# 6h Forecast

## Model 1

model = Sequential()  
model.add(LSTM(50, return\_sequences=True, input\_shape=sample\_input\_shape))  
model.add(Dropout(0.2))  
model.add(LSTM(50))  
model.add(Dropout(0.2))  
model.add(Dense(forecast\_horizon))

### Epoc 250 and LR 0.0005

A graph of a model

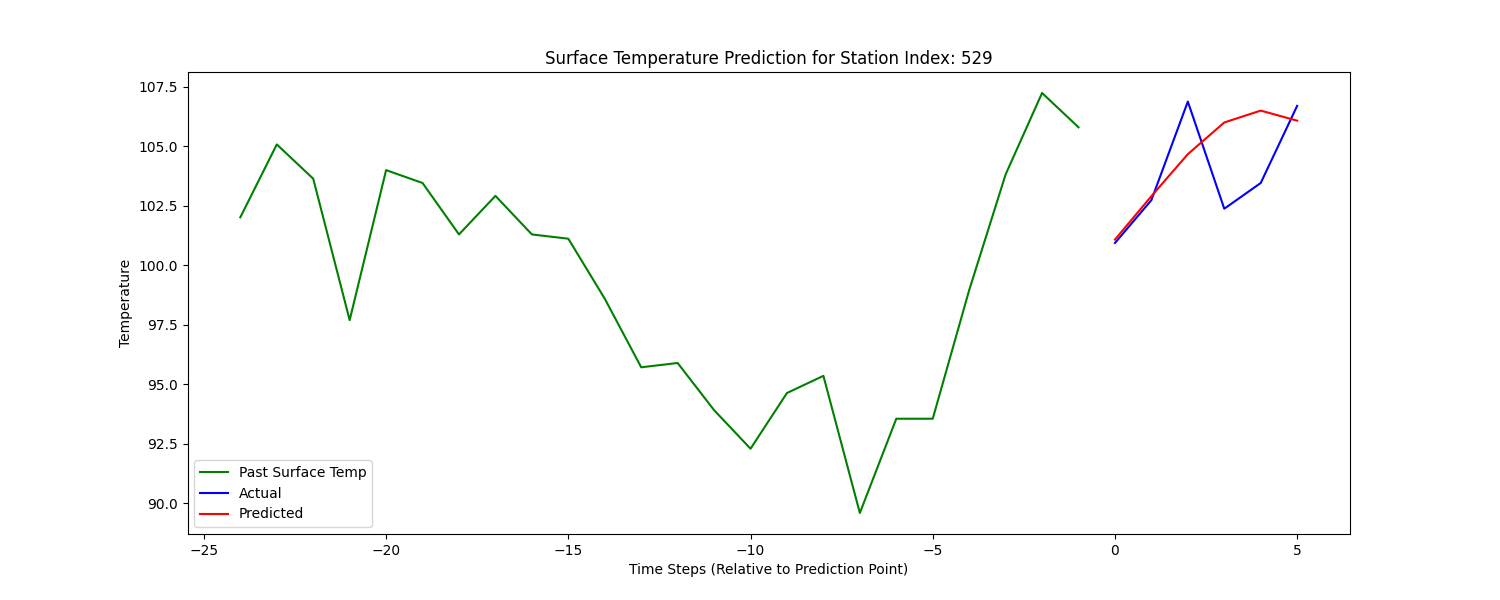
Description automatically generated with medium confidence

A green line graph with white text

Description automatically generated A green line graph with white text

Description automatically generated

A green line graph with white text

Description automatically generated

A green line graph with white text

Description automatically generated A green line graph with white text

Description automatically generated

## Model 2

model = Sequential()  
model.add(Bidirectional(LSTM(100, return\_sequences=True), input\_shape=sample\_input\_shape))  
model.add(Dropout(0.3))  
model.add(BatchNormalization())  
model.add(Bidirectional(LSTM(100, return\_sequences=True)))  
model.add(Dropout(0.3))  
model.add(BatchNormalization())  
model.add(Bidirectional(LSTM(100)))  
model.add(Dropout(0.3))  
model.add(BatchNormalization())  
model.add(Dense(50, activation='relu'))  
model.add(Dropout(0.3))  
model.add(Dense(forecast\_horizon)) # Assuming your output size is 12  
optimizer = tf.keras.optimizers.Adam(learning\_rate=LR)  
model.compile(optimizer=optimizer, loss='mean\_squared\_error', metrics=['mse', 'mae'])  
  
# Model Training  
history = model.fit(X\_train, Y\_train, epochs=Epoch, batch\_size=32, validation\_data=(X\_val, Y\_val), verbose=1)  
  
# After training the model  
model.save('RNNLSTMV1HourForecast6.h5')

### Epoc 250 and LR 0.0005

A graph with numbers and lines

Description automatically generated

A green line graph with white text

Description automatically generatedA green line graph with white text

Description automatically generated

A green line graph with white text

Description automatically generatedA green line graph with white text

Description automatically generated

A green line graph with white text

Description automatically generatedA green line graph with white text

Description automatically generated

# 12h Forecast

## Model 1

model = Sequential()  
model.add(LSTM(50, return\_sequences=True, input\_shape=sample\_input\_shape))  
model.add(Dropout(0.2))  
model.add(LSTM(50))  
model.add(Dropout(0.2))  
model.add(Dense(forecast\_horizon))

### Epoc 250 and LR 0.0005

A graph of a graph

Description automatically generated with medium confidence

A graph with different colored lines

Description automatically generatedA graph with lines and numbers

Description automatically generated with medium confidence

A graph showing a line

Description automatically generated with medium confidenceA graph with green and red lines

Description automatically generated

A graph with green line and red line

Description automatically generatedA graph with green and red lines

Description automatically generated

## Model 2

model = Sequential()  
model.add(Bidirectional(LSTM(100, return\_sequences=True), input\_shape=sample\_input\_shape))  
model.add(Dropout(0.3))  
model.add(BatchNormalization())  
model.add(Bidirectional(LSTM(100, return\_sequences=True)))  
model.add(Dropout(0.3))  
model.add(BatchNormalization())  
model.add(Bidirectional(LSTM(100)))  
model.add(Dropout(0.3))  
model.add(BatchNormalization())  
model.add(Dense(50, activation='relu'))  
model.add(Dropout(0.3))  
model.add(Dense(forecast\_horizon)) # Assuming your output size is 12  
optimizer = tf.keras.optimizers.Adam(learning\_rate=LR)  
model.compile(optimizer=optimizer, loss='mean\_squared\_error', metrics=['mse', 'mae'])  
  
# Model Training  
history = model.fit(X\_train, Y\_train, epochs=Epoch, batch\_size=32, validation\_data=(X\_val, Y\_val), verbose=1)  
  
# After training the model  
model.save('RNNLSTMV1HourForecast6.h5')

### Epoc 250 and LR 0.0005

A graph of a graph

Description automatically generated with medium confidence

A graph with different colored lines

Description automatically generated A graph with lines and points

Description automatically generated with medium confidence

A graph showing a line

Description automatically generated with medium confidence A graph with green lines and red lines

Description automatically generated

A graph with a line and a line

Description automatically generated with medium confidence A graph with different colored lines

Description automatically generated

# 24h Forecast

## Model 1

model = Sequential()  
model.add(LSTM(50, return\_sequences=True, input\_shape=sample\_input\_shape))  
model.add(Dropout(0.2))  
model.add(LSTM(50))  
model.add(Dropout(0.2))  
model.add(Dense(forecast\_horizon))

### Epoc 250 and LR 0.0005

A graph of a graph

Description automatically generated with medium confidence

A graph showing different colored lines

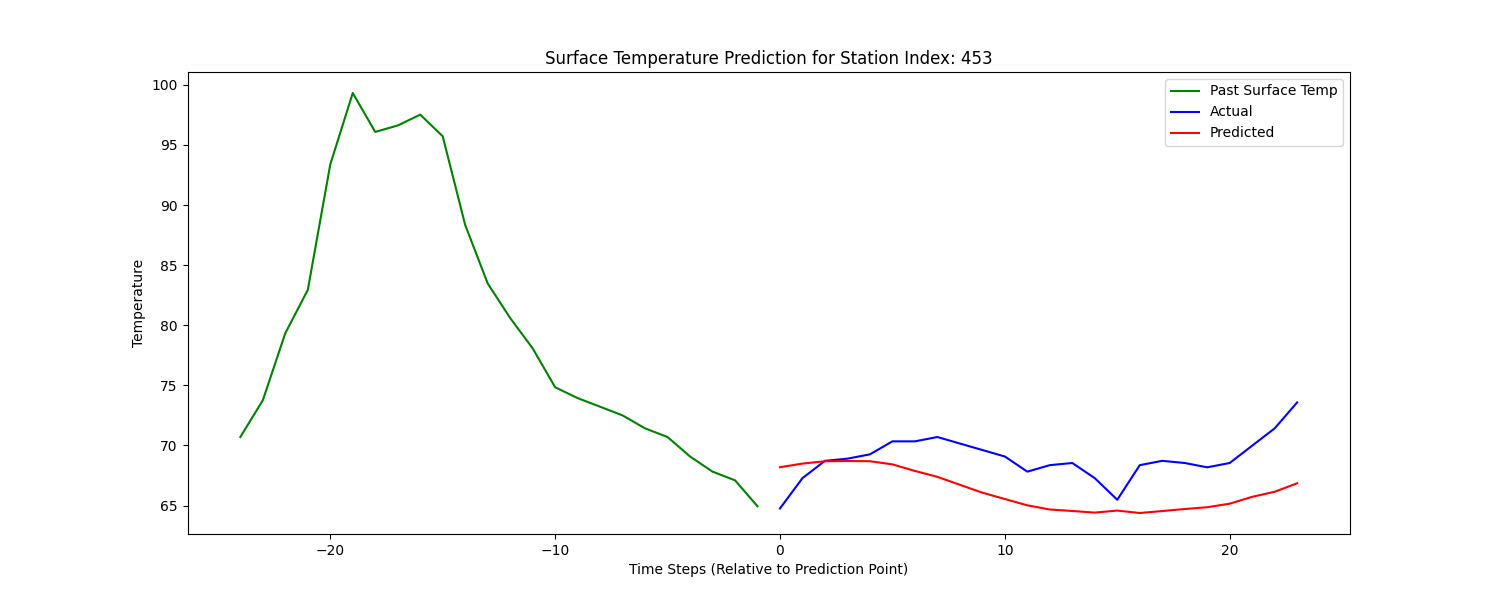
Description automatically generated A graph of a graph

Description automatically generated with medium confidence

A graph with lines and a line in the middle

Description automatically generated with medium confidence A graph showing the value of a stock market

Description automatically generated with medium confidence

 A graph of a graph

Description automatically generated with medium confidence

## Model 2

model = Sequential()  
model.add(Bidirectional(LSTM(100, return\_sequences=True), input\_shape=sample\_input\_shape))  
model.add(Dropout(0.3))  
model.add(BatchNormalization())  
model.add(Bidirectional(LSTM(100, return\_sequences=True)))  
model.add(Dropout(0.3))  
model.add(BatchNormalization())  
model.add(Bidirectional(LSTM(100)))  
model.add(Dropout(0.3))  
model.add(BatchNormalization())  
model.add(Dense(50, activation='relu'))  
model.add(Dropout(0.3))  
model.add(Dense(forecast\_horizon)) # Assuming your output size is 12  
optimizer = tf.keras.optimizers.Adam(learning\_rate=LR)  
model.compile(optimizer=optimizer, loss='mean\_squared\_error', metrics=['mse', 'mae'])  
  
# Model Training  
history = model.fit(X\_train, Y\_train, epochs=Epoch, batch\_size=32, validation\_data=(X\_val, Y\_val), verbose=1)  
  
# After training the model  
model.save('RNNLSTMV1HourForecast6.h5')

### Epoc 250 and LR 0.0005

A graph with numbers and lines

Description automatically generated

A graph showing a line graph

Description automatically generated with medium confidence A graph of a graph

Description automatically generated with medium confidence  
A graph showing a line of a graph

Description automatically generated with medium confidence A graph showing the growth of the stock market

Description automatically generated with medium confidence

A graph with lines and numbers

Description automatically generated with medium confidence