Experiment Results

Results for Dallas

In [8]:
```
import warnings
warnings.filterwarnings('ignore')

%run helper_functions.py
%matplotlib inline
```

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

In [2]:
```
city='Dallas'
results_top_10 = get_results(city=city, top_hm_results=10)
results_top_10
```

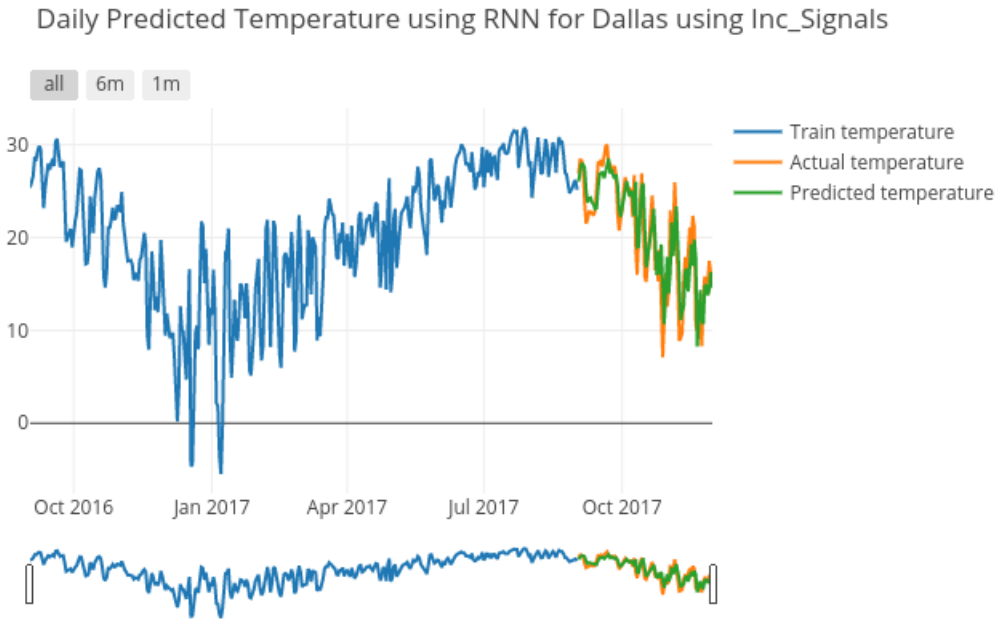
Out[2]:

	RUN_ID	DATETIME	MODEL_NAME	CITY	FEATURE_TYPE	HOST_MACHINE	MODEL_PARAMETERS	MODEL_RESULTS	MEAN_SQUARED_ERROR
1	46	2018-08-14 20:37:30.002193	RNN	Dallas	Inc_Signals	DESKTOP-KN40C32	{'epochs': 50, 'Info': {'feature_set_type': 'I...	{'features': ['temperature_lag1', 'temperature...	8.336575
2	40	2018-08-14 20:34:55.307517	RNN	Dallas	Basic	DESKTOP-KN40C32	{'epochs': 50, 'Info': {'feature_set_type': 'B...	{'features': ['temperature_lag1', 'temperature...	8.563057
3	52	2018-08-14 20:40:06.720627	RNN	Dallas	Enhanced_Signals	DESKTOP-KN40C32	{'epochs': 50, 'Info': {'feature_set_type': 'E...	{'features': ['temperature_lag1', 'temperature...	8.632428
4	22	2018-08-14 19:54:57.660172	RANDOM FOREST	Dallas	Basic	DESKTOP-KN40C32	{'max_depth': 8, 'n_estimators': 50, 'Info': {...	{'features': ['temperature_lag1', 'temperature...	8.983822
5	28	2018-08-14 19:55:47.002009	RANDOM FOREST	Dallas	Inc_Signals	DESKTOP-KN40C32	{'max_depth': 8, 'n_estimators': 50, 'Info': {...	{'features': ['temperature_lag1', 'temperature...	9.260334
6	34	2018-08-14 19:56:37.269205	RANDOM FOREST	Dallas	Enhanced_Signals	DESKTOP-KN40C32	{'max_depth': 8, 'n_estimators': 50, 'Info': {...	{'features': ['temperature_lag1', 'temperature...	9.260334
7	4	2018-08-14 19:42:30.715058	DECISION TREE	Dallas	Basic	DESKTOP-KN40C32	{'max_depth': 8, 'Info': {'feature_set_type': ...	{'features': ['temperature_lag1', 'temperature...	12.523335
8	10	2018-08-14 19:43:32.825549	DECISION TREE	Dallas	Inc_Signals	DESKTOP-KN40C32	{'max_depth': 8, 'Info': {'feature_set_type': ...	{'features': ['temperature_lag1', 'temperature...	14.223161
9	16	2018-08-14 19:44:20.754761	DECISION TREE	Dallas	Enhanced_Signals	DESKTOP-KN40C32	{'max_depth': 8, 'Info': {'feature_set_type': ...	{'features': ['temperature_lag1', 'temperature...	15.432839

RNN has done better than Random forest but the results are close so there is no clear winner, lets take a look at that chart.

In [3]:
```
display_results(results_top_10.head(1), chart_type='predict')
```

Experiment #46 - run by DESKTOP-KN40C32 on 2018-08-14 20:37:30.002193



We can see the forecast follows the trend and it does a better job but like the prior example of Miami it has some difficulty following the peaks and troughs of the actual temperature pattern experiences.

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

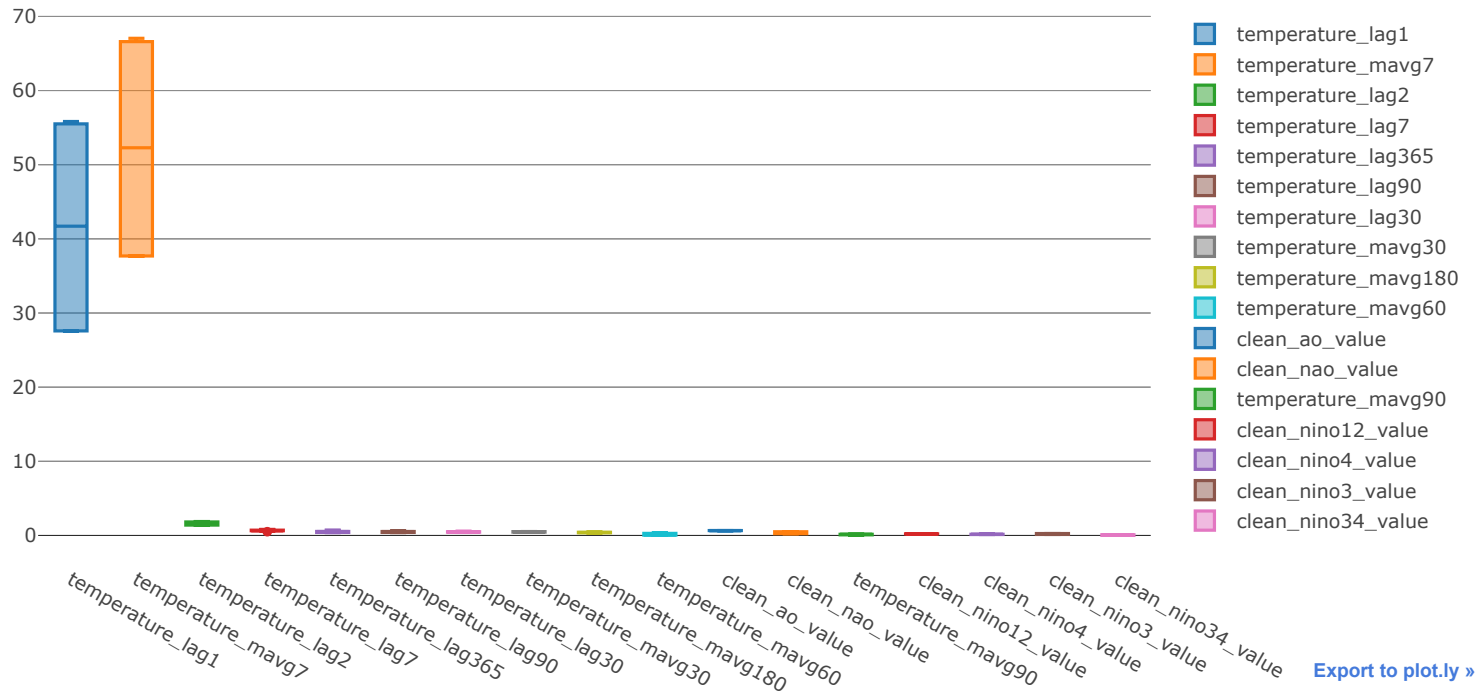
In [4]:
```
features_df = get_feature_importances(results_top_10)
features_df.pivot_table(index=['FEATURE'], columns=['MODEL_NAME','FEATURE_TYPE'], values="IMPORTANCE")
```

Out[4]:

MODEL_NAME	DECISION TREE			RANDOM FOREST		
FEATURE_TYPE	Basic	Enhanced_Signals	Inc_Signals	Basic	Enhanced_Signals	Inc_Signals
FEATURE						
clean_ao_value	NaN	0.651503	0.656238	NaN	0.658562	0.658562
clean_nao_value	NaN	0.226906	0.249332	NaN	0.499492	0.499492
clean_nino12_value	NaN	0.215984	0.212701	NaN	NaN	NaN
clean_nino34_value	NaN	0.006323	0.095549	NaN	NaN	NaN
clean_nino3_value	NaN	0.257658	0.133802	NaN	NaN	NaN
clean_nino4_value	NaN	0.125707	0.157822	NaN	NaN	NaN
temperature_lag1	27.909283	27.582810	27.584224	55.826530	55.508373	55.508373
temperature_lag2	1.816951	1.397116	1.382537	1.875478	1.629844	1.629844
temperature_lag30	0.497862	0.437565	0.403289	0.583795	0.477568	0.477568
temperature_lag365	0.516834	0.400962	0.375246	0.728918	0.538135	0.538135
temperature_lag7	0.503527	0.638941	0.711708	0.814543	0.634003	0.634003
temperature_lag90	0.530253	0.401933	0.383665	0.635825	0.522790	0.522790
temperature_mavg180	0.453587	0.378973	0.344204	0.405867	NaN	NaN
temperature_mavg30	0.486033	0.516936	0.514952	0.526882	0.447634	0.447634
temperature_mavg60	0.181417	0.044381	0.043627	0.373975	NaN	NaN
temperature_mavg7	67.048224	66.559364	66.600341	38.009765	37.689387	37.689387
temperature_mavg90	0.056029	0.156939	0.150763	NaN	NaN	NaN

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

In [9]:
```
traces = create_boxplot_traces_for_features(features_df)
iplot(traces)
```



Lets review the means for the features

In [6]:
```
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[6]:

FEATURE	mean IMPORTANCE
clean_ao_value	0.656216
clean_nao_value	0.368805
clean_nino12_value	0.214342
clean_nino34_value	0.050936
clean_nino3_value	0.195730
clean_nino4_value	0.141765
temperature_lag1	41.653265
temperature_lag2	1.621962
temperature_lag30	0.479608
temperature_lag365	0.516371
temperature_lag7	0.656121
temperature_lag90	0.499543
temperature_mavg180	0.395658
temperature_mavg30	0.490012
temperature_mavg60	0.160850
temperature_mavg7	52.266078
temperature_mavg90	0.121244

Now lets review the top 5 only

In [7]:
```
pivot_df.columns = pivot_df.columns.get_level_values(0)
pivot_df.sort_values(['mean'], ascending=False).head(5)
```

Out[7]:

FEATURE	mean
temperature_mavg7	52.266078
temperature_lag1	41.653265
temperature_lag2	1.621962
clean_ao_value	0.656216
temperature_lag7	0.656121

This is different than the prior cases as we see the moving average taking the lead in the predictions.

In [ ]: