Historical Forecast

Our time series model offers a powerful feature that allows users to retrieve historical forecasts alongside the prospective predictions. This functionality is accessible through the forecast method by setting the `add_history=True` argument.

In [ ]:
```
# | hide
from dotenv import load_dotenv
```

In [ ]:
```
# | hide
load_dotenv()
```

Out[ ]:
```
True
```

In [ ]:
```
import pandas as pd
from nixtlats import TimeGPT
import os
```

In [ ]:
```
timegpt = TimeGPT(token=os.getenv("TIMEGPT_TOKEN"))
```

In [ ]:
```
# | hide
timegpt = TimeGPT()
```

You can test the validate of your token calling the `validate_token` method:

In [ ]:
```
timegpt.validate_token()
```

```
INFO:nixtlats.timegpt:Happy Forecasting! :), If you have questions or need support,
please email ops@nixtla.io
```

Out[ ]:
```
True
```

Now you can start to make forecasts! Let's import an example:

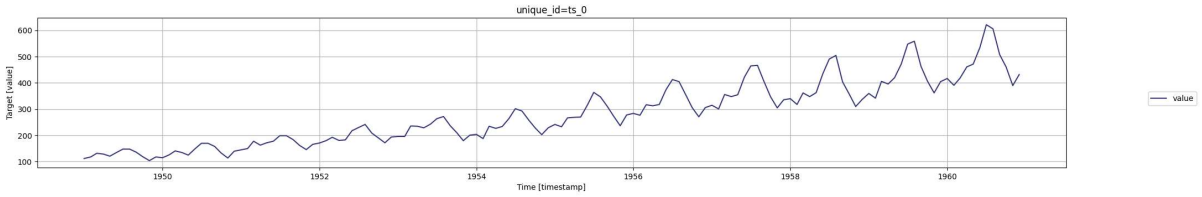
In [ ]:
```
df = pd.read_csv(
    "https://raw.githubusercontent.com/Nixtla/transfer-learning-time-series/main/da
)
df.head()
```

Out[ ]:

	timestamp	value
0	1949-01-01	112
1	1949-02-01	118
2	1949-03-01	132
3	1949-04-01	129
4	1949-05-01	121

In [ ]:
```
timegpt.plot(df, time_col="timestamp", target_col="value")
```

Out[ ]:



Let's add fitted values. When `add_history` is set to True, the output DataFrame will include not only the future forecasts determined by the h argument, but also the historical predictions. Currently, the historical forecasts are not affected by `h`, and have a fix horizon depending on the frequency of the data. The historical forecasts are produced in a rolling window fashion, and concatenated.

In [ ]:
```
timegpt_fcst_with_history_df = timegpt.forecast(
    df=df,
    h=12,
    time_col="timestamp",
    target_col="value",
    add_history=True,
)
```

```
INFO:nixtlats.timegpt:Validating inputs...
INFO:nixtlats.timegpt:Preprocessing dataframes...
INFO:nixtlats.timegpt:Inferred freq: MS
INFO:nixtlats.timegpt:Calling Forecast Endpoint...
INFO:nixtlats.timegpt:Calling Historical Forecast Endpoint...
```

> BEFORE: *18 API Calls | 422590 Tokens | 694.94 Spent*
> AFTER: *20 API Calls | 423130 Tokens | 696.09 Spent*
> **USAGE: 2 API Calls | 540 Tokens | 1.15 Spent**

In [ ]:
```
timegpt_fcst_with_history_df.head()
```

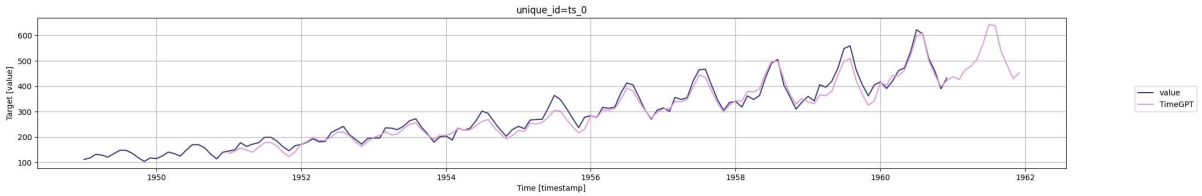
Out[ ]:

	timestamp	TimeGPT
0	1951-01-01	135.483673
1	1951-02-01	144.442398
2	1951-03-01	157.191910
3	1951-04-01	148.769363
4	1951-05-01	140.472946

Let's plot the results. This consolidated view of past and future predictions can be invaluable for understanding the model's behavior and for evaluating its performance over time.

In [ ]:
```
timegpt.plot(df, timegpt_fcst_with_history_df, time_col="timestamp", target_col="va
```

Out[ ]:



Please note, however, that the initial values of the series are not included in these historical forecasts. This is because our model, `TimeGPT`, requires a certain number of initial observations to generate reliable forecasts. Therefore, while interpreting the output, it's important to be aware that the first few observations serve as the basis for the model's predictions and are not themselves predicted values.