 16.2450 - mean_squared_error: 16.2450 - val_loss: 17.9144 - val_mean_squared_error: 17.9144
Epoch 3/10
23/23 [==============================] - 0s 5ms/step - loss: 6.0527 - mean_squared_error: 6.0527 - val_loss: 3.8894 - val_mean_squared_error: 3.8894
Epoch 4/10
23/23 [==============================] - 0s 5ms/step - loss: 0.8818 - mean_squared_error: 0.8818 - val_loss: 0.5947 - val_mean_squared_error: 0.5947
Epoch 5/10
23/23 [==============================] - 0s 4ms/step - loss: 0.1997 - mean_squared_error: 0.1997 - val_loss: 0.4352 - val_mean_squared_error: 0.4352
Epoch 6/10
23/23 [==============================] - 0s 5ms/step - loss: 0.1369 - mean_squared_error: 0.1369 - val_loss: 0.4154 - val_mean_squared_error: 0.4154
Epoch 7/10
23/23 [==============================] - 0s 5ms/step - loss: 0.1182 - mean_squared_error: 0.1182 - val_loss: 0.3916 - val_mean_squared_error: 0.3916
Epoch 8/10
23/23 [==============================] - 0s 5ms/step - loss: 0.1095 - mean_squared_error: 0.1095 - val_loss: 0.3725 - val_mean_squared_error: 0.3725
Epoch 9/10
23/23 [==============================] - 0s 5ms/step - loss: 0.1019 - mean_squared_error: 0.1019 - val_loss: 0.3794 - val_mean_squared_error: 0.3794
Epoch 10/10
23/23 [==============================] - 0s 4ms/step - loss: 0.0972 - mean_squared_error: 0.0972 - val_loss: 0.3649 - val_mean_squared_error: 0.3649
```

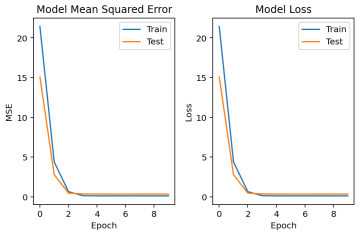



```
Epoch 1/10
23/23 [==============================] - 1s 35ms/step - loss: 25.6772 - mean_squared_error: 25.6772 - val_loss: 26.0422 - val_mean_squared_error: 26.0422
Epoch 2/10
23/23 [==============================] - 0s 17ms/step - loss: 8.7240 - mean_squared_error: 8.7240 - val_loss: 5.0269 - val_mean_squared_error: 5.0269
Epoch 3/10
23/23 [==============================] - 0s 15ms/step - loss: 1.2660 - mean_squared_error: 1.2660 - val_loss: 1.5292 - val_mean_squared_error: 1.5292
Epoch 4/10
23/23 [==============================] - 0s 15ms/step - loss: 0.2611 - mean_squared_error: 0.2611 - val_loss: 0.5839 - val_mean_squared_error: 0.5839
Epoch 5/10
23/23 [==============================] - 0s 15ms/step - loss: 0.2563 - mean_squared_error: 0.2563 - val_loss: 0.6441 - val_mean_squared_error: 0.6441
Epoch 6/10
23/23 [==============================] - 0s 15ms/step - loss: 0.1526 - mean_squared_error: 0.1526 - val_loss: 0.5590 - val_mean_squared_error: 0.5590
Epoch 7/10
23/23 [==============================] - 0s 15ms/step - loss: 0.1136 - mean_squared_error: 0.1136 - val_loss: 0.5727 - val_mean_squared_error: 0.5727
Epoch 8/10
23/23 [==============================] - 0s 15ms/step - loss: 0.1051 - mean_squared_error: 0.1051 - val_loss: 0.5315 - val_mean_squared_error: 0.5315
Epoch 9/10
23/23 [==============================] - 0s 15ms/step - loss: 0.1094 - mean_squared_error: 0.1094 - val_loss: 0.4073 - val_mean_squared_error: 0.4073
Epoch 10/10
23/23 [==============================] - 0s 15ms/step - loss: 0.1112 - mean_squared_error: 0.1112 - val_loss: 0.3823 - val_mean_squared_error: 0.3823
```



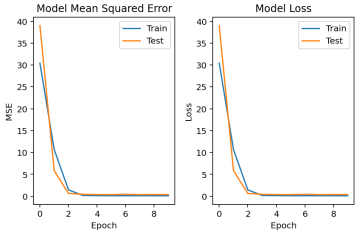

```
Epoch 1/10
23/23 [==============================] - 0s 7ms/step - loss: 21.4271 - mean_squared_error: 21.4271 - val_loss: 15.0839 - val_mean_squared_error: 15.0839
```

```
Epoch 2/10
23/23 [==============================] - 0s 3ms/step - loss: 4.3887 - mean_squared_error: 4.3887 - val_loss: 2.7803 - val_mean_squ
ared_error: 2.7803
Epoch 3/10
23/23 [==============================] - 0s 3ms/step - loss: 0.6439 - mean_squared_error: 0.6439 - val_loss: 0.4615 - val_mean_squ
ared_error: 0.4615
Epoch 4/10
23/23 [==============================] - 0s 3ms/step - loss: 0.1359 - mean_squared_error: 0.1359 - val_loss: 0.3604 - val_mean_squ
ared_error: 0.3604
Epoch 5/10
23/23 [==============================] - 0s 3ms/step - loss: 0.1147 - mean_squared_error: 0.1147 - val_loss: 0.3473 - val_mean_squ
ared_error: 0.3473
Epoch 6/10
23/23 [==============================] - 0s 3ms/step - loss: 0.1117 - mean_squared_error: 0.1117 - val_loss: 0.3347 - val_mean_squ
ared_error: 0.3347
Epoch 7/10
23/23 [==============================] - 0s 3ms/step - loss: 0.1121 - mean_squared_error: 0.1121 - val_loss: 0.3341 - val_mean_squ
ared_error: 0.3341
Epoch 8/10
23/23 [==============================] - 0s 3ms/step - loss: 0.1129 - mean_squared_error: 0.1129 - val_loss: 0.3385 - val_mean_squ
ared_error: 0.3385
Epoch 9/10
23/23 [==============================] - 0s 3ms/step - loss: 0.1099 - mean_squared_error: 0.1099 - val_loss: 0.3394 - val_mean_squ
ared_error: 0.3394
Epoch 10/10
23/23 [==============================] - 0s 3ms/step - loss: 0.1059 - mean_squared_error: 0.1059 - val_loss: 0.3341 - val_mean_squ
ared_error: 0.3341
```

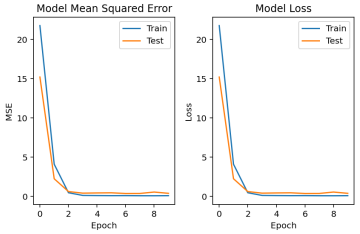


In [9]:
```
final_df2 = individual_predictors2(training, predict_req, 30)
```

```
Epoch 1/10
23/23 [==============================] - 0s 18ms/step - loss: 30.4327 - mean_squared_error: 30.4327 - val_loss: 39.0114 - val_mean_squa
red_error: 39.0114
Epoch 2/10
23/23 [==============================] - 0s 6ms/step - loss: 10.6565 - mean_squared_error: 10.6565 - val_loss: 5.8304 - val_mean_square
d_error: 5.8304
Epoch 3/10
23/23 [==============================] - 0s 5ms/step - loss: 1.4614 - mean_squared_error: 1.4614 - val_loss: 0.6332 - val_mean_squared_
error: 0.6332
Epoch 4/10
23/23 [==============================] - 0s 5ms/step - loss: 0.1748 - mean_squared_error: 0.1748 - val_loss: 0.4367 - val_mean_squared_
error: 0.4367
Epoch 5/10
23/23 [==============================] - 0s 4ms/step - loss: 0.1282 - mean_squared_error: 0.1282 - val_loss: 0.3737 - val_mean_squared_
error: 0.3737
Epoch 6/10
23/23 [==============================] - 0s 5ms/step - loss: 0.1075 - mean_squared_error: 0.1075 - val_loss: 0.3613 - val_mean_squared_
error: 0.3613
Epoch 7/10
23/23 [==============================] - 0s 4ms/step - loss: 0.1094 - mean_squared_error: 0.1094 - val_loss: 0.4452 - val_mean_squared_
error: 0.4452
Epoch 8/10
23/23 [==============================] - 0s 5ms/step - loss: 0.1087 - mean_squared_error: 0.1087 - val_loss: 0.3581 - val_mean_squared_
error: 0.3581
Epoch 9/10
23/23 [==============================] - 0s 6ms/step - loss: 0.1074 - mean_squared_error: 0.1074 - val_loss: 0.3876 - val_mean_squared_
error: 0.3876
Epoch 10/10
23/23 [==============================] - 0s 6ms/step - loss: 0.0999 - mean_squared_error: 0.0999 - val_loss: 0.3736 - val_mean_squared_
error: 0.3736
```

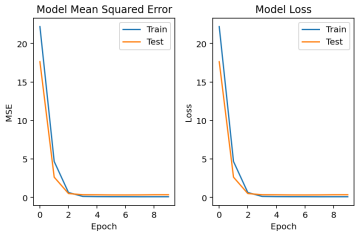



```
Epoch 1/10
23/23 [==============================] - 1s 46ms/step - loss: 21.7670 - mean_squared_error: 21.7670 - val_loss: 15.2118 - val_mean_squa
red_error: 15.2118
Epoch 2/10
23/23 [==============================] - 1s 22ms/step - loss: 4.1029 - mean_squared_error: 4.1029 - val_loss: 2.2510 - val_mean_squared
_error: 2.2510
Epoch 3/10
23/23 [==============================] - 0s 18ms/step - loss: 0.4701 - mean_squared_error: 0.4701 - val_loss: 0.6201 - val_mean_squared
_error: 0.6201
Epoch 4/10
23/23 [==============================] - 0s 18ms/step - loss: 0.1286 - mean_squared_error: 0.1286 - val_loss: 0.4177 - val_mean_squared
_error: 0.4177
Epoch 5/10
23/23 [==============================] - 0s 18ms/step - loss: 0.1098 - mean_squared_error: 0.1098 - val_loss: 0.4434 - val_mean_squared
_error: 0.4434
Epoch 6/10
23/23 [==============================] - 0s 18ms/step - loss: 0.0967 - mean_squared_error: 0.0967 - val_loss: 0.4609 - val_mean_squared
_error: 0.4609
Epoch 7/10
23/23 [==============================] - 0s 18ms/step - loss: 0.1025 - mean_squared_error: 0.1025 - val_loss: 0.3740 - val_mean_squared
_error: 0.3740
Epoch 8/10
23/23 [==============================] - 0s 18ms/step - loss: 0.0875 - mean_squared_error: 0.0875 - val_loss: 0.3785 - val_mean_squared
_error: 0.3785
Epoch 9/10
23/23 [==============================] - 0s 18ms/step - loss: 0.0854 - mean_squared_error: 0.0854 - val_loss: 0.5580 - val_mean_squared
_error: 0.5580
Epoch 10/10
23/23 [==============================] - 0s 18ms/step - loss: 0.0968 - mean_squared_error: 0.0968 - val_loss: 0.3959 - val_mean_squared
_error: 0.3959
```

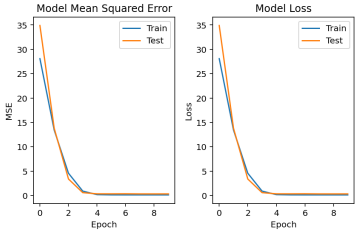



```
Epoch 1/10
23/23 [==============================] - 0s 9ms/step - loss: 22.1979 - mean_squared_error: 22.1979 - val_loss: 17.6415 - val_mean_
squared_error: 17.6415
```

```
Epoch 2/10
23/23 [==============================] - 0s 4ms/step - loss: 4.6799 - mean_squared_error: 4.6799 - val_loss: 2.6459 - val_mean_squ
ared_error: 2.6459
Epoch 3/10
23/23 [==============================] - 0s 4ms/step - loss: 0.6528 - mean_squared_error: 0.6528 - val_loss: 0.5221 - val_mean_squ
ared_error: 0.5221
Epoch 4/10
23/23 [==============================] - 0s 3ms/step - loss: 0.1447 - mean_squared_error: 0.1447 - val_loss: 0.3635 - val_mean_squ
ared_error: 0.3635
Epoch 5/10
23/23 [==============================] - 0s 4ms/step - loss: 0.1180 - mean_squared_error: 0.1180 - val_loss: 0.3556 - val_mean_squ
ared_error: 0.3556
Epoch 6/10
23/23 [==============================] - 0s 3ms/step - loss: 0.1090 - mean_squared_error: 0.1090 - val_loss: 0.3366 - val_mean_squ
ared_error: 0.3366
Epoch 7/10
23/23 [==============================] - 0s 3ms/step - loss: 0.1085 - mean_squared_error: 0.1085 - val_loss: 0.3342 - val_mean_squ
ared_error: 0.3342
Epoch 8/10
23/23 [==============================] - 0s 3ms/step - loss: 0.1018 - mean_squared_error: 0.1018 - val_loss: 0.3414 - val_mean_squ
ared_error: 0.3414
Epoch 9/10
23/23 [==============================] - 0s 3ms/step - loss: 0.1030 - mean_squared_error: 0.1030 - val_loss: 0.3606 - val_mean_squ
ared_error: 0.3606
Epoch 10/10
23/23 [==============================] - 0s 3ms/step - loss: 0.1032 - mean_squared_error: 0.1032 - val_loss: 0.3642 - val_mean_squ
ared_error: 0.3642
```




```
Epoch 1/10
23/23 [==============================] - 0s 6ms/step - loss: 28.0830 - mean_squared_error: 28.0830 - val_loss: 34.8911 - val_mean_squar
ed_error: 34.8911
Epoch 2/10
23/23 [==============================] - 0s 2ms/step - loss: 13.4369 - mean_squared_error: 13.4369 - val_loss: 13.8031 - val_mean_squar
ed_error: 13.8031
Epoch 3/10
23/23 [==============================] - 0s 2ms/step - loss: 4.5677 - mean_squared_error: 4.5677 - val_loss: 3.3817 - val_mean_squared_
error: 3.3817
Epoch 4/10
23/23 [==============================] - 0s 2ms/step - loss: 0.9016 - mean_squared_error: 0.9016 - val_loss: 0.5827 - val_mean_squared_
error: 0.5827
Epoch 5/10
23/23 [==============================] - 0s 3ms/step - loss: 0.1743 - mean_squared_error: 0.1743 - val_loss: 0.3457 - val_mean_squared_
error: 0.3457
Epoch 6/10
23/23 [==============================] - 0s 2ms/step - loss: 0.1245 - mean_squared_error: 0.1245 - val_loss: 0.3423 - val_mean_squared_
error: 0.3423
Epoch 7/10
23/23 [==============================] - 0s 2ms/step - loss: 0.1177 - mean_squared_error: 0.1177 - val_loss: 0.3550 - val_mean_squared_
error: 0.3550
Epoch 8/10
23/23 [==============================] - 0s 2ms/step - loss: 0.1175 - mean_squared_error: 0.1175 - val_loss: 0.3280 - val_mean_squared_
error: 0.3280
Epoch 9/10
23/23 [==============================] - 0s 2ms/step - loss: 0.1175 - mean_squared_error: 0.1175 - val_loss: 0.3263 - val_mean_squared_
error: 0.3263
Epoch 10/10
23/23 [==============================] - 0s 2ms/step - loss: 0.1164 - mean_squared_error: 0.1164 - val_loss: 0.3311 - val_mean_squared_
error: 0.3311
```



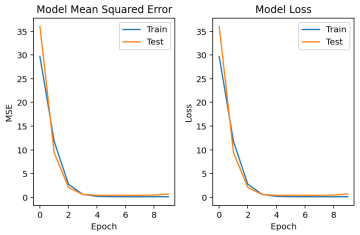

```
WARNING:tensorflow:5 out of the last 5 calls to <function Model.make_predict_function.<locals>.predict_function at 0x0000013E9A88A
820> triggered tf.function retracing. Tracing is expensive and the excessive number of tracings could be due to (1) creating @tf.f
unction repeatedly in a loop, (2) passing tensors with different shapes, (3) passing Python objects instead of tensors. For (1), p
lease define your @tf.function outside of the loop. For (2), @tf.function has experimental_relax_shapes=True option that relaxes a
rgument shapes that can avoid unnecessary retracing. For (3), please refer to https://www.tensorflow.org/tutorials/customization/p
erformance#python_or_tensor_args (https://www.tensorflow.org/tutorials/customization/performance#python_or_tensor_args) and http
s://www.tensorflow.org/api_docs/python/tf/function (https://www.tensorflow.org/api_docs/python/tf/function) for  more details.
WARNING:tensorflow:6 out of the last 6 calls to <function Model.make_predict_function.<locals>.predict_function at 0x0000013E9BF45
EE0> triggered tf.function retracing. Tracing is expensive and the excessive number of tracings could be due to (1) creating @tf.f
```



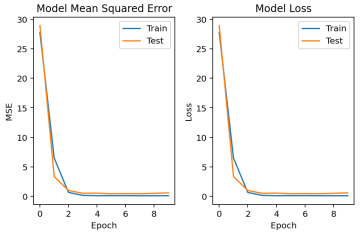
```
unction repeatedly in a loop, (2) passing tensors with different shapes, (3) passing Python objects instead of tensors. For (1), p
lease define your @tf.function outside of the loop. For (2), @tf.function has experimental_relax_shapes=True option that relaxes a
rgument shapes that can avoid unnecessary retracing. For (3), please refer to https://www.tensorflow.org/tutorials/customization/p
erformance#python_or_tensor_args (https://www.tensorflow.org/tutorials/customization/performance#python_or_tensor_args) and http
s://www.tensorflow.org/api_docs/python/tf/function (https://www.tensorflow.org/api_docs/python/tf/function) for  more details.
WARNING:tensorflow:7 out of the last 7 calls to <function Model.make_predict_function.<locals>.predict_function at 0x0000013E9C0CE
C10> triggered tf.function retracing. Tracing is expensive and the excessive number of tracings could be due to (1) creating @tf.f
unction repeatedly in a loop, (2) passing tensors with different shapes, (3) passing Python objects instead of tensors. For (1), p
lease define your @tf.function outside of the loop. For (2), @tf.function has experimental_relax_shapes=True option that relaxes a
rgument shapes that can avoid unnecessary retracing. For (3), please refer to https://www.tensorflow.org/tutorials/customization/p
erformance#python_or_tensor_args (https://www.tensorflow.org/tutorials/customization/performance#python_or_tensor_args) and http
s://www.tensorflow.org/api_docs/python/tf/function (https://www.tensorflow.org/api_docs/python/tf/function) for  more details.
```

In [10]:
```
final_df3 = individual_predictors3(training, predict_req, 30)
```

```
Epoch 1/10
23/23 [==============================] - 0s 20ms/step - loss: 29.6601 - mean_squared_error: 29.6601 - val_loss: 35.9774 - val_mean_squa
red_error: 35.9774
Epoch 2/10
23/23 [==============================] - 0s 6ms/step - loss: 11.7590 - mean_squared_error: 11.7590 - val_loss: 9.3843 - val_mean_square
d_error: 9.3843
Epoch 3/10
23/23 [==============================] - 0s 6ms/step - loss: 2.7463 - mean_squared_error: 2.7463 - val_loss: 2.0723 - val_mean_squared_
error: 2.0723
Epoch 4/10
23/23 [==============================] - 0s 5ms/step - loss: 0.6116 - mean_squared_error: 0.6116 - val_loss: 0.5865 - val_mean_squared_
error: 0.5865
Epoch 5/10
23/23 [==============================] - 0s 5ms/step - loss: 0.1779 - mean_squared_error: 0.1779 - val_loss: 0.3964 - val_mean_squared_
error: 0.3964
Epoch 6/10
23/23 [==============================] - 0s 5ms/step - loss: 0.1137 - mean_squared_error: 0.1137 - val_loss: 0.4152 - val_mean_squared_
error: 0.4152
Epoch 7/10
23/23 [==============================] - 0s 5ms/step - loss: 0.1018 - mean_squared_error: 0.1018 - val_loss: 0.3989 - val_mean_squared_
error: 0.3989
Epoch 8/10
23/23 [==============================] - 0s 6ms/step - loss: 0.0969 - mean_squared_error: 0.0969 - val_loss: 0.3918 - val_mean_squared_
error: 0.3918
Epoch 9/10
23/23 [==============================] - 0s 5ms/step - loss: 0.0973 - mean_squared_error: 0.0973 - val_loss: 0.4505 - val_mean_squared_
error: 0.4505
Epoch 10/10
23/23 [==============================] - 0s 6ms/step - loss: 0.1054 - mean_squared_error: 0.1054 - val_loss: 0.6749 - val_mean_squared_
error: 0.6749
```



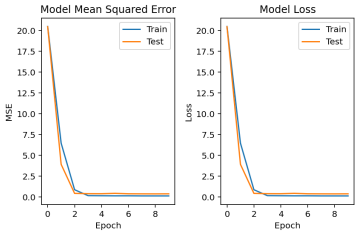

```
Epoch 1/10
23/23 [==============================] - 1s 47ms/step - loss: 27.7333 - mean_squared_error: 27.7333 - val_loss: 28.9184 - val_mean_squa
red_error: 28.9184
Epoch 2/10
23/23 [==============================] - 0s 19ms/step - loss: 6.4702 - mean_squared_error: 6.4702 - val_loss: 3.3519 - val_mean_squared
_error: 3.3519
Epoch 3/10
23/23 [==============================] - 0s 18ms/step - loss: 0.6644 - mean_squared_error: 0.6644 - val_loss: 0.9968 - val_mean_squared
_error: 0.9968
Epoch 4/10
23/23 [==============================] - 0s 17ms/step - loss: 0.1600 - mean_squared_error: 0.1600 - val_loss: 0.5185 - val_mean_squared
_error: 0.5185
Epoch 5/10
23/23 [==============================] - 0s 18ms/step - loss: 0.1114 - mean_squared_error: 0.1114 - val_loss: 0.5455 - val_mean_squared
_error: 0.5455
Epoch 6/10
23/23 [==============================] - 0s 19ms/step - loss: 0.1143 - mean_squared_error: 0.1143 - val_loss: 0.4453 - val_mean_squared
_error: 0.4453
Epoch 7/10
23/23 [==============================] - 0s 18ms/step - loss: 0.1176 - mean_squared_error: 0.1176 - val_loss: 0.4659 - val_mean_squared
_error: 0.4659
Epoch 8/10
23/23 [==============================] - 0s 18ms/step - loss: 0.1012 - mean_squared_error: 0.1012 - val_loss: 0.4399 - val_mean_squared
_error: 0.4399
Epoch 9/10
23/23 [==============================] - 0s 19ms/step - loss: 0.0998 - mean_squared_error: 0.0998 - val_loss: 0.5217 - val_mean_squared
_error: 0.5217
Epoch 10/10
23/23 [==============================] - 0s 19ms/step - loss: 0.1054 - mean_squared_error: 0.1054 - val_loss: 0.5866 - val_mean_squared
_error: 0.5866
```



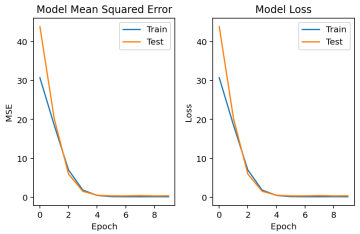

```
Epoch 1/10
23/23 [==============================] - 0s 11ms/step - loss: 20.4467 - mean_squared_error: 20.4467 - val_loss: 20.4242 - val_mean_squa
red_error: 20.4242
```



```
Epoch 2/10
23/23 [==============================] - 0s 4ms/step - loss: 6.4413 - mean_squared_error: 6.4413 - val_loss: 3.8961 - val_mean_squared_
error: 3.8961
Epoch 3/10
23/23 [==============================] - 0s 4ms/step - loss: 0.8489 - mean_squared_error: 0.8489 - val_loss: 0.4101 - val_mean_squared_
error: 0.4101
Epoch 4/10
23/23 [==============================] - 0s 3ms/step - loss: 0.1374 - mean_squared_error: 0.1374 - val_loss: 0.3752 - val_mean_squared_
error: 0.3752
Epoch 5/10
23/23 [==============================] - 0s 4ms/step - loss: 0.1255 - mean_squared_error: 0.1255 - val_loss: 0.3707 - val_mean_squared_
error: 0.3707
Epoch 6/10
23/23 [==============================] - 0s 3ms/step - loss: 0.1150 - mean_squared_error: 0.1150 - val_loss: 0.4154 - val_mean_squared_
error: 0.4154
Epoch 7/10
23/23 [==============================] - 0s 3ms/step - loss: 0.1183 - mean_squared_error: 0.1183 - val_loss: 0.3618 - val_mean_squared_
error: 0.3618
Epoch 8/10
23/23 [==============================] - 0s 3ms/step - loss: 0.1060 - mean_squared_error: 0.1060 - val_loss: 0.3530 - val_mean_squared_
error: 0.3530
Epoch 9/10
23/23 [==============================] - 0s 3ms/step - loss: 0.1047 - mean_squared_error: 0.1047 - val_loss: 0.3492 - val_mean_squared_
error: 0.3492
Epoch 10/10
23/23 [==============================] - 0s 3ms/step - loss: 0.1040 - mean_squared_error: 0.1040 - val_loss: 0.3555 - val_mean_squared_
error: 0.3555
```

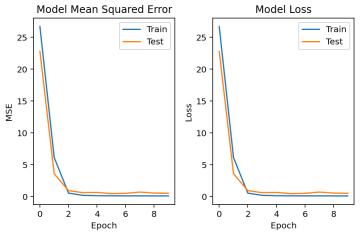



```
Epoch 1/10
23/23 [==============================] - 0s 8ms/step - loss: 30.7411 - mean_squared_error: 30.7411 - val_loss: 43.8812 - val_mean_squar
ed_error: 43.8812
Epoch 2/10
23/23 [==============================] - 0s 2ms/step - loss: 18.5245 - mean_squared_error: 18.5245 - val_loss: 20.2292 - val_mean_squar
ed_error: 20.2292
Epoch 3/10
23/23 [==============================] - 0s 2ms/step - loss: 7.0275 - mean_squared_error: 7.0275 - val_loss: 5.9685 - val_mean_squared_
error: 5.9685
Epoch 4/10
23/23 [==============================] - 0s 2ms/step - loss: 1.8447 - mean_squared_error: 1.8447 - val_loss: 1.5321 - val_mean_squared_
error: 1.5321
Epoch 5/10
23/23 [==============================] - 0s 2ms/step - loss: 0.4682 - mean_squared_error: 0.4682 - val_loss: 0.5169 - val_mean_squared_
error: 0.5169
Epoch 6/10
23/23 [==============================] - 0s 2ms/step - loss: 0.1601 - mean_squared_error: 0.1601 - val_loss: 0.3540 - val_mean_squared_
error: 0.3540
Epoch 7/10
23/23 [==============================] - 0s 2ms/step - loss: 0.1252 - mean_squared_error: 0.1252 - val_loss: 0.3406 - val_mean_squared_
error: 0.3406
Epoch 8/10
23/23 [==============================] - 0s 2ms/step - loss: 0.1211 - mean_squared_error: 0.1211 - val_loss: 0.4376 - val_mean_squared_
error: 0.4376
Epoch 9/10
23/23 [==============================] - 0s 2ms/step - loss: 0.1324 - mean_squared_error: 0.1324 - val_loss: 0.3347 - val_mean_squared_
error: 0.3347
Epoch 10/10
23/23 [==============================] - 0s 2ms/step - loss: 0.1204 - mean_squared_error: 0.1204 - val_loss: 0.3609 - val_mean_squared_
error: 0.3609
```




```
Epoch 1/10
23/23 [==============================] - 1s 50ms/step - loss: 26.6945 - mean_squared_error: 26.6945 - val_loss: 22.7921 - val_mean
_squared_error: 22.7921
Epoch 2/10
23/23 [==============================] - 1s 28ms/step - loss: 6.1267 - mean_squared_error: 6.1267 - val_loss: 3.5776 - val_mean_sq
uared_error: 3.5776
Epoch 3/10
23/23 [==============================] - 1s 24ms/step - loss: 0.5366 - mean_squared_error: 0.5366 - val_loss: 0.9336 - val_mean_sq
uared_error: 0.9336
Epoch 4/10
```

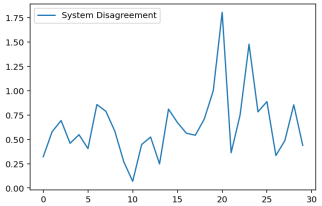
```
23/23 [==============================] - 1s 24ms/step - loss: 0.1861 - mean_squared_error: 0.1861 - val_loss: 0.6075 - val_mean_sq
uared_error: 0.6075
Epoch 5/10
23/23 [==============================] - 1s 23ms/step - loss: 0.1302 - mean_squared_error: 0.1302 - val_loss: 0.6424 - val_mean_sq
uared_error: 0.6424
Epoch 6/10
23/23 [==============================] - 1s 22ms/step - loss: 0.1103 - mean_squared_error: 0.1103 - val_loss: 0.4573 - val_mean_sq
uared_error: 0.4573
Epoch 7/10
23/23 [==============================] - 1s 23ms/step - loss: 0.1014 - mean_squared_error: 0.1014 - val_loss: 0.5013 - val_mean_sq
uared_error: 0.5013
Epoch 8/10
23/23 [==============================] - 1s 26ms/step - loss: 0.1066 - mean_squared_error: 0.1066 - val_loss: 0.6955 - val_mean_sq
uared_error: 0.6955
Epoch 9/10
23/23 [==============================] - 1s 22ms/step - loss: 0.0977 - mean_squared_error: 0.0977 - val_loss: 0.5426 - val_mean_sq
uared_error: 0.5426
Epoch 10/10
23/23 [==============================] - 1s 23ms/step - loss: 0.0945 - mean_squared_error: 0.0945 - val_loss: 0.5119 - val_mean_sq
uared_error: 0.5119
```

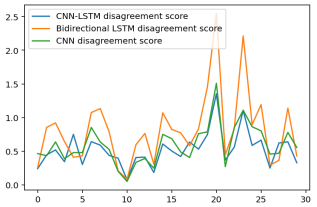



```
WARNING:tensorflow:8 out of the last 8 calls to <function Model.make_predict_function.<locals>.predict_function at 0x0000013EA7628820>
triggered tf.function retracing. Tracing is expensive and the excessive number of tracings could be due to (1) creating @tf.function re
peatedly in a loop, (2) passing tensors with different shapes, (3) passing Python objects instead of tensors. For (1), please define yo
ur @tf.function outside of the loop. For (2), @tf.function has experimental_relax_shapes=True option that relaxes argument shapes that
can avoid unnecessary retracing. For (3), please refer to https://www.tensorflow.org/tutorials/customization/performance#python_or_tens
or_args (https://www.tensorflow.org/tutorials/customization/performance#python_or_tensor_args) and https://www.tensorflow.org/api_docs/
python/tf/function (https://www.tensorflow.org/api_docs/python/tf/function) for  more details.
WARNING:tensorflow:9 out of the last 9 calls to <function Model.make_predict_function.<locals>.predict_function at 0x0000013EA772EEE0>
 triggered tf.function retracing. Tracing is expensive and the excessive number of tracings could be due to (1) creating @tf.function r
epeatedly in a loop, (2) passing tensors with different shapes, (3) passing Python objects instead of tensors. For (1), please define y
our @tf.function outside of the loop. For (2), @tf.function has experimental_relax_shapes=True option that relaxes argument shapes that
can avoid unnecessary retracing. For (3), please refer to https://www.tensorflow.org/tutorials/customization/performance#python_or_tens
or_args (https://www.tensorflow.org/tutorials/customization/performance#python_or_tensor_args) and https://www.tensorflow.org/api_docs/
python/tf/function (https://www.tensorflow.org/api_docs/python/tf/function) for  more details.
WARNING:tensorflow:10 out of the last 10 calls to <function Model.make_predict_function.<locals>.predict_function at 0x0000013EA7D5DA60
> triggered tf.function retracing. Tracing is expensive and the excessive number of tracings could be due to (1) creating @tf.function
 repeatedly in a loop, (2) passing tensors with different shapes, (3) passing Python objects instead of tensors. For (1), please define
your @tf.function outside of the loop. For (2), @tf.function has experimental_relax_shapes=True option that relaxes argument shapes tha
t can avoid unnecessary retracing. For (3), please refer to https://www.tensorflow.org/tutorials/customization/performance#python_or_te
nsor_args (https://www.tensorflow.org/tutorials/customization/performance#python_or_tensor_args) and https://www.tensorflow.org/api_doc
s/python/tf/function (https://www.tensorflow.org/api_docs/python/tf/function) for  more details.
WARNING:tensorflow:11 out of the last 11 calls to <function Model.make_predict_function.<locals>.predict_function at 0x0000013EA8DC1790
> triggered tf.function retracing. Tracing is expensive and the excessive number of tracings could be due to (1) creating @tf.function
 repeatedly in a loop, (2) passing tensors with different shapes, (3) passing Python objects instead of tensors. For (1), please define
your @tf.function outside of the loop. For (2), @tf.function has experimental_relax_shapes=True option that relaxes argument shapes tha
t can avoid unnecessary retracing. For (3), please refer to https://www.tensorflow.org/tutorials/customization/performance#python_or_te
nsor_args (https://www.tensorflow.org/tutorials/customization/performance#python_or_tensor_args) and https://www.tensorflow.org/api_doc
s/python/tf/function (https://www.tensorflow.org/api_docs/python/tf/function) for  more details.
WARNING:tensorflow:11 out of the last 11 calls to <function Model.make_predict_function.<locals>.predict_function at 0x0000013EA8E913A0
> triggered tf.function retracing. Tracing is expensive and the excessive number of tracings could be due to (1) creating @tf.function
 repeatedly in a loop, (2) passing tensors with different shapes, (3) passing Python objects instead of tensors. For (1), please define
your @tf.function outside of the loop. For (2), @tf.function has experimental_relax_shapes=True option that relaxes argument shapes tha
t can avoid unnecessary retracing. For (3), please refer to https://www.tensorflow.org/tutorials/customization/performance#python_or_te
nsor_args (https://www.tensorflow.org/tutorials/customization/performance#python_or_tensor_args) and https://www.tensorflow.org/api_doc
s/python/tf/function (https://www.tensorflow.org/api_docs/python/tf/function) for  more details.
```

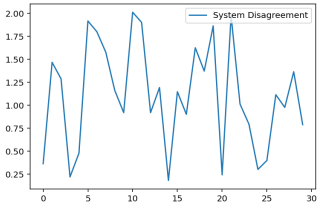
System Disagreement

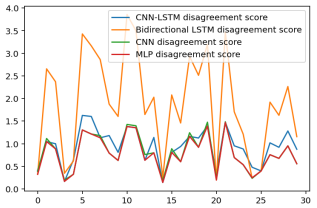
In [11]:
```
system_disagreement(final_df1)
```



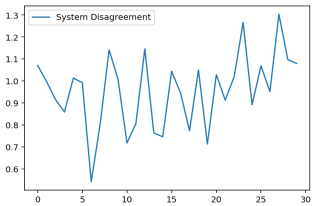


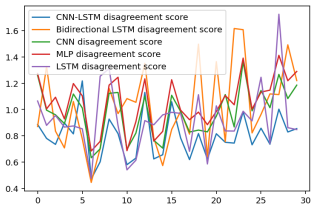
In [12]:
```
system_disagreement(final_df2)
```





In [13]:
```
system_disagreement(final_df3)
```





System consensus

In [14]:
```
algos1 = consensus(final_df1, real)
```

In [15]:
```
algos2 = consensus(final_df2, real)
```

In [16]:
```
algos3 = consensus(final_df3, real)
```

In [17]:
```
ui1 = combined_frame(final_df1, algos1, real)
```

In [18]:
```
ui2 = combined_frame(final_df2, algos2, real)
```

In [19]:
```
ui3 = combined_frame(final_df3, algos3, real)
```

In [20]:
```
mse_score(ui1)
```

Out[20]:
```
([('Average', 0.20496873722722495),
  ('NoMemory', 0.17699168687180886),
  ('Memory', 0.1807511369165537),
  ('Focus', 0.27487009835596116),
  ('Anchor', 0.17272885706646157)],
 [('CNN-LSTM', 0.1841403082654324),
  ('Bidirectional LSTM', 1.446602981119251),
  ('CNN', 0.5410572280319987)])
```

In [21]:
```
mse_log_score(ui1)
```

Out[21]:
```
([('Average', 0.0019427222820790549),
  ('NoMemory', 0.0016761030717148979),
  ('Memory', 0.0017032524822600 56),
  ('Focus', 0.0027141724306226495),
  ('Anchor', 0.0016299279853178518)],
 [('CNN-LSTM', 0.0017542505116592947),
  ('Bidirectional LSTM', 0.016622518690 35305),
  ('CNN', 0.004885966994636406)])
```

In [22]:
```
mae_score(ui1)
```

Out[22]:
```
([('Average', 0.35450726085239),
  ('NoMemory', 0.34008221816281403),
  ('Memory', 0.3367090159133964),
  ('Focus', 0.4192292478349474),
  ('Anchor', 0.33545853111731383)],
 [('CNN-LSTM', 0.3436264991760254),
  ('Bidirectional LSTM', 1.003633197148641),
  ('CNN', 0.6664865811665853)])
```

In [23]:
```
mse_score(ui2)
```

Out[23]:
```
([('Average', 1.0157362131132046),
  ('NoMemory', 0.6758191098381787),
  ('Memory', 0.6531138772786519),
  ('Focus', 0.7145915087670289),
  ('Anchor', 1.1485441036477442)],
 [('CNN-LSTM', 0.11780226625281405),
  ('Bidirectional LSTM', 9.313614535802458),
  ('CNN', 0.6649586830728367),
  ('MLP', 0.6358286175522153)])
```

In [24]:
```
mse_log_score(ui2)
```

Out[24]:
```
([('Average', 0.008970078799601412),
  ('NoMemory', 0.006090216219721094),
  ('Memory', 0.0058652111380590 99),
  ('Focus', 0.009039145253608693),
  ('Anchor', 0.010070528451976982)],
 [('CNN-LSTM', 0.0011200422730614616),
  ('Bidirectional LSTM', 0.09643297690189197),
  ('CNN', 0.0059741107700032 08),
  ('MLP', 0.005707803552718884)])
```

In [25]:
```
mae_score(ui2)
```

Out[25]:
```
([('Average', 0.8808257063229878),
  ('NoMemory', 0.6816583258462536),
  ('Memory', 0.6658055645029967),
  ('Focus', 0.490217185020446 76),
  ('Anchor', 0.9270972578474819)],
 [('CNN-LSTM', 0.28057549794514974),
  ('Bidirectional LSTM', 2.698656956354777),
  ('CNN', 0.767161432902018 2),
  ('MLP', 0.7306918462117513)])
```


In [26]:
```
mse_score(ui3)
```

Out[26]:
```
([('Average', 0.13776263735565283),
  ('NoMemory', 0.11259251307942982),
  ('Memory', 0.11603360609926917),
  ('Focus', 0.37688356596347633),
  ('Anchor', 0.08611763139356717)],
 [('CNN-LSTM', 0.6816633545102074),
  ('Bidirectional LSTM', 1.50741997003191 64),
  ('CNN', 0.5501653545551562),
  ('MLP', 0.7445143001637613),
  ('LSTM', 1.2289150522068135)])
```

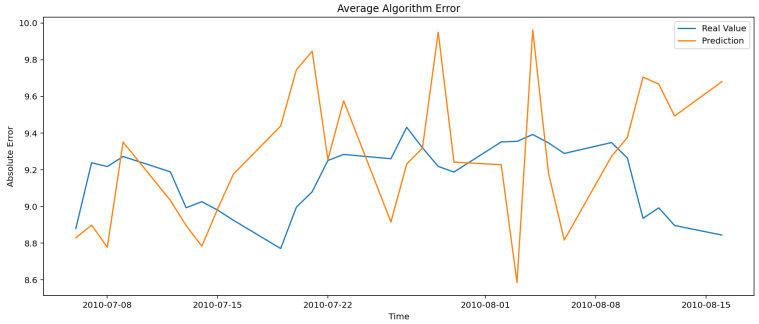
In [27]:
```
mse_log_score(ui3)
```

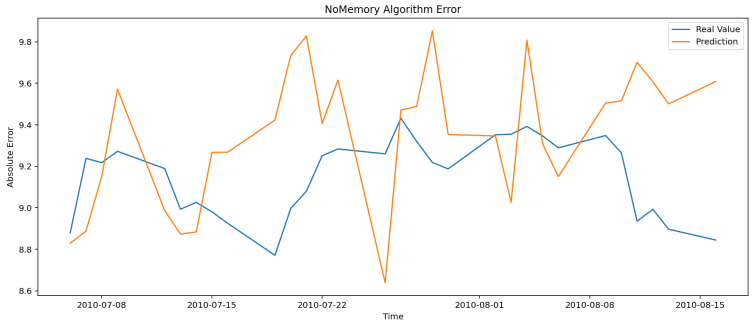
Out[27]:
```
([('Average', 0.0014016761086117033),
  ('NoMemory', 0.0011404311027858004),
  ('Memory', 0.0011761746618706682),
  ('Focus', 0.0037583785762367945),
  ('Anchor', 0.0008629343409986154)],
 [('CNN-LSTM', 0.007487505401257654),
  ('Bidirectional LSTM', 0.017040263047827797),
  ('CNN', 0.004980738693844787),
  ('MLP', 0.006657697960523836),
  ('LSTM', 0.013765194132793568)])
```

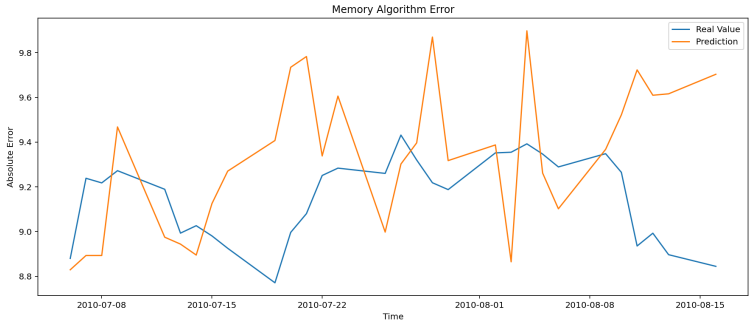
In [28]:
```
mae_score(ui3)
```

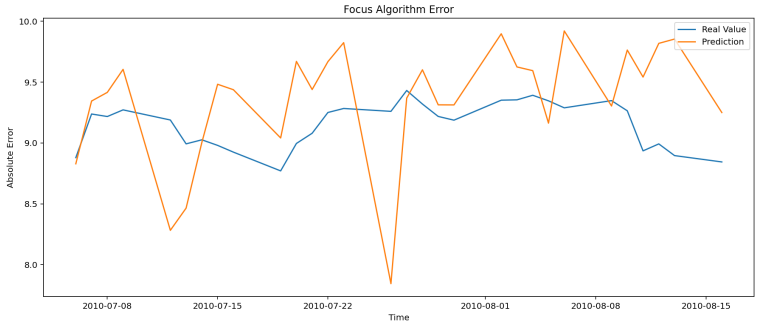
Out[28]:
```
([('Average', 0.2923722839355467),
  ('NoMemory', 0.2621894657297 9435),
  ('Memory', 0.2603477820303833),
  ('Focus', 0.5395410410563152),
  ('Anchor', 0.2235011986588768)],
 [('CNN-LSTM', 0.6469920794169108),
  ('Bidirectional LSTM', 1.12891147931411 84),
  ('CNN', 0.6934393882751465),
  ('MLP', 0.8120713233947754),
  ('LSTM', 0.9920098145802816)])
```

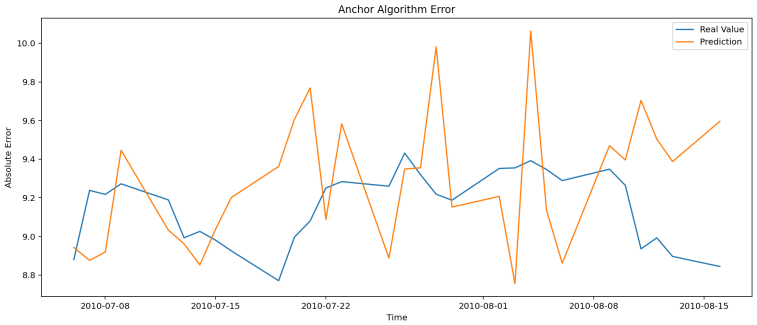
In [29]:
```
plot_performance(ui1)
```

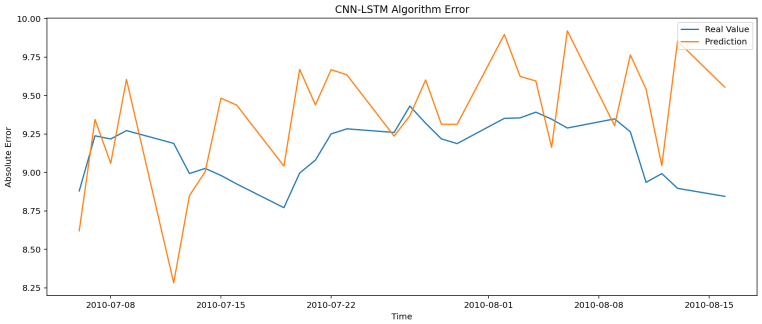


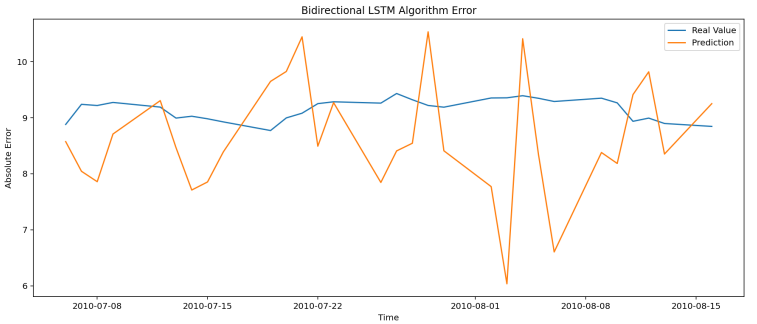


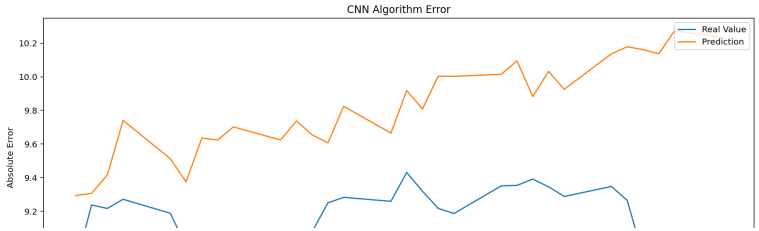




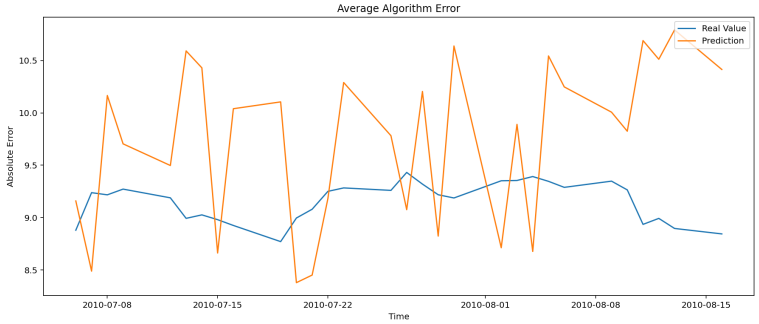


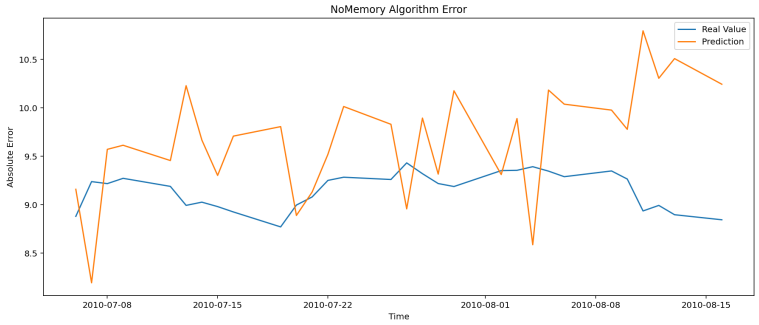


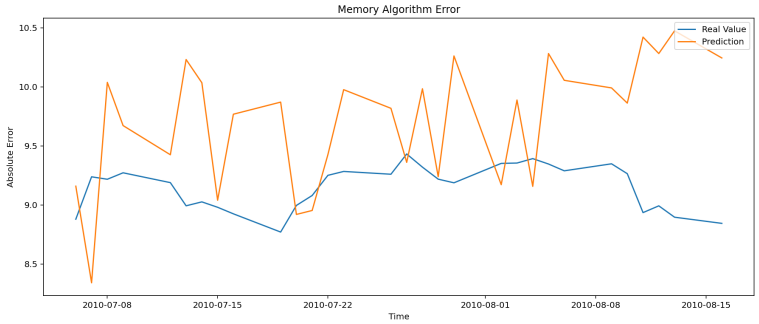


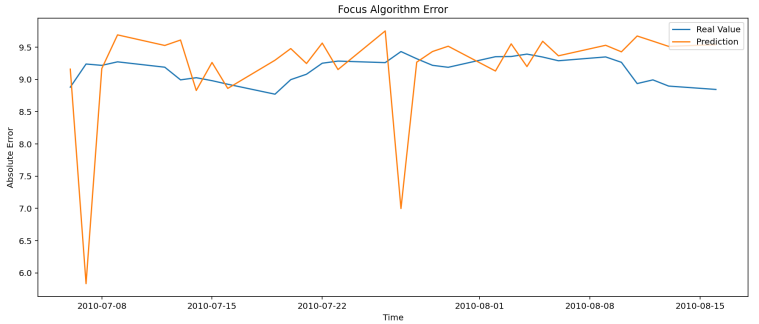


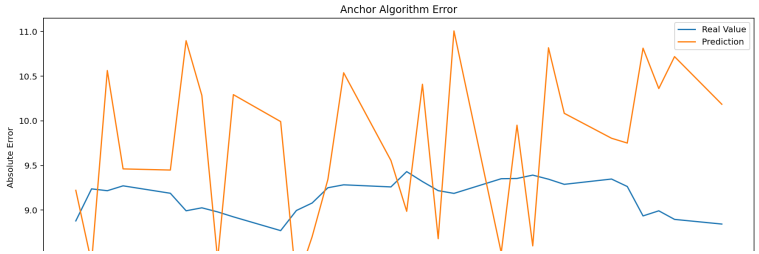
In [30]:
```
plot_performance(ui2)
```

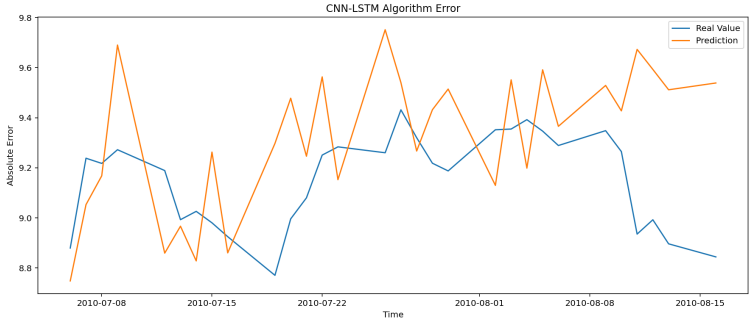


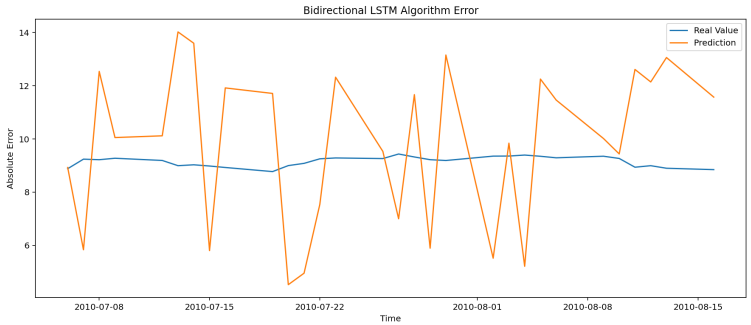


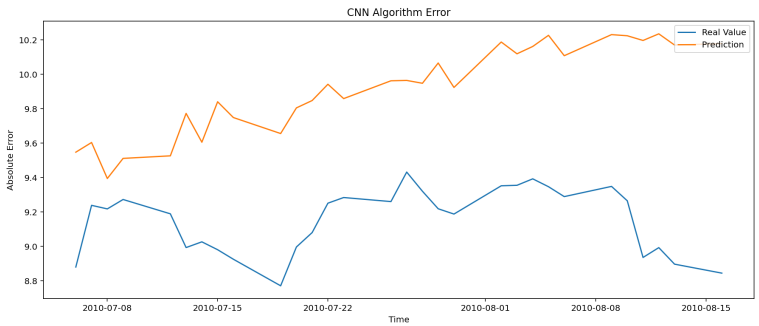


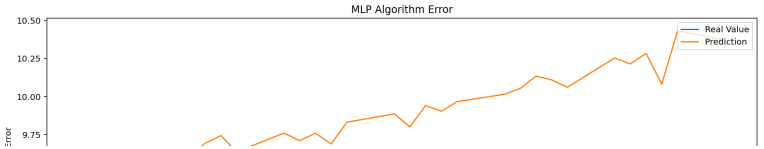



Anchor Algorithm Error
Real Value
Prediction
Absolute Error

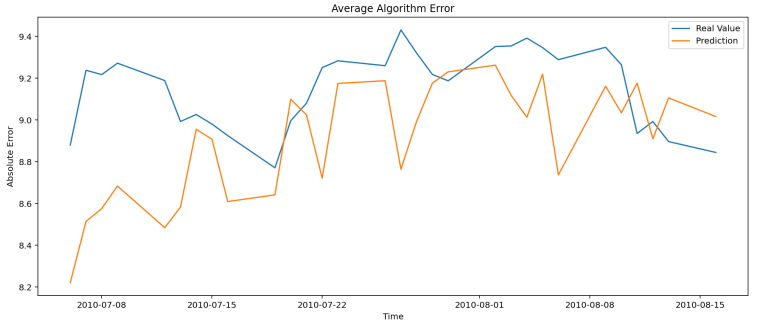

CNN-LSTM Algorithm Error
Real Value
Prediction
Absolute Error
2010-07-08
2010-07-15
2010-07-22
2010-08-01
2010-08-08
2010-08-15
Time

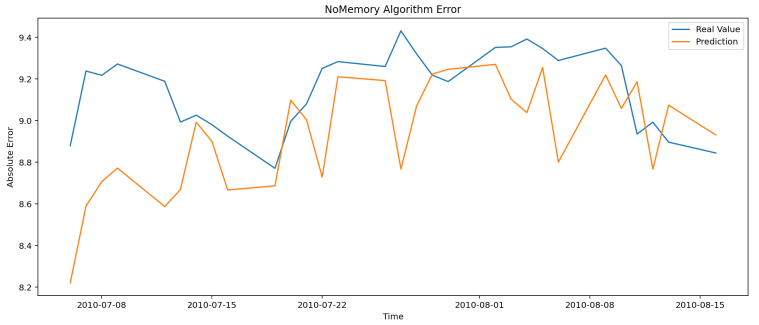

Bidirectional LSTM Algorithm Error
Real Value
Prediction
Absolute Error
2010-07-08
2010-07-15
2010-07-22
2010-08-01
2010-08-08
2010-08-15
Time

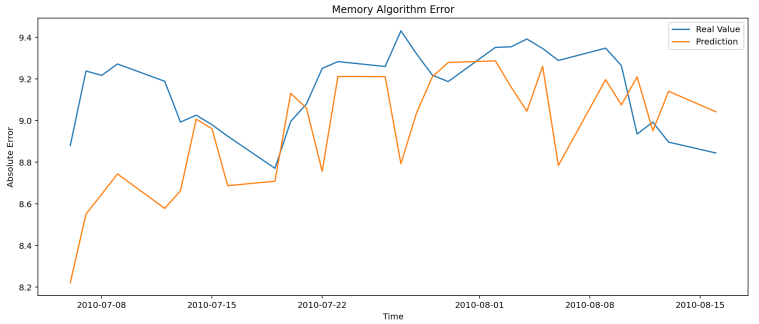

CNN Algorithm Error
Real Value
Prediction
Absolute Error
2010-07-08
2010-07-15
2010-07-22
2010-08-01
2010-08-08
2010-08-15
Time

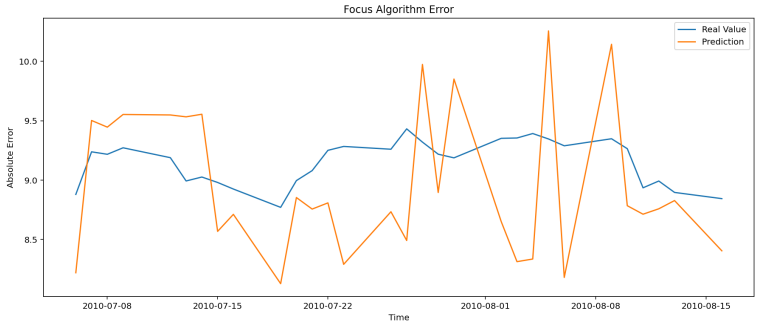

MLP Algorithm Error
Real Value
Prediction

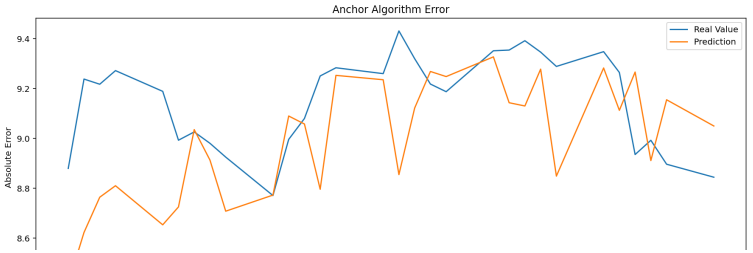
In [31]:
```
plot_performance(ui3)
```

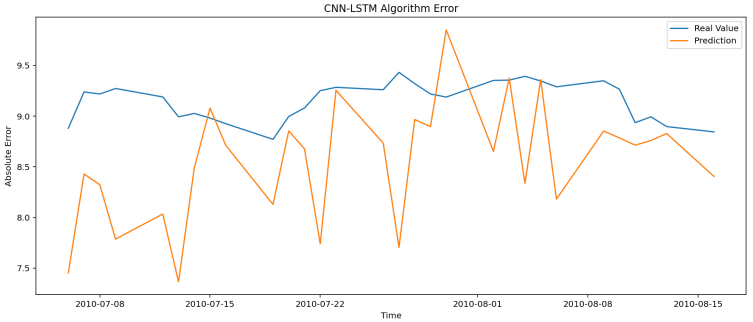


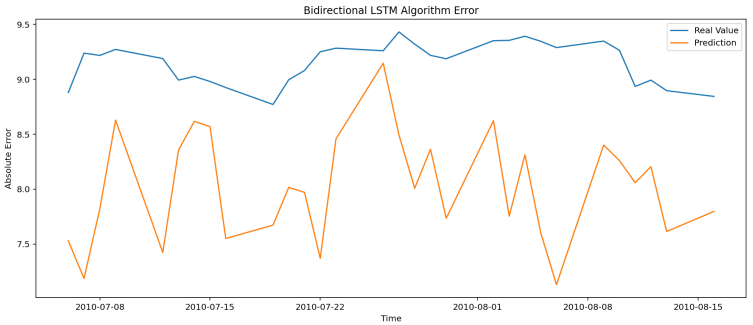


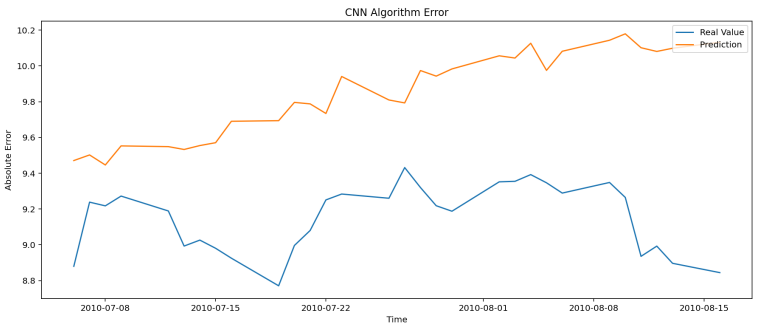


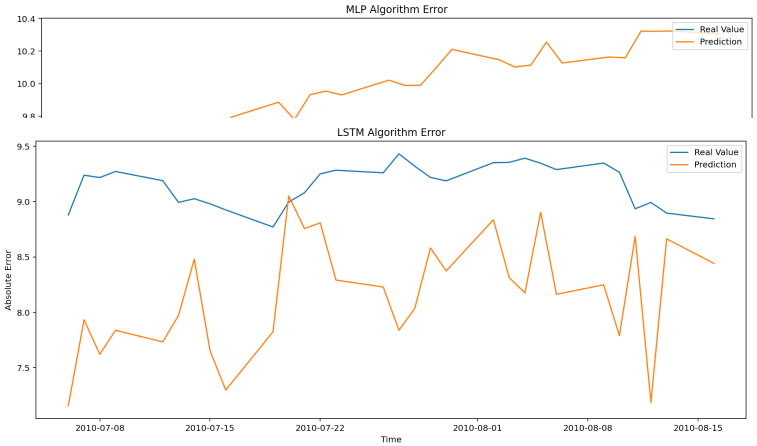











MLP Algorithm Error
Real Value
Prediction
10.4
10.2
10.0
LSTM Algorithm Error
Real Value
Prediction
Absolute Error
9.5
9.0
8.5
8.0
7.5
2010-07-08
2010-07-15
2010-07-22
2010-08-01
2010-08-08
2010-08-15
Time