Experiment Results

Results for New York

In [6]:
```
import warnings
warnings.filterwarnings('ignore')

%run helper_functions.py
%matplotlib inline
```

Lets review the top 10 results, ordered by Mean Square Error (ascending)

In [7]:
```
city='New York'
results_top_10 = get_results(city=city, top_hm_results=10)
results_top_10
```

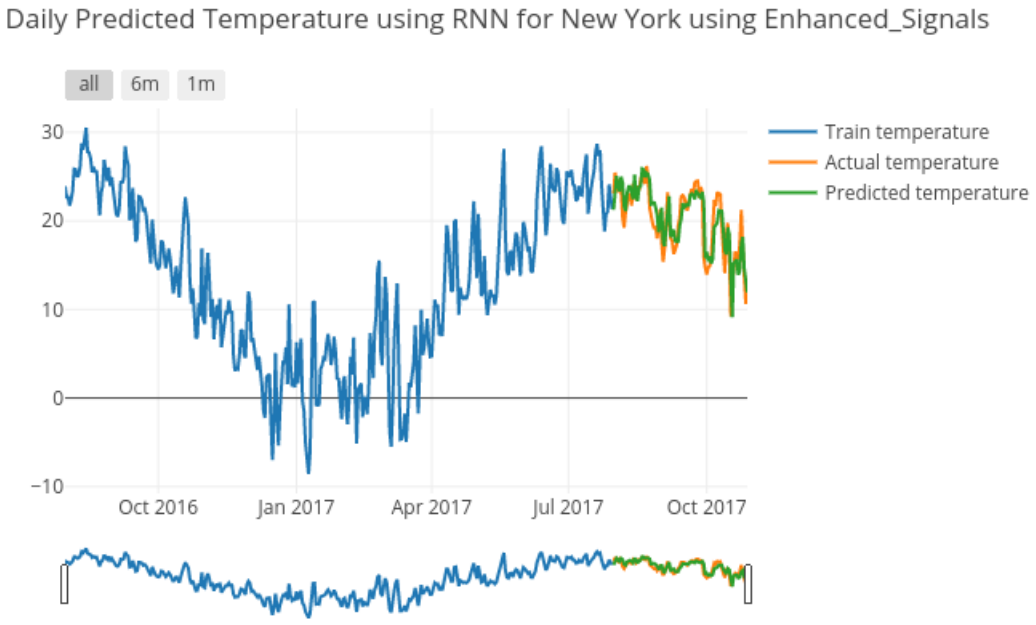
Out[7]:

	RUN_ID	DATETIME	MODEL_NAME	CITY	FEATURE_TYPE	HOST_MACHINE	MODEL_PARAMETERS	MODEL_RESULTS	MEAN_SQUARED_ERROR
1	49	2018-08-14 20:38:45.591679	RNN	New York	Enhanced_Signals	DESKTOP-KN40C32	{'epochs': 50, 'Info': {'feature_set_type': 'E...	{'features': ['temperature_lag1', 'temperature...	4.095801
2	25	2018-08-14 19:55:22.399240	RANDOM FOREST	New York	Inc_Signals	DESKTOP-KN40C32	{'max_depth': 8, 'n_estimators': 50, 'Info': {...	{'features': ['temperature_lag1', 'temperature...	4.216172
3	31	2018-08-14 19:56:16.424827	RANDOM FOREST	New York	Enhanced_Signals	DESKTOP-KN40C32	{'max_depth': 8, 'n_estimators': 50, 'Info': {...	{'features': ['temperature_lag1', 'temperature...	4.216172
4	19	2018-08-14 19:53:51.786200	RANDOM FOREST	New York	Basic	DESKTOP-KN40C32	{'max_depth': 8, 'n_estimators': 50, 'Info': {...	{'features': ['temperature_lag1', 'temperature...	4.369905
5	43	2018-08-14 20:35:48.643558	RNN	New York	Inc_Signals	DESKTOP-KN40C32	{'epochs': 50, 'Info': {'feature_set_type': 'I...	{'features': ['temperature_lag1', 'temperature...	4.700272
6	37	2018-08-14 20:33:25.810145	RNN	New York	Basic	DESKTOP-KN40C32	{'epochs': 50, 'Info': {'feature_set_type': 'B...	{'features': ['temperature_lag1', 'temperature...	4.839433
7	1	2018-08-14 19:41:45.275297	DECISION TREE	New York	Basic	DESKTOP-KN40C32	{'max_depth': 8, 'Info': {'feature_set_type': ...	{'features': ['temperature_lag1', 'temperature...	5.799872
8	7	2018-08-14 19:43:04.972617	DECISION TREE	New York	Inc_Signals	DESKTOP-KN40C32	{'max_depth': 8, 'Info': {'feature_set_type': ...	{'features': ['temperature_lag1', 'temperature...	6.518000
9	13	2018-08-14 19:44:00.243495	DECISION TREE	New York	Enhanced_Signals	DESKTOP-KN40C32	{'max_depth': 8, 'Info': {'feature_set_type': ...	{'features': ['temperature_lag1', 'temperature...	6.767755

RNN has done much better than Random forest so in this instance it is the winner, lets take a look at that chart.

In [8]:
```
display_results(results_top_10.head(1), chart_type='predict')
```

Experiment #49 - run by DESKTOP-KN40C32 on 2018-08-14 20:38:45.591679



We can see the forecast follows the trend and it does a good job following the peaks and troughs of the actual temperature patterns.

Lets examine the top 10 features across the model runs for this location in a pivot table form.

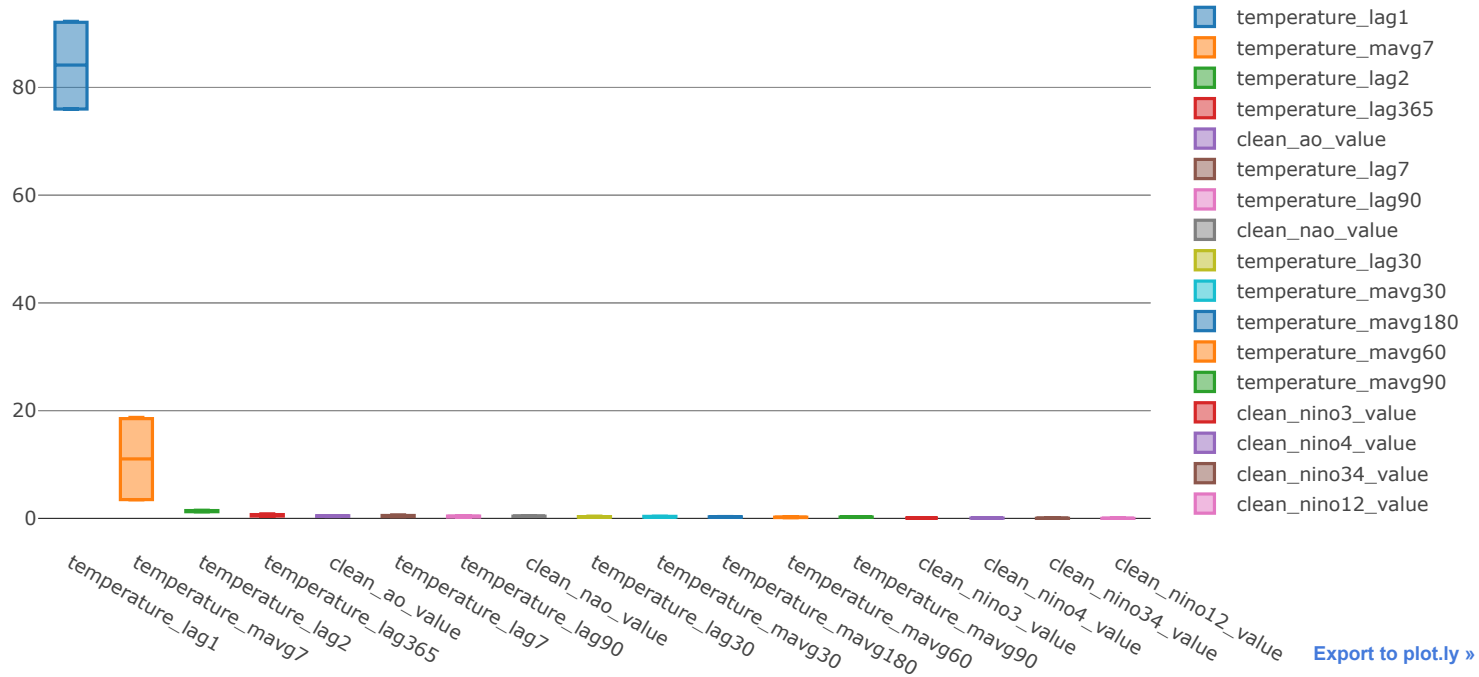
In [13]:
```
features_df = get_feature_importances(results_top_10)
features_df.pivot_table(index=['FEATURE'], columns=['MODEL_NAME','FEATURE_TYPE'], values="IMPORTANCE")
```

Out[13]:

MODEL_NAME	DECISION TREE			RANDOM FOREST		
FEATURE_TYPE	Basic	Enhanced_Signals	Inc_Signals	Basic	Enhanced_Signals	Inc_Signals
FEATURE						
clean_ao_value	NaN	0.342321	0.405794	NaN	0.495784	0.495784
clean_nao_value	NaN	0.413380	0.409221	NaN	0.362524	0.362524
clean_nino12_value	NaN	0.004266	0.000000	NaN	NaN	NaN
clean_nino34_value	NaN	0.016905	0.004148	NaN	NaN	NaN
clean_nino3_value	NaN	0.066890	0.067450	NaN	NaN	NaN
clean_nino4_value	NaN	0.039745	0.062396	NaN	NaN	NaN
temperature_lag1	92.275579	92.092587	92.086427	76.219275	75.992544	75.992544
temperature_lag2	1.509527	1.452181	1.434437	1.444294	1.242860	1.242860
temperature_lag30	0.222598	0.209085	0.215933	0.418034	0.329091	0.329091
temperature_lag365	0.592770	0.433306	0.411147	0.840668	0.707942	0.707942
temperature_lag7	0.540986	0.326928	0.326868	0.634435	0.483584	0.483584
temperature_lag90	0.315943	0.247336	0.219708	0.507118	0.429263	0.429263
temperature_mavg180	0.252067	0.197706	0.240038	0.345253	NaN	NaN
temperature_mavg30	0.376335	0.234558	0.236628	0.385914	0.316122	0.316122
temperature_mavg60	0.179414	0.206461	0.135184	0.240559	NaN	NaN
temperature_mavg7	3.558182	3.466374	3.489681	18.744591	18.520411	18.520411
temperature_mavg90	0.176598	0.249971	0.254940	NaN	NaN	NaN

Lets look a boxplot of this data to see how the distributions look across our experiments for this location.

In [14]:
```
traces = create_boxplot_traces_for_features(features_df)
iplot(traces)
```



Lets review the means for the features

In [15]:
```
features_df = get_feature_importances(results_top_10)
pivot_df = features_df.pivot_table(index=['FEATURE'], columns=[], values="IMPORTANCE", aggfunc= [np.mean])
pivot_df
```

Out[15]:

FEATURE	mean IMPORTANCE
clean_ao_value	0.434921
clean_nao_value	0.386912
clean_nino12_value	0.002133
clean_nino34_value	0.010526
clean_nino3_value	0.067170
clean_nino4_value	0.051071
temperature_lag1	84.109826
temperature_lag2	1.387693
temperature_lag30	0.287305
temperature_lag365	0.615629
temperature_lag7	0.466064
temperature_lag90	0.358105
temperature_mavg180	0.258766
temperature_mavg30	0.310946
temperature_mavg60	0.190405
temperature_mavg7	11.049942
temperature_mavg90	0.227170

Now lets review the top 5 only

In [16]:
```
pivot_df.columns = pivot_df.columns.get_level_values(0)
pivot_df.sort_values(['mean'], ascending=False).head(5)
```

Out[16]:

FEATURE	mean
temperature_lag1	84.109826
temperature_mavg7	11.049942
temperature_lag2	1.387693
temperature_lag365	0.615629
temperature_lag7	0.466064

This is again totally different than all the prior cases as the prior day temperature is the significant factor in the prediction of the temperature.

In [ ]:
```
```