```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

import xgboost as xgb
from sklearn.metrics import mean_squared_error
color_pal = sns.color_palette()
plt.style.use('fivethirtyeight')
```

```
df = pd.read_csv('/content/AEP_hourly.csv')
df =df.set_index('Datetime')
df.index=pd.to_datetime(df.index)
```

```
df.head()

```

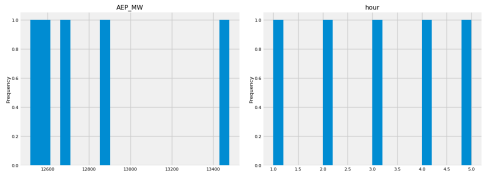
1 to 5 of 5 entries Filter

Datetime	AEP_MW	hour	dayofweek	quarter	month	year	dayofyear
2004-12-31 01:00:00	13478.0	1	4	4	12	2004	366
2004-12-31 02:00:00	12865.0	2	4	4	12	2004	366
2004-12-31 03:00:00	12577.0	3	4	4	12	2004	366
2004-12-31 04:00:00	12517.0	4	4	4	12	2004	366
2004-12-31 05:00:00	12670.0	5	4	4	12	2004	366

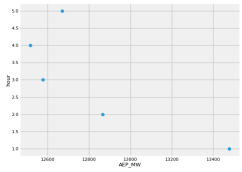
Show 25 per page

Like what you see? Visit the data table notebook to learn more about interactive tables.

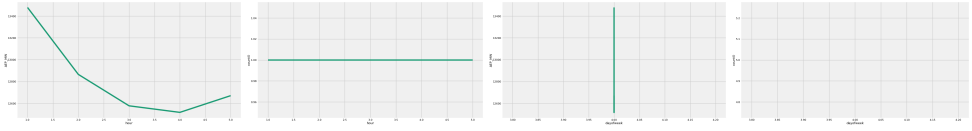
Distributions



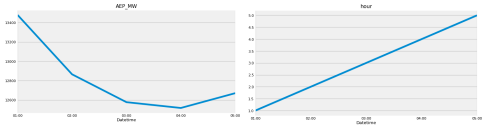
2-d distributions



Time series



Values




```
df.tail()
```

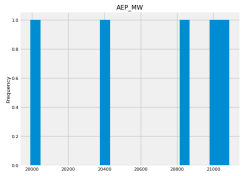
1 to 5 of 5 entries Filter

Datetime	AEP_MW
2018-01-01 20:00:00	21089.0
2018-01-01 21:00:00	20999.0
2018-01-01 22:00:00	20820.0
2018-01-01 23:00:00	20415.0
2018-01-02 00:00:00	19993.0

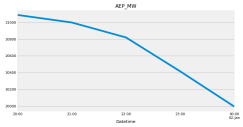
Show 25 per page

Like what you see? Visit the data table notebook to learn more about interactive tables.

Distributions



Values



```
df.describe()
```

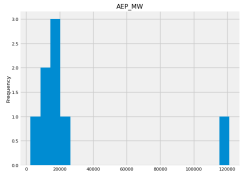
1 to 8 of 8 entries Filter

index	AEP_MW
count	121273.0
mean	15499.513716985644
std	2591.399065407914
min	9581.0
25%	13630.0
50%	15310.0
75%	17200.0
max	25695.0

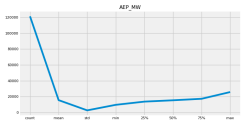
Show 25 per page

Like what you see? Visit the data table notebook to learn more about interactive tables.

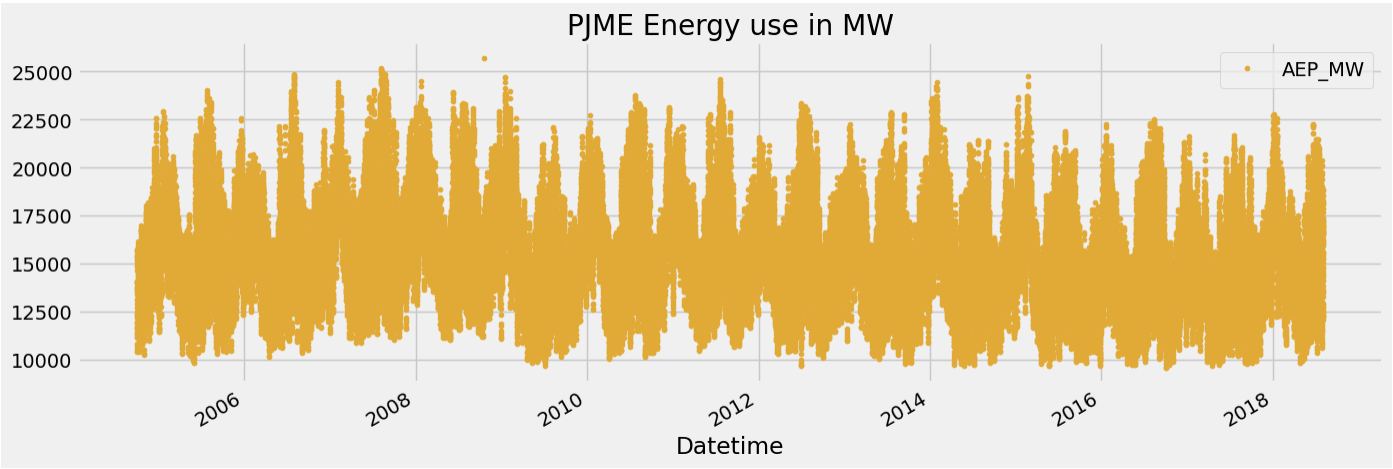
Distributions



Values



```
df.plot(style='.',
        figsize=(15, 5),
        color=color_pal[2],
        title='PJME Energy use in MW')
plt.show()
```

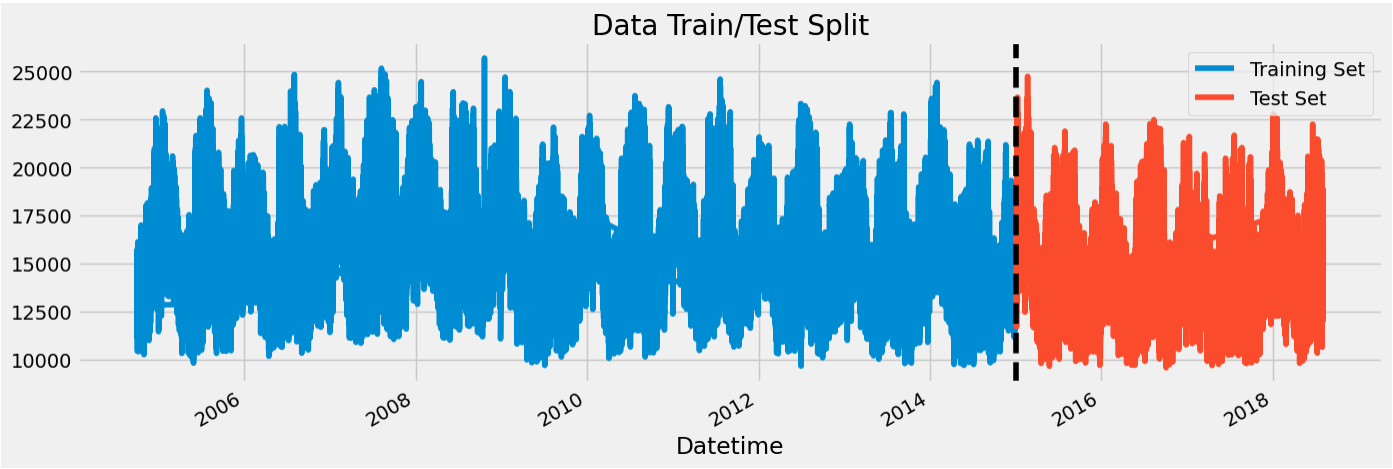



```
Start coding or generate with AI.
```

TRAIN/TEST SPLIT

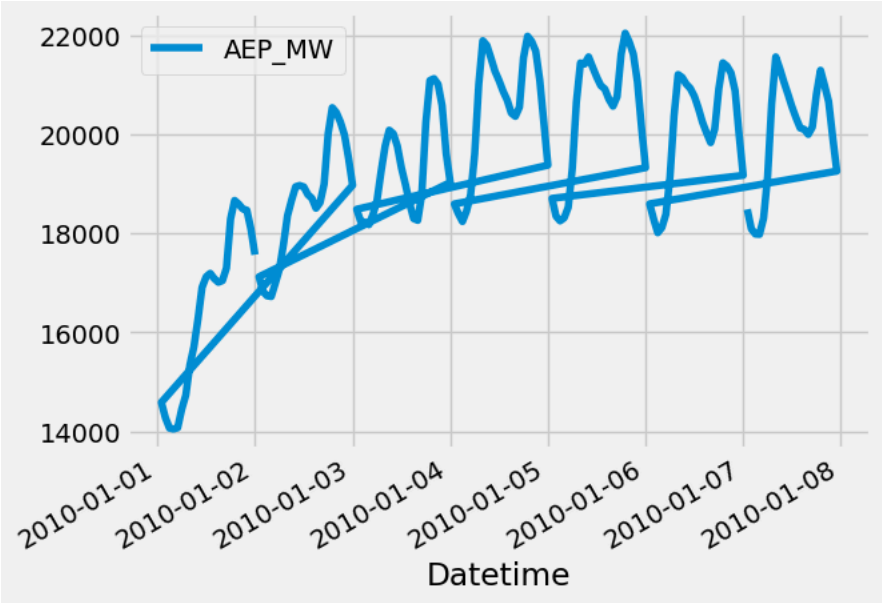
```
train = df.loc[df.index < '01-01-2015']
test = df.loc[df.index >= '01-01-2015']

fig, ax = plt.subplots(figsize=(15, 5))
train.plot(ax=ax, label = 'Training set',title='Data Train/Test Split')
test.plot(ax=ax, label = 'Test Set')
ax.axvline('01-01-2015', color ='black', ls='--')
ax.legend(['Training Set','Test Set'])
plt.show()
```




```
df.loc[(df.index > '01-01-2010') & (df.index < '01-08-2010' )].plot()
```

```
<Axes: xlabel='Datetime'>
```



FEATURE CREATION

```
df.index.hour
```

```
Index([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10,
       ...
       15, 16, 17, 18, 19, 20, 21, 22, 23,  0],
      dtype='int32', name='Datetime', length=121273)
```

```
def create_features(df):

    df['hour'] = df.index.hour
    df['dayofweek'] = df.index.dayofweek
    df['quarter'] = df.index.quarter
    df['month'] = df.index.month
    df['year'] = df.index.year
    df['dayofyear'] = df.index.dayofyear
    return df

df = create_features(df)
```

VISUALIZE OUR FEATURE /TARGET RELATIONSHIP

```
print(df.columns)

```

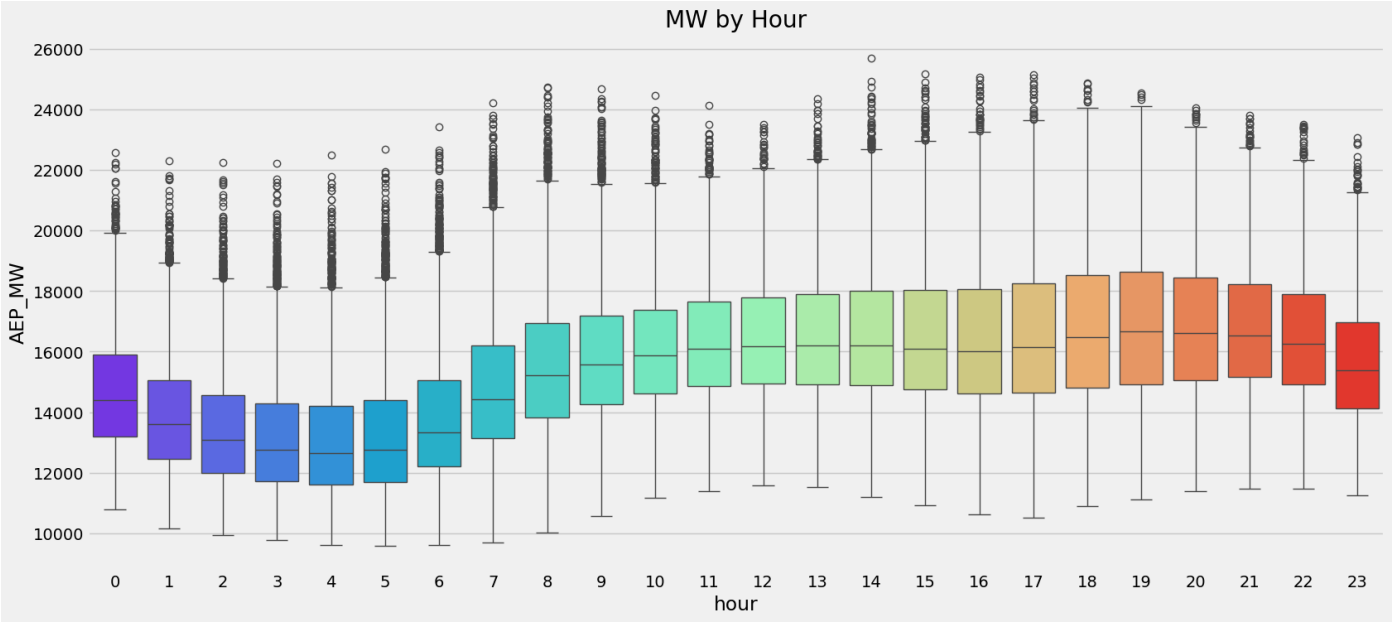
```
Index(['AEP_MW', 'hour', 'dayofweek', 'quarter', 'month', 'year', 'dayofyear'], dtype='object')
```

```
fig, ax = plt.subplots(figsize=(18,8))
sns.boxplot(data=df, x='hour', y ='AEP_MW', palette ='rainbow')
ax.set_title('MW by Hour')
plt.show()
```

```
<ipython-input-37-148aa888781f>:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend

  sns.boxplot(data=df, x='hour', y ='AEP_MW', palette ='rainbow')
```

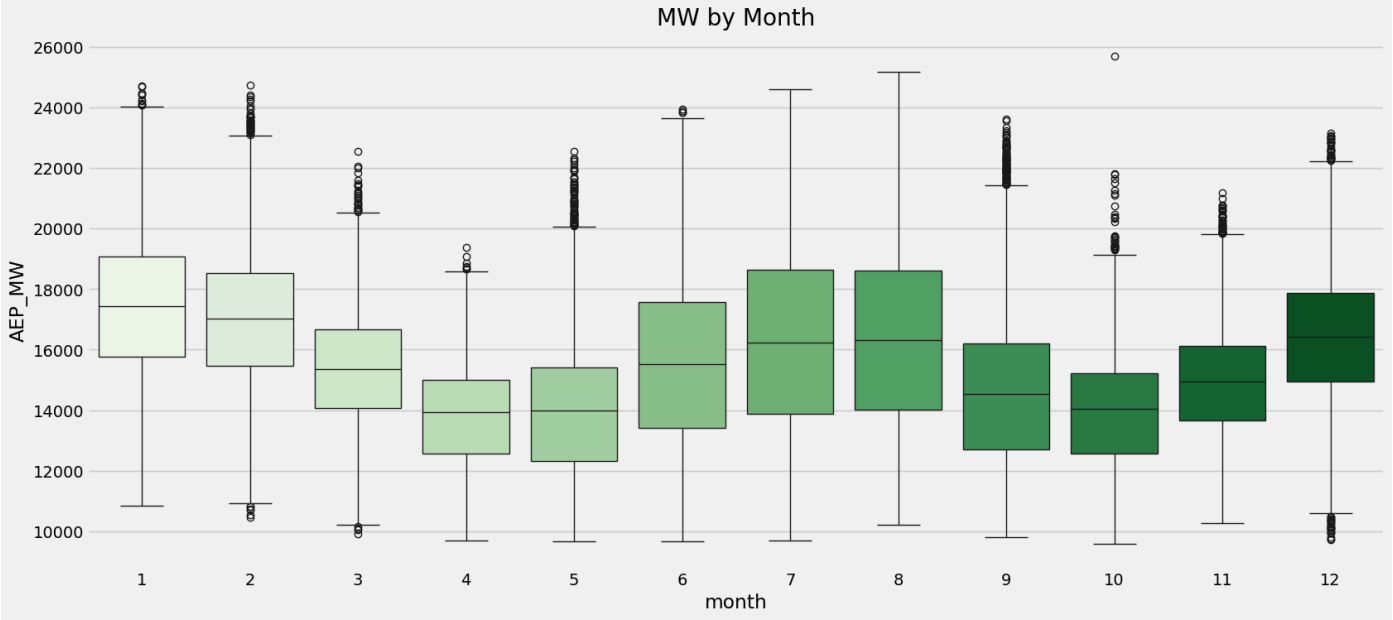



```
fig, ax = plt.subplots(figsize=(18,8))
sns.boxplot(data=df, x='month', y ='AEP_MW', palette ='Greens')
ax.set_title('MW by Month')
plt.show()
```

```
<ipython-input-38-c92272204d98>:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend

  sns.boxplot(data=df, x='month', y ='AEP_MW', palette ='Greens')
```



CREATE A MODEL

```
train = create_features(train)
test =  create_features(test)
FEATURES = ['hour', 'dayofweek', 'quarter', 'month', 'year', 'dayofyear']
TARGET = 'AEP_MW'


```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a
  df['quarter'] = df.index.quarter
<ipython-input-35-e87537c8a040>:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a
  df['month'] = df.index.month
<ipython-input-35-e87537c8a040>:7: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

```
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a
  df['dayofweek'] = df.index.dayofweek
<ipython-input-35-e87537c8a040>:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a
  df['quarter'] = df.index.quarter
<ipython-input-35-e87537c8a040>:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a
  df['month'] = df.index.month
<ipython-input-35-e87537c8a040>:7: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a
  df['year'] = df.index.year
<ipython-input-35-e87537c8a040>:8: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a
  df['dayofyear'] = df.index.dayofyear
```

```
X_train = train[FEATURES]
y_train = train[TARGET]

X_test = test[FEATURES]
y_test = test[TARGET]
```

```
!pip install -U xgboost
```

```
Requirement already satisfied: xgboost in /usr/local/lib/python3.11/dist-packages (2.1.4)
Requirement already satisfied: numpy in /usr/local/lib/python3.11/dist-packages (from xgboost) (1.26.4)
Requirement already satisfied: nvidia-nccl-cu12 in /usr/local/lib/python3.11/dist-packages (from xgboost) (2.21.5)
Requirement already satisfied: scipy in /usr/local/lib/python3.11/dist-packages (from xgboost) (1.13.1)
```

```
import xgboost
print(xgboost.__version__)
```

```
2.1.4
```

```
reg = xgb.XGBRegressor(n_estimators=1000,early_stopping_rounds=50, learning_rate=0.001)

reg.fit(X_train, y_train,
    eval_set=[(X_train, y_train), (X_test, y_test)],
    verbose=100
)


```

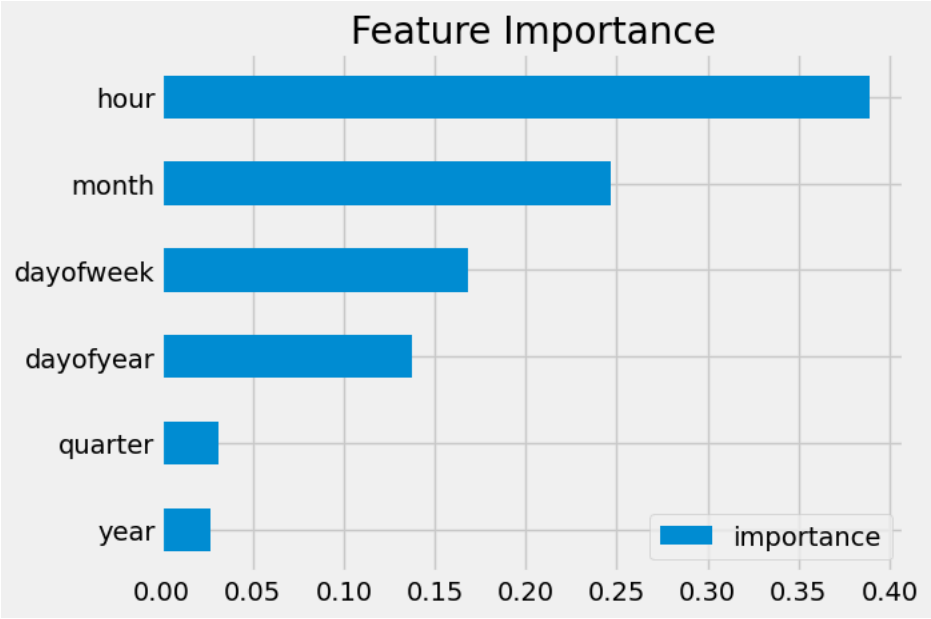
```
[0]   validation_0-rmse:2571.77747   validation_1-rmse:2683.12724
[100] validation_0-rmse:2418.78537   validation_1-rmse:2543.72673
[200] validation_0-rmse:2285.21737   validation_1-rmse:2424.24267
[300] validation_0-rmse:2168.46028   validation_1-rmse:2320.84646
[400] validation_0-rmse:2066.27063   validation_1-rmse:2231.99828
[500] validation_0-rmse:1977.35291   validation_1-rmse:2156.44867
[600] validation_0-rmse:1900.68213   validation_1-rmse:2095.13001
[700] validation_0-rmse:1832.44385   validation_1-rmse:2040.87265
[800] validation_0-rmse:1772.78374   validation_1-rmse:1993.65343
[900] validation_0-rmse:1720.60363   validation_1-rmse:1949.47821
[999] validation_0-rmse:1674.54889   validation_1-rmse:1911.14625
```

```
XGBRegressor

XGBRegressor(base_score=None, booster=None, callbacks=None,
             colsample_bylevel=None, colsample_bynode=None,
             colsample_bytree=None, device=None, early_stopping_rounds=50,
             enable_categorical=False, eval_metric=None, feature_types=None,
             gamma=None, grow_policy=None, importance_type=None,
             interaction_constraints=None, learning_rate=0.001, max_bin=None,
             max_cat_threshold=None, max_cat_to_onehot=None,
             max_delta_step=None, max_depth=None, max_leaves=None,
             min_child_weight=None, missing=nan, monotone_constraints=None,
             multi_strategy=None, n_estimators=1000, n_jobs=None,
             num_parallel_tree=None, random_state=None, ...)
```

FEATURE IMPORTANCE

```
fi = pd.DataFrame(data=reg.feature_importances_,
             index=reg.feature_names_in_,
             columns=['importance'])
fi.sort_values('importance').plot(kind='barh', title ='Feature Importance')
plt.show()
```



FORECAST ON TEST

```
test['prediction'] = reg.predict(X_test)
```

```
<ipython-input-62-33b5c0e814cb>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cc
  test['prediction'] = reg.predict(X_test)
```

```
df.columns
```

```
Index(['AEP_MW', 'hour', 'dayofweek', 'quarter', 'month', 'year', 'dayofyear'], dtype='object')
```

```
print(type(test))
print(test.head())
```

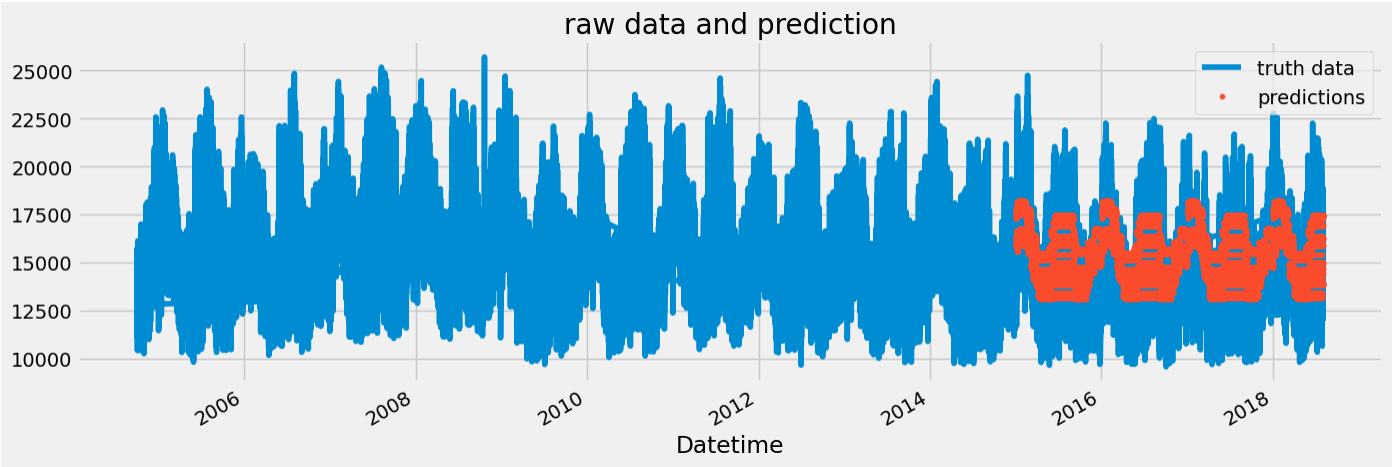
```
<class 'pandas.core.frame.DataFrame'>
                     AEP_MW  hour  dayofweek  quarter  month  year  \
Datetime
2015-01-01 00:00:00  16375.0     0          3        1      1  2015
2015-12-31 01:00:00  12415.0     1          3        4     12  2015
2015-12-31 02:00:00  12087.0     2          3        4     12  2015
2015-12-31 03:00:00  12010.0     3          3        4     12  2015
2015-12-31 04:00:00  11972.0     4          3        4     12  2015

                     dayofyear    prediction
Datetime
2015-01-01 00:00:00          1  15923.991211
2015-12-31 01:00:00        365  14933.623047
2015-12-31 02:00:00        365  14924.480469
2015-12-31 03:00:00        365  14924.480469
2015-12-31 04:00:00        365  14924.480469
```

```
print(f"X_test shape: {X_test.shape}")
print(f"test shape: {test.shape}")
```

```
X_test shape: (31440, 6)
test shape: (31440, 8)
```

```
df = df.merge(test[['prediction']], how='left', left_index=True, right_index=True)
ax = df[['AEP_MW']].plot(figsize=(15, 5))
df['prediction'].plot(ax=ax, style='.')
plt.legend(['truth data', 'predictions'])
ax.set_title('raw data and prediction')
plt.show()
```




```
score = np.sqrt(mean_squared_error(test['AEP_MW'], test['prediction']))
```