

Automate Time Series Forecasting using Auto-TS

[ADVANCED](#)[DATA VISUALIZATION](#)[MACHINE LEARNING](#)[PYTHON](#)[STRUCTURED DATA](#)[SUPERVISED](#)[TIME SERIES FORECASTING](#)

This article was published as a part of the [Data Science Blogathon](#).

"Prediction is very difficult, especially if it's about the future."

I hope you all are doing well! Today I have come up with another interesting article, I hope you got some idea after reading the above beautiful quote. I have explained different automated libraries to automate machine learning and NLP task in my previous articles. Similarly in this article, I will explain "How to automate Time Series Forecasting using Auto-TS".

Auto-TS is a part of AutoML which will automate some of the components of the machine learning pipeline. This automates libraries helps non-experts train a basic machine learning model without being much knowledgeable in the field. Here In this article, I will discuss how to automate a time-series forecasting model implementation using the Auto-TS library.

What is Auto-TS?

It is an open-source python library basically used to automate Time Series Forecasting. It will automatically train multiple time series models using a single line of code, which will help us to choose the best one for our problem statement.

In the python open-source library Auto-TS, `auto-ts.Auto_TimeSeries()` is the main function that you will call with your train data. We can then choose what kind of models you want such as stats, ml, or FB prophet-based models. We can also tune the parameters which will automatically select the best model based on the scoring parameter we want it to be based on. It will return the best model and a dictionary containing predictions for the number of forecast_periods you mentioned (default=2).

Features of Auto-TS library :

- It finds the optimal time series forecasting model using genetic programming optimization.
- It trains naive, statistical, machine learning, and deep learning models, with all possible hyperparameter configurations, and cross-validation.
- It performs data transformations to handle messy data by learning optimal NaN imputation and outlier removal.
- Choice of the combination of metrics for model selection.

Installation:

```
pip install autots OR pip3 install auto-ts OR pip install git+git://github.com/AutoViML/Auto_TS
```

Requirements :

```
dask scikit-learn FB Prophet statsmodels pmdarima XGBoost
```

Import library using :

```
from auto_ts import auto_timeseries
```

Parameters available in auto_timeseries :

```
model = auto_timeseries( score_type='rmse', time_interval='Month', non_seasonal_pdq=None, seasonality=False,
seasonal_period=12, model_type=['Prophet'], verbose=2)
```

You can tune the parameters and analyze the change in model performance. For more details about parameters click [here](#).

Dataset Used :

Here I have used the [Amazon Stock Price](#) dataset from Jan 2006 to Jan 2018, which is downloaded from Kaggle. This library comes up with only train time series forecasting models. The dataset should have a time or date format column.

Initially, load the time series dataset with a time/date column :

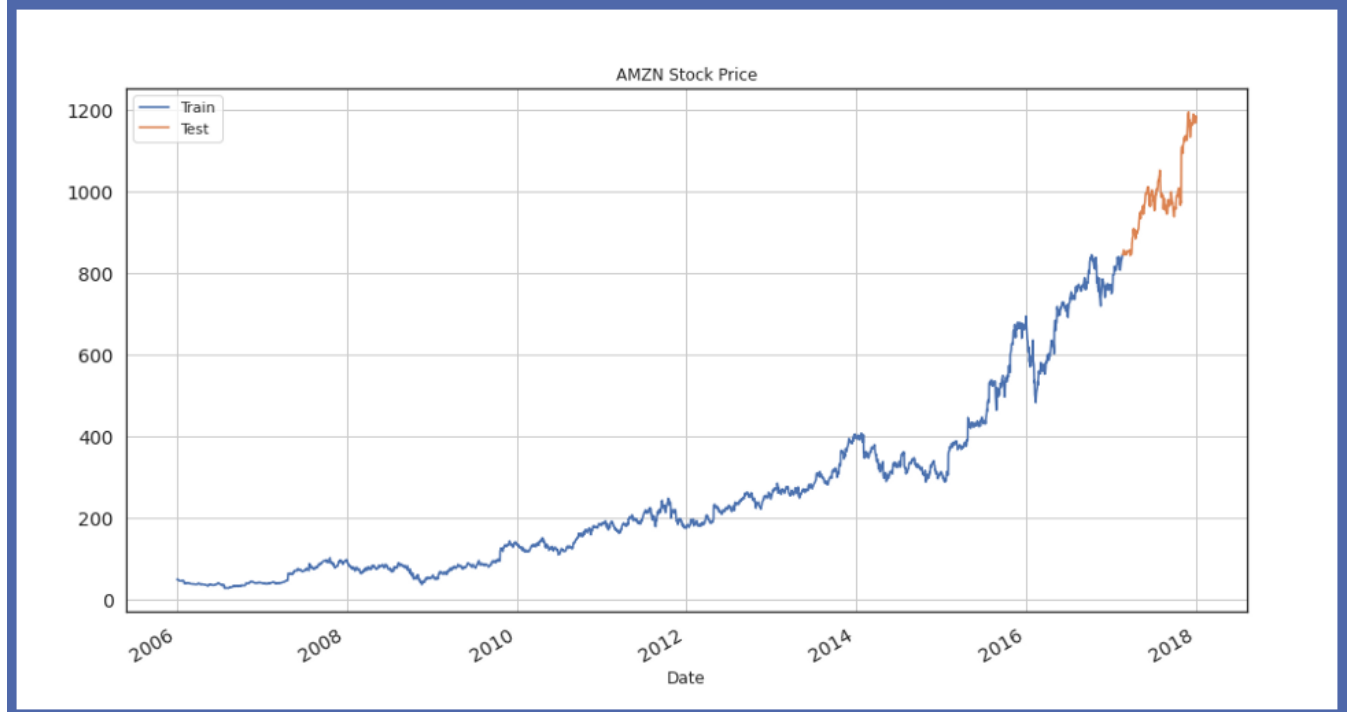
```
df = pd.read_csv("Amazon_Stock_Price.csv", usecols=['Date', 'Close']) df['Date'] = pd.to_datetime(df['Date'])
df = df.sort_values('Date')
```

Now, split the entire data into train and test data :

```
train_df = df.iloc[:2800] test_df = df.iloc[2800:]
```

Now, we will visualize the train test split :

```
train_df.Close.plot(figsize=(15,8), title='AMZN Stock Price', fontsize=14, label='Train')
test_df.Close.plot(figsize=(15,8), title='AMZN Stock Price', fontsize=14, label='Test') plt.legend()
plt.grid() plt.show()
```



Now, let's initialize the Auto-TS model object, and fit the training data :

```
model = auto_timeseries(forecast_period=219, score_type='rmse', time_interval='D', model_type='best')
model.fit(traindata= train_df, ts_column="Date", target="Close")
```

Now let's compare the accuracy of different models :

```
model.get_leaderboard() model.plot_cv_scores()
```

Now let's test our model on test data :

```
future_predictions = model.predict(testdata=219)
```

Finally, visualize test data value and prediction :

Parameters available in auto_timeseries :

```
model = auto_timeseries( score_type='rmse', time_interval='Month', non_seasonal_pdq=None, seasonality=False,
seasonal_period=12, model_type=['Prophet'], verbose=2)
```

Parameters available in the model.fit() :

```
model.fit( traindata=train_data, ts_column=ts_column, target=target, cv=5, sep=", " )
```

Parameters available in model.predict() :

```
predictions = model.predict( testdata = can be either a dataframe or an integer standing for the
forecast_period, model = 'best' or any other string that stands for the trained model )
```

You can play with all these parameters and analyze the performance of our model and then you can select the best suitable model for your problem statement. You can check all these parameters in detail by clicking [here](#).

Conclusion :

In this article, I have discussed that how you can automate the time series model in one line of Python code. Auto-TS does the preprocessing of data, as it removes the outliers from the data and handles messy data by learning optimal NaN imputation. Using just a single line of code, by initializing the Auto-TS object and fitting the train data, it will automatically train multiple time series models such as ARIMA, SARIMAX, FB Prophet, VAR, and come up with the best performing model which is suitable for our problem statement. The result of the model seems to depend on the size of the dataset. If we try to increase the size of the dataset, the result may definitely improve.

EndNote

I hope you enjoyed this article. Any question? Have I missed something? Please reach out on my [LinkedIn](#) or leave a comment below. And finally, ... it doesn't go without saying,

Thank you for reading!

Cheers!!

The media shown in this article are not owned by Analytics Vidhya and is used at the Author's discretion.

Article Url - <https://www.analyticsvidhya.com/blog/2021/04/automate-time-series-forecasting-using-auto-ts/>



[Ronil Patil](#)