	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

Start coding or generate with AI.

Project Python

```
import numpy as np
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense, Dropout
from tensorflow.keras.callbacks import EarlyStopping
import matplotlib.pyplot as plt
```

Load your dataset

```
import pandas as pd

# Try specifying a different delimiter if it's not a comma
data = pd.read_csv('/content/Daily Avg. Humidity.csv', sep=';')  # Try with semicolon
```

```
print(data.head())
```

```
                              Station :Khulna
0                           Daily & Monthly Avera...
1                           ----------------------...
2  Year,Month,Dt(01,02,03,04,05,06,07,08,09,10,11...
3  1993, 1, 82, 86, 82, 84, 81, 79, 83, 85, 86, 8...
4  1993, 2, 76, 76, 72, 68, 66, 74, 70, 75, 80, 7...
```

Select features (assuming 'Temperature' is the target and others are features)

```
# Print the actual column names to check for discrepancies
print(data.columns)

# Modify the 'features' list to match the actual column names
features = ['Year', 'Month', 'avg']  # Example: Corrected column names
# If your column names have spaces, try enclosing them in backticks: `Daily Avg`
# Adjust based on the output of data.columns

# Select the desired columns
data = data[features]
```

```
Index(['                                Station :Khulna       '], dtype='object')
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-29-513968a9f5b9> in <cell line: 10>()
      8
      9 # Select the desired columns
---> 10 data = data[features]

                              2 frames
/usr/local/lib/python3.10/dist-packages/pandas/core/indexes/base.py in _raise_if_missing(self, key, indexer, axis_name)
   6247         if nmissing:
   6248             if nmissing == len(indexer):
-> 6249                 raise KeyError(f"None of [{key}] are in the [{axis_name}]")
   6250
   6251             not_found = list(ensure_index(key)[missing_mask.nonzero()[0]].unique())

KeyError: "None of [Index(['Year', 'Month', 'avg'], dtype='object')] are in the [columns]"
```

Start coding or generate with AI.

Temperature Prediction with Python and Machine Learning FOR Dhaka City Corporation.

```
import pandas as pd
```

```
weather = pd.read_csv("/content/weather.csv", index_col="DATE")
```

```
weather
```

	STATION	NAME	PRCP	TAVG	TMAX	TMIN
DATE						
1990-01-01	BGM00041923	TEJGAON, BG	0.00	63	74.0	53.0
1990-01-03	BGM00041923	TEJGAON, BG	0.00	61	75.0	52.0
1990-01-04	BGM00041923	TEJGAON, BG	NaN	64	NaN	53.0
1990-01-06	BGM00041923	TEJGAON, BG	0.00	63	74.0	53.0
1990-01-07	BGM00041923	TEJGAON, BG	0.00	64	77.0	55.0
...	...	...	...	...	...	...
2024-10-21	BGM00041923	TEJGAON, BG	0.00	83	NaN	76.0
2024-10-22	BGM00041923	TEJGAON, BG	0.00	86	NaN	77.0
2024-10-23	BGM00041923	TEJGAON, BG	0.10	83	NaN	NaN
2024-10-24	BGM00041923	TEJGAON, BG	0.61	76	82.0	NaN
2024-10-25	BGM00041923	TEJGAON, BG	0.01	83	90.0	72.0

8403 rows × 6 columns

```
null_pct = weather.apply(pd.isnull).sum()/weather.shape[0]
null_pct
```

	0
STATION	0.000000
NAME	0.000000
PRCP	0.114007
TAVG	0.000000
TMAX	0.124360
TMIN	0.669166

dtype: float64

```
weather.apply(pd.isnull).sum()
```

	0
STATION	0
NAME	0
PRCP	958
TAVG	0
TMAX	1045
TMIN	5623

dtype: int64

```
valid_columns = weather.columns[null_pct < .05]
```

```
valid_columns
```

```
Index(['STATION', 'NAME', 'TAVG'], dtype='object')
```

```
weather = weather[valid_columns].copy()
```

```
weather.columns = weather.columns.str.lower()
```

```
weather
```

DATE	station	name	tavg
1990-01-01	BGM00041923	TEJGAON, BG	63
1990-01-03	BGM00041923	TEJGAON, BG	61
1990-01-04	BGM00041923	TEJGAON, BG	64
1990-01-06	BGM00041923	TEJGAON, BG	63
1990-01-07	BGM00041923	TEJGAON, BG	64
...	...	...	...
2024-10-21	BGM00041923	TEJGAON, BG	83
2024-10-22	BGM00041923	TEJGAON, BG	86
2024-10-23	BGM00041923	TEJGAON, BG	83
2024-10-24	BGM00041923	TEJGAON, BG	76
2024-10-25	BGM00041923	TEJGAON, BG	83

8403 rows × 3 columns

```
weather['tavg'] = weather['tavg'].fillna(weather['tavg'].mean())
# Display the modified dataset
print(weather)
print(weather.info())
```

```
                station         name  tavg
DATE
1990-01-01  BGM00041923  TEJGAON, BG    63
1990-01-03  BGM00041923  TEJGAON, BG    61
1990-01-04  BGM00041923  TEJGAON, BG    64
1990-01-06  BGM00041923  TEJGAON, BG    63
1990-01-07  BGM00041923  TEJGAON, BG    64
...                ...          ...   ...
2024-10-21  BGM00041923  TEJGAON, BG    83
2024-10-22  BGM00041923  TEJGAON, BG    86
2024-10-23  BGM00041923  TEJGAON, BG    83
2024-10-24  BGM00041923  TEJGAON, BG    76
2024-10-25  BGM00041923  TEJGAON, BG    83

[8403 rows x 3 columns]
<class 'pandas.core.frame.DataFrame'>
Index: 8403 entries, 1990-01-01 to 2024-10-25
Data columns (total 3 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   station  8403 non-null   object
 1   name     8403 non-null   object
 2   tavg     8403 non-null   int64
dtypes: int64(1), object(2)
memory usage: 262.6+ KB
None
```

```
weather = weather.ffill()
```

```
weather.apply(pd.isnull).sum()
```

	0
station	0
name	0
tavg	0

dtype: int64

```
weather.apply(lambda x: (x == 9999).sum())
```

	0
station	0
name	0
tavg	0

dtype: int64

```
weather.dtypes
```

	0
station	object
name	object
tavg	int64

dtype: object

```
weather.index
```

```
Index(['1990-01-01', '1990-01-03', '1990-01-04', '1990-01-06', '1990-01-07',
       '1990-01-08', '1990-01-09', '1990-01-10', '1990-01-12', '1990-01-13',
       ...
       '2024-10-16', '2024-10-17', '2024-10-18', '2024-10-19', '2024-10-20',
       '2024-10-21', '2024-10-22', '2024-10-23', '2024-10-24', '2024-10-25'],
      dtype='object', name='DATE', length=8403)
```

```
weather.index = pd.to_datetime(weather.index)
```

```
weather.index.year.value_counts().sort_index()
```

DATE	count
1990	257
1991	290
1992	343
1993	321
1994	250
1995	289
1996	284
1997	166
1998	172
1999	132
2000	209
2001	278
2002	225
2003	201
2004	193
2005	230
2006	291
2007	214
2008	170
2009	290
2010	269
2011	99
2012	148
2013	185
2014	318
2015	329
2016	312
2017	296
2018	311
2019	221

Start coding or generate with AI.

2021	25

Filling with Mean, Median, or Mode

```
weather['tavg'] = weather['tavg'].fillna(weather['tavg'].mean())
# Display the modified dataset
print(weather)
print(weather.info())
```

```
                station         name  tavg
DATE
1990-01-01  BGM00041923  TEJGAON, BG    63
1990-01-03  BGM00041923  TEJGAON, BG    61
1990-01-04  BGM00041923  TEJGAON, BG    64
1990-01-06  BGM00041923  TEJGAON, BG    63
1990-01-07  BGM00041923  TEJGAON, BG    64
...                ...           ...   ...
2024-10-21  BGM00041923  TEJGAON, BG    83
2024-10-22  BGM00041923  TEJGAON, BG    86
```

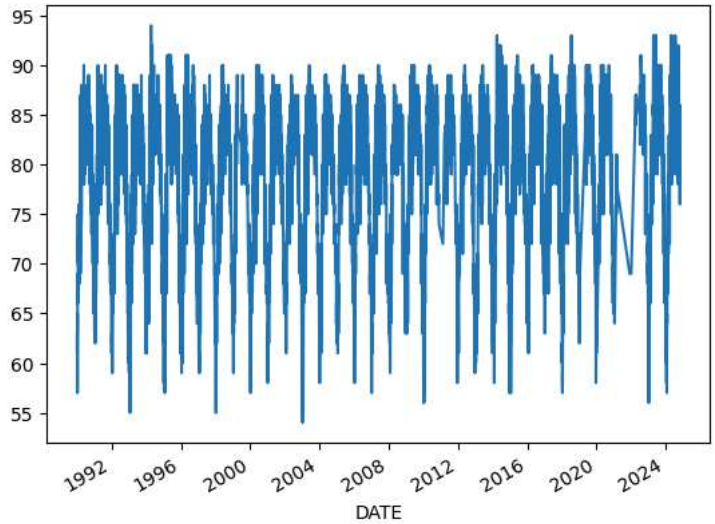
```
2024-10-23  BGM00041923  TEJGAON, BG    83
2024-10-24  BGM00041923  TEJGAON, BG    76
2024-10-25  BGM00041923  TEJGAON, BG    83

[8403 rows x 3 columns]
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 8403 entries, 1990-01-01 to 2024-10-25
Data columns (total 3 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   station  8403 non-null   object
 1   name     8403 non-null   object
 2   tavg     8403 non-null   int64
dtypes: int64(1), object(2)
memory usage: 262.6+ KB
None
```

```
weather.index.year.value_counts().sort_index()
```

DATE	count
1990	257
1991	290
1992	343
1993	321
1994	250
1995	289
1996	284
1997	166
1998	172
1999	132
2000	209
2001	278
2002	225
2003	201
2004	193
2005	230
2006	291
2007	214
2008	170
2009	290
2010	269
2011	99
2012	148
2013	185
2014	318
2015	329
2016	312
2017	296
2018	311
2019	221

```
weather["tavg"].plot()
```

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



Start coding or generate with AI.

Tomorrow Weather Pattern

```
weather["target"] = weather.shift(-1)["tavg"]
```

```
weather
```

DATE	station	name	tavg	target
1990-01-01	BGM00041923	TEJGAON, BG	63	61.0
1990-01-03	BGM00041923	TEJGAON, BG	61	64.0
1990-01-04	BGM00041923	TEJGAON, BG	64	63.0
1990-01-06	BGM00041923	TEJGAON, BG	63	64.0
1990-01-07	BGM00041923	TEJGAON, BG	64	65.0
...	...	...	...	...
2024-10-21	BGM00041923	TEJGAON, BG	83	86.0
2024-10-22	BGM00041923	TEJGAON, BG	86	83.0
2024-10-23	BGM00041923	TEJGAON, BG	83	76.0
2024-10-24	BGM00041923	TEJGAON, BG	76	83.0
2024-10-25	BGM00041923	TEJGAON, BG	83	NaN

8403 rows × 4 columns

```
weather = weather.ffill()
```

```
weather
```

DATE	station	name	tavg	target
1990-01-01	BGM00041923	TEJGAON, BG	63	61.0
1990-01-03	BGM00041923	TEJGAON, BG	61	64.0
1990-01-04	BGM00041923	TEJGAON, BG	64	63.0
1990-01-06	BGM00041923	TEJGAON, BG	63	64.0
1990-01-07	BGM00041923	TEJGAON, BG	64	65.0
...	...	...	...	...
2024-10-21	BGM00041923	TEJGAON, BG	83	86.0
2024-10-22	BGM00041923	TEJGAON, BG	86	83.0
2024-10-23	BGM00041923	TEJGAON, BG	83	76.0
2024-10-24	BGM00041923	TEJGAON, BG	76	83.0
2024-10-25	BGM00041923	TEJGAON, BG	83	83.0

8403 rows × 4 columns

```
from sklearn.linear_model import Ridge

rr = Ridge(alpha=.1)
```

```
predictors = weather.columns[~weather.columns.isin(["target", "name", "station"])]
```

```
predictors
```

```
Index(['tavg'], dtype='object')
```

```
def backtest(weather, model, predictors, start=3650, step=90):
    all_predictions = []

    for i in range(start, weather.shape[0], step):
        train = weather.iloc[:i,:]
        test = weather.iloc[i:(i+step),:]

        model.fit(train[predictors], train["target"])

        preds = model.predict(test[predictors])
        preds = pd.Series(preds, index=test.index)
        combined = pd.concat([test["target"], preds], axis=1)
        combined.columns = ["actual", "prediction"]
        combined["diff"] = (combined["prediction"] - combined["actual"]).abs()

        all_predictions.append(combined)
    return pd.concat(all_predictions)
```

```
predictions = backtest(weather, rr, predictors)
```

```
from sklearn.metrics import mean_absolute_error, mean_squared_error

mean_absolute_error(predictions["actual"], predictions["prediction"])
```

```
2.065274446896808
```

```
predictions.sort_values("diff", ascending=False)
```

DATE	actual	prediction	diff
2014-02-03	79.0	59.625368	19.374632
2022-01-01	87.0	69.814585	17.185415
2019-01-13	85.0	69.813581	15.186419
2014-02-02	58.0	71.610908	13.610908
2014-12-25	57.0	69.771097	12.771097
...	...	...	...
2007-12-26	67.0	67.003209	0.003209
2015-11-05	79.0	78.997211	0.002789
2017-07-25	79.0	78.997617	0.002383
2017-04-19	79.0	78.997617	0.002383
2018-05-18	79.0	78.999381	0.000619

4753 rows × 3 columns

```
pd.Series(rr.coef_, index=predictors)
```

	0
tavg	0.922487

dtype: float64

```
def pct_diff(old, new):
    return (new - old) / old

def compute_rolling(weather, horizon, col):
    label = f"rolling_{horizon}_{col}"
    weather[label] = weather[col].rolling(horizon).mean()
    weather[f"{label}_pct"] = pct_diff(weather[label], weather[col])
    return weather

rolling_horizons = [3, 14]
for horizon in rolling_horizons:
    for col in ["tavg"]:
        weather = compute_rolling(weather, horizon, col)


def expand_mean(df):
    return df.expanding(1).mean()

for col in ["tavg"]:
    weather[f"month_avg_{col}"] = weather[col].groupby(weather.index.month, group_keys=False).apply(expand_mean)
    weather[f"day_avg_{col}"] = weather[col].groupby(weather.index.day_of_year, group_keys=False).apply(expand_mean)


weather
```

DATE	station	name	tavg	target	rolling_3_tavg	rolling_3_tavg_pct	rolling_14_tavg	rolling_14_tavg_pct	month_avg_tavg
1990-01-01	BGM00041923	TEJGAON, BG	63	61.0	NaN	NaN	NaN	NaN	63.000000
1990-01-03	BGM00041923	TEJGAON, BG	61	64.0	NaN	NaN	NaN	NaN	62.000000
1990-01-04	BGM00041923	TEJGAON, BG	64	63.0	62.666667	0.021277	NaN	NaN	62.666667
1990-01-06	BGM00041923	TEJGAON, BG	63	64.0	62.666667	0.005319	NaN	NaN	62.750000
1990-01-07	BGM00041923	TEJGAON, BG	64	65.0	63.666667	0.005236	NaN	NaN	63.000000
...	...	...	...	...	...	...	...	...	...
2024-10-21	BGM00041923	TEJGAON, BG	83	86.0	83.000000	0.000000	82.928571	0.000861	81.619110
2024-10-22	BGM00041923	TEJGAON, BG	86	83.0	84.000000	0.023810	83.142857	0.034364	81.624837
2024-10-23	BGM00041923	TEJGAON, BG	83	76.0	84.000000	-0.011905	83.285714	-0.003431	81.626632
2024-10-24	BGM00041923	TEJGAON, BG	76	83.0	81.666667	-0.069388	82.714286	-0.081174	81.619296
2024-10-25	BGM00041923	TEJGAON, BG	83	83.0	80.666667	0.028926	82.642857	0.004322	81.621094

8403 rows × 10 columns

```
weather = weather.iloc[14:,:]
weather = weather.fillna(0)
```

```
predictors = weather.columns[~weather.columns.isin(["target", "name", "station"])]
```

```
predictors
```

```
Index(['tavg', 'rolling_3_tavg', 'rolling_3_tavg_pct', 'rolling_14_tavg',
       'rolling_14_tavg_pct', 'month_avg_tavg', 'day_avg_tavg'],
      dtype='object')
```

```
predictions = backtest(weather, rr, predictors)
mean_absolute_error(predictions["actual"], predictions["prediction"])
```

```
1.9673166596363283
```

```
predictors.sort_values("diff", ascending=False)
```

```
<ipython-input-41-5bad40bdd310>:1: FutureWarning: Starting with pandas version 3.0 all arguments of sort_values will be keyword-only.
  predictors.sort_values("diff", ascending=False)
(Index(['tavg', 'rolling_3_tavg_pct', 'rolling_3_tavg', 'rolling_14_tavg_pct',
        'rolling_14_tavg', 'month_avg_tavg', 'day_avg_tavg'],
       dtype='object'),
 array([0, 2, 1, 4, 3, 5, 6]))
```

```
weather.loc["1990-03-07": "1990-03-17"]
```

DATE	station	name	tavg	target	rolling_3_tavg	rolling_3_tavg_pct	rolling_14_tavg	rolling_14_tavg_pct	month_avg_tavg
1990-03-07	BGM00041923	TEJGAON, BG	71	72.0	73.000000	-0.027397	72.071429	-0.014866	73.000000
1990-03-08	BGM00041923	TEJGAON, BG	72	81.0	73.000000	-0.013699	72.142857	-0.001980	72.833333
1990-03-11	BGM00041923	TEJGAON, BG	81	69.0	74.666667	0.084821	72.785714	0.112856	74.000000
1990-03-12	BGM00041923	TEJGAON, BG	69	70.0	74.000000	-0.067568	72.571429	-0.049213	73.375000
1990-03-13	BGM00041923	TEJGAON, BG	70	79.0	73.333333	-0.045455	72.357143	-0.032577	73.000000
1990-03-16	BGM00041923	TEJGAON, BG	79	87.0	72.666667	0.087156	73.000000	0.082192	73.600000
1990-03-17	BGM00041923	TEJGAON, BG	87	83.0	78.666667	0.105932	73.928571	0.176812	74.818182

```
predictions["diff"].round().value_counts().sort_index() / predictions.shape[0]
```

diff	count
0.0	0.166702
1.0	0.303018
2.0	0.239291
3.0	0.144334
4.0	0.074066
5.0	0.035451
6.0	0.018569
7.0	0.008019
8.0	0.004220
9.0	0.002532
10.0	0.001688
11.0	0.000422
12.0	0.000633
13.0	0.000422
16.0	0.000211
17.0	0.000211
19.0	0.000211

dtype: float64

```
mean_squared_error(predictions["actual"], predictions["prediction"])
```

```
6.638035649016272
```

```
predictions.sort_values("diff", ascending=False)
```

DATE	actual	prediction	diff
2022-01-01	87.0	68.479289	18.520711
2019-01-13	85.0	67.595650	17.404350
2014-02-03	79.0	62.828462	16.171538
2014-02-02	58.0	70.868049	12.868049
2021-01-18	78.0	65.223965	12.776035
...	...	...	...
2024-08-07	85.0	85.002833	0.002833
2023-01-12	63.0	63.002649	0.002649
2007-12-07	69.0	69.001455	0.001455
2009-06-05	85.0	85.001127	0.001127
2023-11-23	75.0	75.000713	0.000713

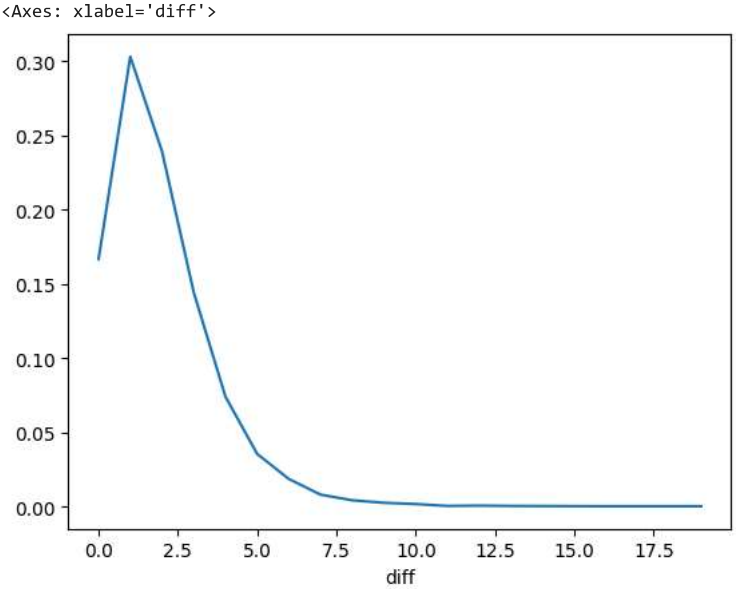
4739 rows × 3 columns

```
weather.loc["1990-03-07": "1990-03-17"]
```

DATE	station	name	tavg	target	rolling_3_tavg	rolling_3_tavg_pct	rolling_14_tavg	rolling_14_tavg_pct	month_avg_tavg	c
1990-03-07	BGM00041923	TEJGAON, BG	71	72.0	73.000000	-0.027397	72.071429	-0.014866	73.000000	
1990-03-08	BGM00041923	TEJGAON, BG	72	81.0	73.000000	-0.013699	72.142857	-0.001980	72.833333	
1990-03-11	BGM00041923	TEJGAON, BG	81	69.0	74.666667	0.084821	72.785714	0.112856	74.000000	
1990-03-12	BGM00041923	TEJGAON, BG	69	70.0	74.000000	-0.067568	72.571429	-0.049213	73.375000	
1990-03-13	BGM00041923	TEJGAON, BG	70	79.0	73.333333	-0.045455	72.357143	-0.032577	73.000000	
1990-03-16	BGM00041923	TEJGAON, BG	79	87.0	72.666667	0.087156	73.000000	0.082192	73.600000	
1990-03-17	BGM00041923	TEJGAON, BG	87	83.0	78.666667	0.105932	73.928571	0.176812	74.818182	

```
(predictions["diff"].round().value_counts().sort_index() / predictions.shape[0]).plot()
```

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

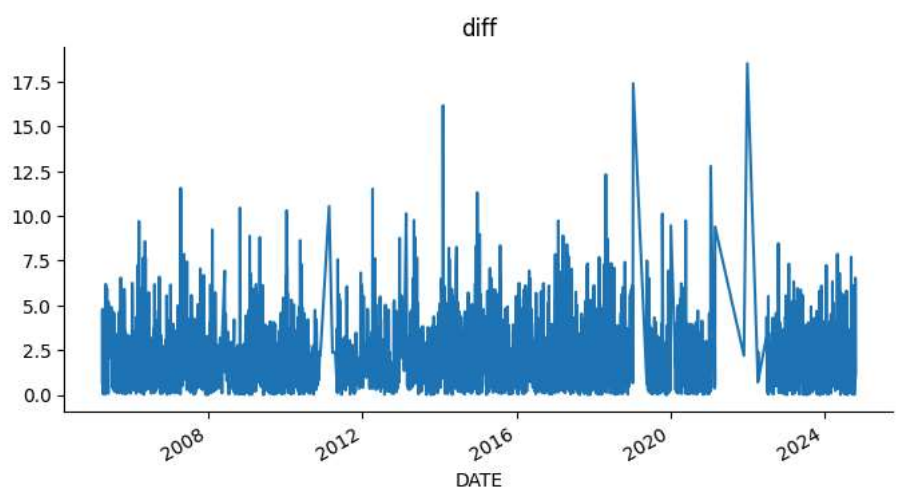


```
predictions
```

DATE	actual	prediction	diff
2005-03-25	83.0	78.308554	4.691446
2005-03-28	82.0	81.434499	0.565501
2005-03-29	77.0	80.903031	3.903031
2005-03-30	74.0	78.795638	4.795638
2005-03-31	81.0	76.718622	4.281378
...	...	...	...
2024-10-21	86.0	82.623531	3.376469
2024-10-22	83.0	84.120212	1.120212
2024-10-23	76.0	82.543249	6.543249
2024-10-24	83.0	78.403666	4.596334
2024-10-25	83.0	82.005782	0.994218

4739 rows × 3 columns

```
from matplotlib import pyplot as plt
predictions['diff'].plot(kind='line', figsize=(8, 4), title='diff')
plt.gca().spines[['top', 'right']].set_visible(False)
```



New Test

```
from sklearn.linear_model import Ridge

reg = Ridge(alpha=.1)


train = weather.loc[:"2020-12-31"]
test = weather.loc["2021-01-01":]


train
```

DATE	station	name	tavg	target	rolling_3_tavg	rolling_3_tavg_pct	rolling_14_tavg	rolling_14_tavg_pct	month_avg_tavg
1990-01-22	BGM00041923	TEJGAON, BG	72	71.0	71.333333	0.009346	65.285714	0.102845	65.133333
1990-01-23	BGM00041923	TEJGAON, BG	71	70.0	72.666667	-0.022936	66.000000	0.075758	65.500000
1990-01-25	BGM00041923	TEJGAON, BG	70	73.0	71.000000	-0.014085	66.428571	0.053763	65.764706
1990-01-27	BGM00041923	TEJGAON, BG	73	73.0	71.333333	0.023364	67.142857	0.087234	66.166667
1990-01-29	BGM00041923	TEJGAON, BG	73	73.0	72.000000	0.013889	67.785714	0.076923	66.526316
...	...	...	...	...	...	...	...	...	...
2020-12-27	BGM00041923	TEJGAON, BG	67	66.0	68.333333	-0.019512	67.214286	-0.003188	68.364501
2020-12-28	BGM00041923	TEJGAON, BG	66	68.0	67.333333	-0.019802	67.000000	-0.014925	68.360759
2020-12-29	BGM00041923	TEJGAON, BG	68	69.0	67.000000	0.014925	67.000000	0.014925	68.360190
2020-12-30	BGM00041923	TEJGAON, BG	69	68.0	67.666667	0.019704	66.928571	0.030950	68.361199
2020-12-31	BGM00041923	TEJGAON, BG	68	69.0	68.333333	-0.004878	66.857143	0.017094	68.360630

7599 rows × 10 columns

```
test
```

DATE	station	name	tavg	target	rolling_3_tavg	rolling_3_tavg_pct	rolling_14_tavg	rolling_14_tavg_pct	month_avg_tavg
2021-01-01	BGM00041923	TEJGAON, BG	69	69.0	68.666667	0.004854	67.071429	0.028754	65.642234
2021-01-02	BGM00041923	TEJGAON, BG	69	68.0	68.666667	0.004854	67.357143	0.024390	65.648084
2021-01-03	BGM00041923	TEJGAON, BG	68	70.0	68.666667	-0.009709	67.571429	0.006342	65.652174
2021-01-04	BGM00041923	TEJGAON, BG	70	70.0	69.000000	0.014493	67.928571	0.030494	65.659722
2021-01-05	BGM00041923	TEJGAON, BG	70	73.0	69.333333	0.009615	68.285714	0.025105	65.667244
...	...	...	...	...	...	...	...	...	...
2024-10-21	BGM00041923	TEJGAON, BG	83	86.0	83.000000	0.000000	82.928571	0.000861	81.619110
2024-10-22	BGM00041923	TEJGAON, BG	86	83.0	84.000000	0.023810	83.142857	0.034364	81.624837
2024-10-23	BGM00041923	TEJGAON, BG	83	76.0	84.000000	-0.011905	83.285714	-0.003431	81.626632
2024-10-24	BGM00041923	TEJGAON, BG	76	83.0	81.666667	-0.069388	82.714286	-0.081174	81.619296
2024-10-25	BGM00041923	TEJGAON, BG	83	83.0	80.666667	0.028926	82.642857	0.004322	81.621094

790 rows × 10 columns

```
reg.fit(train[predictors], train["target"])
```

```
predictions = reg.predict(test[predictors])
```

```
from sklearn.metrics import mean_squared_error

mean_squared_error(test["target"], predictions)
```

```
6.809982018410759
```

```
combined = pd.concat([test["target"], pd.Series(predictions, index=test.index)], axis=1)
combined.columns = ["actual", "predictions"]
```

```
combined
```

	actual	predictions
DATE		
2021-01-01	69.0	68.063660
2021-01-02	68.0	68.051769
2021-01-03	70.0	67.557499
2021-01-04	70.0	68.481596
2021-01-05	73.0	68.439596
...	...	...
2024-10-21	86.0	82.563976
2024-10-22	83.0	84.004835
2024-10-23	76.0	82.471707
2024-10-24	83.0	78.421917