01-Prophet-Forecast-Evaluations

October 19, 2022

For more information, visit us at www.pieriandata.com

1 Forecast Evaluations

1.0.1 Classic Train/Test Split

[1]:
```
import pandas as pd
from fbprophet import Prophet
%matplotlib inline
```

[183]:
```
df = pd.read_csv('../Data/Miles_Traveled.csv')
```

[184]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 588 entries, 0 to 587
Data columns (total 2 columns):
DATE                 588 non-null object
TRFVOLUSM227NFWA     588 non-null float64
dtypes: float64(1), object(1)
memory usage: 9.3+ KB
```

[185]:
```
df.head()
```

[185]:

	DATE	TRFVOLUSM227NFWA
0	1970-01-01	80173.0
1	1970-02-01	77442.0
2	1970-03-01	90223.0
3	1970-04-01	89956.0
4	1970-05-01	97972.0

[186]:
```
df.columns = ['ds','y']
```

[187]:
```
# YOU CAN SAFELY IGNORE THIS COPY WARNING
# CHECK THE DOCS LINK FOR MORE INFO ON THIS COPY WARNING
df['ds'] = pd.to_datetime(df['ds'])
```

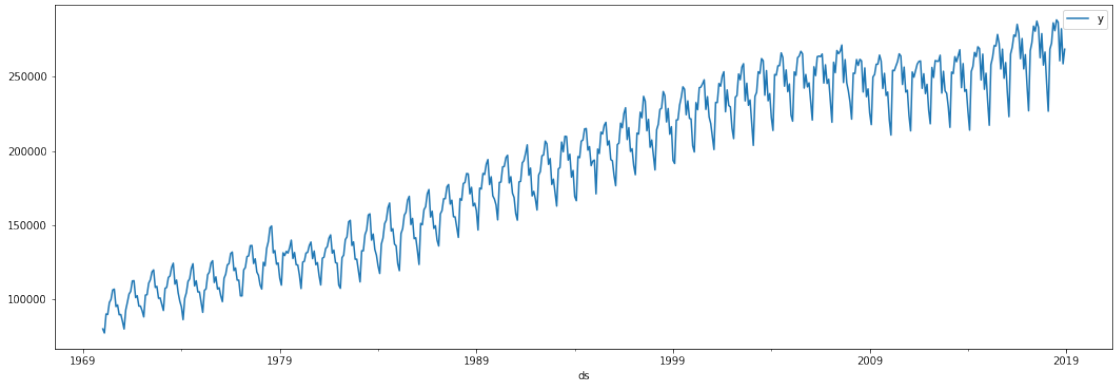
[188]:
```
df.head()
```

[188]:

	ds	y
0	1970-01-01	80173.0
1	1970-02-01	77442.0
2	1970-03-01	90223.0
3	1970-04-01	89956.0
4	1970-05-01	97972.0

[189]:
```
df.plot(x='ds',y='y',figsize=(18,6))
```

[189]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1a1bcf1860>
```



[190]:
```
len(df)
```

[190]:
```
588
```

[191]:
```
len(df) - 12
```

[191]:
```
576
```

[192]:
```
train = df.iloc[:576]
test = df.iloc[576:]
```

[193]:
```
m = Prophet()
m.fit(train)
future = m.make_future_dataframe(periods=12,freq='MS')
forecast = m.predict(future)
```

```
INFO:fbprophet:Disabling weekly seasonality. Run prophet with
weekly_seasonality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with
daily_seasonality=True to override this.
/anaconda3/lib/python3.6/site-packages/pystan/misc.py:399: FutureWarning:
Conversion of the second argument of issubdtype from `float` to `np.floating` is
deprecated. In future, it will be treated as `np.float64 ==
np.dtype(float).type`.
  elif np.issubdtype(np.asarray(v).dtype, float):
```

[194]:
```
forecast.tail()
```

[194]:

	ds	trend	yhat_lower	yhat_upper	trend_lower	trend_upper	additive_terms	additive_terms_lower	additive_terms_upper	yearly	yearly_lower	yearly_upper	multiplicative_terms	multiplicative_terms_lower	multiplicative_terms_upper	yhat
583	2018-08-01	263348.877048	274069.034629	285530.763538	263272.182929	263408.698329	16448.763644	16448.763644	16448.763644	16448.763644	16448.763644	16448.763644	0.0	0.0	0.0	279797.640692
584	2018-09-01	263489.606772	255853.724067	267198.537533	263369.019000	263577.835001	-1669.492450	-1669.492450	-1669.492450	-1669.492450	-1669.492450	-1669.492450	0.0	0.0	0.0	261820.114321
585	2018-10-01	263625.796827	263247.354895	274539.143701	263477.532695	263745.882773	5307.799177	5307.799177	5307.799177	5307.799177	5307.799177	5307.799177	0.0	0.0	0.0	268933.596004
586	2018-11-01	263766.526551	250087.267929	261329.148126	263586.328108	263936.075269	-8206.692440	-8206.692440	-8206.692440	-8206.692440	-8206.692440	-8206.692440	0.0	0.0	0.0	255559.834111
587	2018-12-01	263902.716607	251026.012016	262766.066957	263680.350517	264106.895858	-6920.633214	-6920.633214	-6920.633214	-6920.633214	-6920.633214	-6920.633214	0.0	0.0	0.0	256982.083393

[195]:
```
test.tail()
```

[195]:

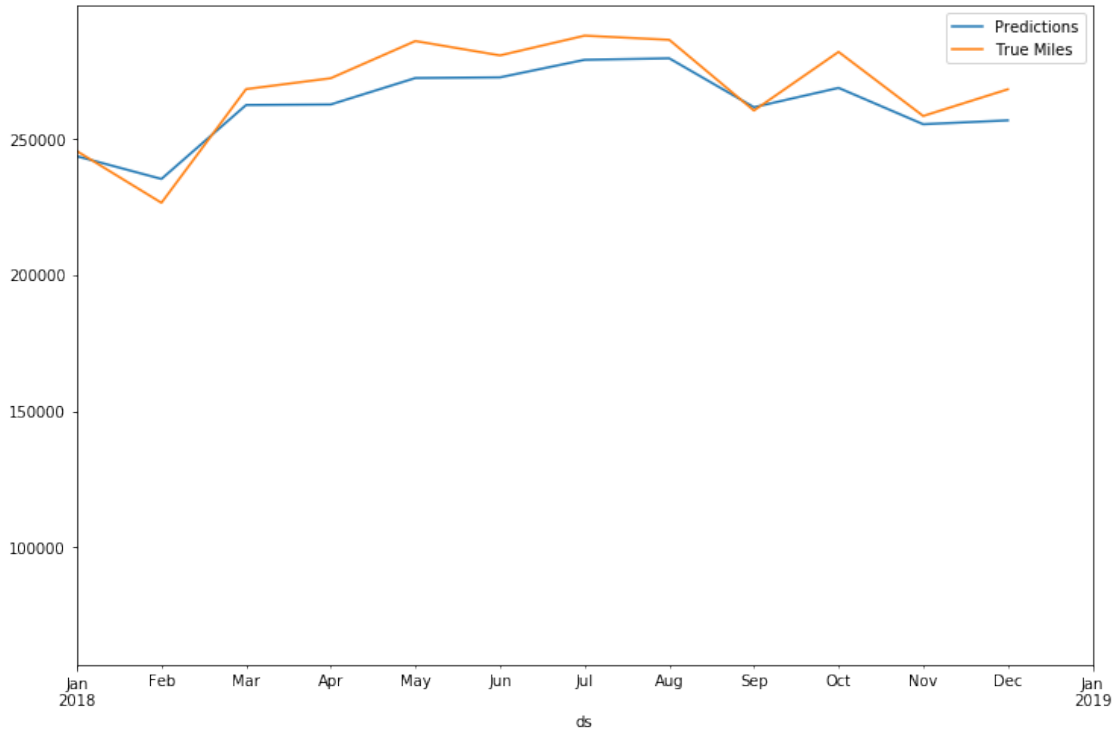
	ds	y
583	2018-08-01	286608.0
584	2018-09-01	260595.0
585	2018-10-01	282174.0
586	2018-11-01	258590.0
587	2018-12-01	268413.0

Info on ax= parameter

[196]:
```
ax = forecast.
↪plot(x='ds',y='yhat',label='Predictions',legend=True,figsize=(12,8))

test.plot(x='ds',y='y',label='True␣
↪Miles',legend=True,ax=ax,xlim=('2018-01-01','2019-01-01'))
```

[196]: <matplotlib.axes._subplots.AxesSubplot at 0x1a1be0d9e8>



[197]:
```
from statsmodels.tools.eval_measures import rmse
```

[198]:
```
predictions = forecast.iloc[-12:]['yhat']
```

[199]:
```
predictions
```

[199]:
```
576    243796.609534
577    235426.627562
578    262627.330849
579    262828.742010
580    272552.156586
581    272803.877580
582    279261.432668
583    279797.640692
584    261820.114321
585    268933.596004
586    255559.834111
587    256982.083393
Name: yhat, dtype: float64
```

[200]:
```
test['y']
```

[200]:
```
576    245695.0
577    226660.0
578    268480.0
579    272475.0
580    286164.0
581    280877.0
582    288145.0
583    286608.0
584    260595.0
585    282174.0
586    258590.0
587    268413.0
Name: y, dtype: float64
```

[201]:
```
rmse(predictions,test['y'])
```

[201]:
```
8661.095901709492
```

[202]:
```
test.mean()
```

[202]:
```
y    268739.666667
dtype: float64
```

2 Prophet Diagnostics

Prophet includes functionality for time series cross validation to measure forecast error using historical data. This is done by selecting cutoff points in the history, and for each of them fitting the

model using data only up to that cutoff point. We can then compare the forecasted values to the actual values.

[204]:
```
from fbprophet.diagnostics import cross_validation,performance_metrics
from fbprophet.plot import plot_cross_validation_metric
```

[205]:
```
len(df)
```

[205]: 588

[206]:
```
len(df)/12
```

[206]: 49.0

The initial period should be long enough to capture all of the components of the model, in particular seasonalities and extra regressors: at least a year for yearly seasonality, at least a week for weekly seasonality, etc.

[207]:
```
# help(pd.Timedelta)
```

[162]:
```
# Initial 5 years training period
initial = 5 * 365
initial = str(initial) + ' days'
# Fold every 5 years
period = 5 * 365
period = str(period) + ' days'
# Forecast 1 year into the future
horizon = 365
horizon = str(horizon) + ' days'
```

[210]:
```
df_cv = cross_validation(m, initial=initial, period=period, horizon = horizon)
```

```
INFO:fbprophet:Making 9 forecasts with cutoffs between 1976-12-11 00:00:00 and
2016-12-01 00:00:00
/anaconda3/lib/python3.6/site-packages/pystan/misc.py:399: FutureWarning:
Conversion of the second argument of issubdtype from `float` to `np.floating` is
deprecated. In future, it will be treated as `np.float64 ==
np.dtype(float).type`.
  elif np.issubdtype(np.asarray(v).dtype, float):
```

[213]:
```
df_cv.head()
```

[213]:

	ds	yhat	yhat_lower	yhat_upper	y	cutoff
0	1977-01-01	108545.172580	107099.748151	110016.346952	102445.0	1976-12-11
1	1977-02-01	103082.245846	101525.992338	104689.450272	102416.0	1976-12-11
2	1977-03-01	118966.635618	117449.707291	120459.804913	119960.0	1976-12-11
3	1977-04-01	120606.270683	119041.821340	122263.557564	121513.0	1976-12-11
4	1977-05-01	127889.386836	126347.199524	129456.699558	128884.0	1976-12-11

[214]:
```
df_cv.tail()
```

[214]:

	ds	yhat	yhat_lower	yhat_upper	y	cutoff
103	2017-08-01	273254.523501	267202.320371	279511.599691	283184.0	2016-12-01
104	2017-09-01	255372.720203	249330.743827	261084.844672	262673.0	2016-12-01
105	2017-10-01	262476.518564	256769.520993	268001.419001	278937.0	2016-12-01
106	2017-11-01	249126.648938	243459.338900	254747.159187	257712.0	2016-12-01
107	2017-12-01	250371.688410	244625.193390	256090.260107	266535.0	2016-12-01

[215]:
```
performance_metrics(df_cv)
```

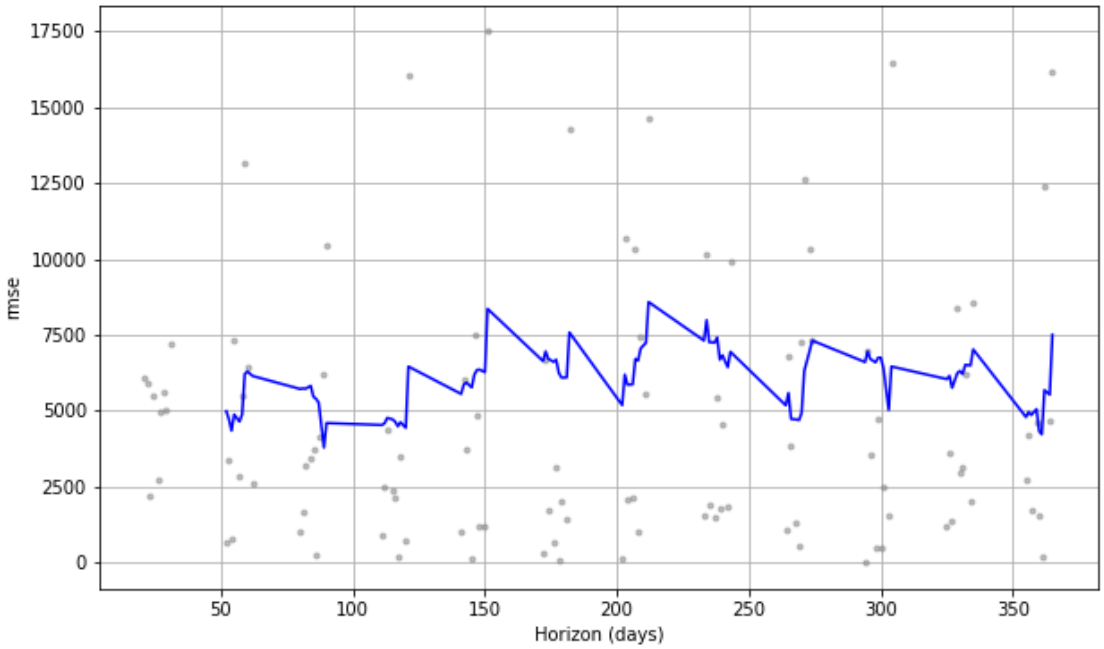
[215]:

	horizon	mse	rmse	mae	mape	coverage
1	52 days	2.484109e+07	4984.083210	4590.014835	0.028137	0.3
13	53 days	2.226842e+07	4718.943023	4318.899977	0.025335	0.3
25	54 days	1.882583e+07	4338.874820	3803.991852	0.020509	0.4
37	55 days	2.374039e+07	4872.411127	4319.472747	0.023528	0.3
49	57 days	2.150039e+07	4636.851520	4050.449106	0.021775	0.4
61	58 days	2.377014e+07	4875.463292	4327.071181	0.022979	0.3
73	59 days	3.857268e+07	6210.691064	5144.794773	0.026666	0.3
85	60 days	3.959640e+07	6292.566974	5229.654576	0.027225	0.3
97	62 days	3.773627e+07	6142.985167	4985.238970	0.026153	0.4
2	80 days	3.264865e+07	5713.899872	4364.416238	0.024037	0.5
14	81 days	3.287214e+07	5733.423310	4461.461888	0.024663	0.5
26	82 days	3.273578e+07	5721.519435	4440.708849	0.023530	0.6
38	84 days	3.384815e+07	5817.916077	4706.128999	0.024828	0.6
50	85 days	2.983983e+07	5462.584849	4343.936561	0.022000	0.7
62	86 days	2.905115e+07	5389.912117	4085.532750	0.020567	0.7
74	87 days	2.775881e+07	5268.662783	3951.324112	0.019528	0.8
86	89 days	1.430251e+07	3781.865304	3254.512729	0.015944	0.8
98	90 days	2.108837e+07	4592.207361	3655.515480	0.016900	0.8
3	111 days	2.050330e+07	4528.056835	3487.869704	0.016508	0.8
15	112 days	2.102938e+07	4585.780229	3638.485039	0.017605	0.8
27	113 days	2.267923e+07	4762.271752	3912.733905	0.019068	0.7
39	115 days	2.221524e+07	4713.304385	3828.720432	0.018308	0.7
51	116 days	2.148911e+07	4635.634353	3697.477130	0.017444	0.7
63	117 days	2.010623e+07	4483.996740	3341.901127	0.015758	0.7
75	118 days	2.133499e+07	4618.981764	3669.814813	0.017050	0.7
87	120 days	1.967055e+07	4435.148827	3327.928592	0.015744	0.7
99	121 days	4.166916e+07	6455.165095	4317.240942	0.019222	0.7
4	141 days	3.081409e+07	5551.044078	3370.089073	0.016079	0.8

```
16  142 days  3.438259e+07  5863.667285  3883.627983  0.019640       0.7
28  143 days  3.514306e+07  5928.158078  4005.862058  0.019933       0.6
..       …            …            …            …         …         …
80  271 days  3.987511e+07  6314.674121  4976.697924  0.023250       0.4
92  273 days  4.844885e+07  6960.520511  5553.268374  0.025891       0.4
104 274 days  5.344181e+07  7310.390781  6099.873093  0.027976       0.3
9   294 days  4.358237e+07  6601.694686  5107.910769  0.024477       0.4
21  295 days  4.833680e+07  6952.467359  5700.946543  0.028662       0.3
33  296 days  4.493632e+07  6703.455933  5371.935619  0.025776       0.4
45  298 days  4.346401e+07  6592.723880  5032.356394  0.023663       0.5
57  299 days  4.552935e+07  6747.544186  5373.830620  0.025112       0.4
69  300 days  4.552068e+07  6746.901166  5365.366312  0.025043       0.4
81  301 days  4.089633e+07  6395.023462  4890.486857  0.022897       0.5
93  303 days  2.522518e+07  5022.467234  3784.087415  0.018380       0.6
105 304 days  4.167387e+07  6455.530562  4398.338718  0.019962       0.6
10  325 days  3.647964e+07  6039.837575  3784.574064  0.018164       0.7
22  326 days  3.777531e+07  6146.162380  4143.544816  0.020989       0.6
34  327 days  3.309616e+07  5752.926051  3581.869386  0.016836       0.7
46  329 days  3.890957e+07  6237.753416  4069.502509  0.019459       0.6
58  330 days  3.977554e+07  6306.784866  4320.509403  0.020690       0.6
70  331 days  3.850361e+07  6205.127551  4158.292100  0.019899       0.7
82  332 days  4.235969e+07  6508.432077  4734.018144  0.022242       0.6
94  334 days  4.215073e+07  6492.359713  4687.867562  0.022133       0.6
106 335 days  4.928129e+07  7020.063687  5391.397587  0.024853       0.5
11  355 days  2.293877e+07  4789.443607  4019.615883  0.021311       0.5
23  356 days  2.458287e+07  4958.111216  4325.166305  0.023648       0.4
35  357 days  2.357850e+07  4855.769282  4135.888952  0.021912       0.5
47  359 days  2.551276e+07  5051.015397  4460.452054  0.023583       0.4
59  360 days  1.868912e+07  4323.091979  3772.440320  0.019581       0.5
71  361 days  1.780565e+07  4219.674693  3495.847307  0.018200       0.5
83  362 days  3.222687e+07  5676.871671  4425.396521  0.022014       0.4
95  364 days  3.053159e+07  5525.539664  4269.861337  0.021438       0.5
107 365 days  5.624352e+07  7499.567688  5682.885529  0.026656       0.4

[99 rows x 6 columns]
```

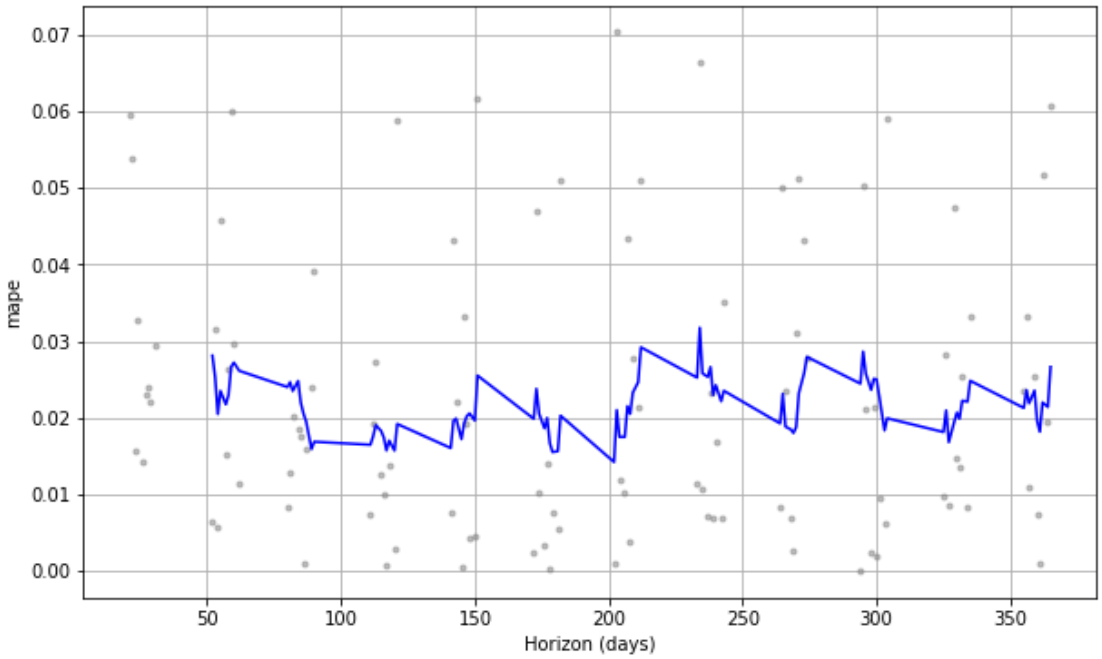
[217]:
```
plot_cross_validation_metric(df_cv, metric='rmse');
```



MAPE is Mean Absolute Percentage Error

https://en.wikipedia.org/wiki/Mean_absolute_percentage_error

[218]:
```
plot_cross_validation_metric(df_cv, metric='mape');
```



3 Great Job!