Data Driven Investor™

Photo by Markus Spiske on Unsplash

# Predict Stock Prices with Time-Series Statistical Learning

Jacques Sham
May 19 · 9 min read

In my recent stock analytics project, I am planning to predict the S&P 500 by aggregating its predicted stock prices. The very first step is to predict stock prices. Building a model to predict the stock price is not easy work, but the easiest way to predict the stock price is to learn with time-series techniques.

In my mind, there are 3 algorithms to make predictions: Adaptive model, Box-Jerkins method (ARIMA model), and Holt-Winters method; in Python, we can use Facebook Prophet, pmdarima, and statsmodels to help us. Let's use these packages to make predictions on Apple stock price for a prototype predictive model.

Using one of those three techniques is purely predicting stock price with historical performance; it is purely technical analysis. The selection of the time interval of the stock price for model training can have a huge impact on the accuracy of the predictive model. Initially, I have selected the stock price between 1997 and 2018 to be the training data set and selected the stock price between 2019 and May 2020 to be the testing data set. In my initial thought, the longer the time frame provides more insights on the stock price pattern for the model, the more accurate of the predictive model. So, having at least two bear market periods should be sufficient for model training.

## Facebook Prophet

First, let's build a predictive model using Facebook Prophet. Facebook Prophet is an open-source Python package built by Facebook that helps developers to build an adaptive model. Facebook Prophet (Prophet) is a very user-friendly package because the syntax is Sklearn-style. Models built by Prophet require a training data set in Pandas data frame with two columns which must be named with 'ds' and 'y', which represented date and responsive variable, respectively. Then, prepare a data frame with a column of dates you wish to predict where the column name also must be named with 'ds'. To be the convenience, you may obtain the testing data set with dates and stock price, then take the dates column to pass to the model for prediction.

```
# Training data set
X_train = (Source of data…)

# Change column names for Prophet, if not, it will not work
X_train.columns = ['ds', 'y']

# Prepare the testing data set
df_test = (Source of data…)

# Prepare dates for Prophet for prediction, the column also must called 'ds'
X_test = pd.DataFrame(df_test['ds'])

# Train the model and make prediction, same as SKlearn-style syntax
model = Prophet()
model.fit(X_train)
pred = model.predict(X_test)
```

The prediction made by Prophet returns prediction, confidence interval. The column of prediction is called 'yhat'.

Let's evaluate the predictive model accuracy with R-squared. It is calculated with 1 — SSE/SST. The R-squared of this model, unfortunately, is -4.56%. A negative R-square means the model performs worse than taking the average.

Time series models are very sensitive to historical trend pattern; building stock price predictive model is very sensitive to stock price performance that selecting a time frame for training data set is can change the trend pattern by a lot. We know that the stock price of Apple hits the lowest area in 1997 and suffer until the release of ipod in 2000s. The stock price of Apple did not gain momentum until the iPhone gained popularity. If we select the time frame between 2010 and 2018 as training data set. The R-squared raised up to 44.73%. In this example, we can see training data select is an important step to boost accuracy. Let's call the model trained with 1997–2018 data "model_max" and the model trained with 2010–2018 be "model_8yr".

Prediction Apple (AAPL) Stock Price learnt using Facebook Prophet

Both models were able to find the trend of the stock price. model_max consists of a longer time interval where including the years that Apple stock price has a weaker momentum, therefore, we can see the price rising trend of model_max is weaker than model_8yr. Thus, the prediction with model_8yr is closer to the real stock price between 2019 and May 2020.

## pmdarima

Another time-series statistical learning technique we can use is the Box-Jerkin method. In R, there is a very handy package to search the best hyperparameters for ARMIA models, *auto.arima()* in the *forecast* package. pmdarima is the *forecast* package in the Python equivalent to R and aimed to find ARMIA models hyperparameters like *auto.arima()*. Unlike Facebook Prophet, you are only required to prepare response variables without date like auto.arima() in R. To build a model with pmdarima, first, prepare the stock price in a Numpy array format, then, call the auto_arima() from pmdarima, and predict the stock price afterward.

```
# Prepare training data set in ndarray
X_train = (Source of data…)

#Obtain the trading days in testing data
X_test = (Source of data…)

# Declare model object and assign minimum and maximum of
hyperparameters
model = pmdarima.auto_arima(X_train, start_p=1, start_q=1, max_p=3,
max_q=3 m=12,
 max_P=3, max_Q=3, seasonal=True,
 d=1, D=1, max_d=3, max_D=3, trace=True,
 error_action='ignore',
 suppress_warnings=True,
 stepwise=True)

# Print model summary if you would like to
print(model.summary())

# Make prediction, pass a number of predictions you would like to make
# We can pass the number of trading days between 2019 and May 2020
pred = model.predict(X_test.shape[0])
```

The pmdarima follows R style and syntax. The syntax is very similar to the forecast package in R but pmdarima only takes data in NumPy format; you have to turn the

Pandas Data Frame column to a 1-D NumPy array. The syntax for looking for the best hyperparameters for the best ARIMA model is auto_arima() where you can assign the minimum and maximum of hyperparameters. It would return the best ARIMA model and ready to make predictions. You may use .predict() with a number of predictions you would like to make. For example, if we are interested to obtain the stock price of Apple stock between 2019 and May 2020, we may enter 343 (Or the rows in the testing data set) to represents 343 trading days between 2019 and May 2020.

Let’s try building two ARIMA models with two different time frames for training data set like we did — between 1997–2018 and between 2010–2018, and call them model_max and model_8yr again.

As a result, the R-square of model_max and model_8yr are both negative. And if we plot on a line chart, it looks like this:

Prediction Apple (AAPL) Stock Price learnt using pmdarima

As both models trained with pmdarima achieved negative R-square, we may conclude that it is not useful to predict Apple stock price with the Box-Jerkins method.

## Holt-Winters Method

Holt-Winters method is the last time series statistical learning technique we are going over in this article. It used exponential smoothing techniques and the prediction is made purely on historical performance. We can use the ExponentialSmoothing() function in statsmodels to build the predictive models.

The style and syntax of ExponentialSmoothing() is similar to auto_arima() in pmdarima. Like the previous package, you would prepare the stock price in a Numpy array format again and pass it into ExponentialSmoothing(). After that, you may make the prediction after the model is trained by passing the number of predictions you wish to make.

```
# Obtain Training data and Testing data
X_train = (Source of data…)
X_test = (Source of data…)

# Model training, need to add alpha and beta in fit()
# Alpha and beta needs to obtain in grid search
model = ExponentialSmoothing(X_train, trend=‘mul’,
 seasonal=‘mul’, seasonal_periods=4).fit(
 smoothing_level=0.6,smoothing_slope=0.25)

# Make prediction
pred = model.forecast(X_test.shape[0])
```

You do not need a lot of data to build the training data; we can obtain a 1-year interval of the stock price that is more than sufficient for the model. However, the hardest part is to find the best smoothing_level (alpha) and smoothing_slope (beta). In this example, I have tried the smoothing_level be 0.6 and smoothing_slope=0.25 and found the R-square of this model is 41%.

Prediction on Apple (AAPL) stock price using Holt-Winters Method

## The Best Predictive Model of Apple Stock Price

After comparing all models we have built for the predictive model of Apple stock price, we have found out that the predictive model trained with stock price between 2010–2018 using Facebook Prophet is the most accurate as it achieved 44% R-square. The predictive model trained with Holt-winters method achieved 41% R-square, while the models trained with the Box-Jerkins method were both having negative R-square. We would use the model trained with stock price between 2010–2018 using Facebook Prophet to predict future Apple stock price.

## All Stocks are not the Same

The hardest part of making 505 stock prices is that we cannot use the same method or the same time frame for training data set; the same method and hyperparameters used in predicting Apple stock price does not work in other stocks. Let’s pick Lockheed Martin (LMT) for another example. If we train the predictive model for Lockheed Martin with

1997–2018 stock price and 2010–2018 stock price using Facebook Prophet, the R-square are 42% and 34%, respectively. Or if we trained with the Box-Jerkins method and the Holt-winters method with the best hyperparameters, the R-squared are both negative R-square.

Prediction on Lockheed Martin (LMT) stock price learnt using Facebook Prophet

## Conclusion

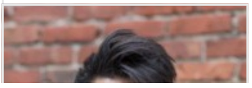
Making prediction on the stock price of different stocks are not the same and each predictive model should be tail-made for each stock. In this experience, we have explored the 3 different Python packages to build predictive models with 3 different time-series statistical learning. We have found out that the same method and hyperparameters cannot be used across 505 stocks to achieve the same level of accuracy. To achieve the highest accuracy for all stocks, we have to use a grid search to obtain the best method and hyperparameters for each stock price predictive model. We know that each company has its own business background and management that play a huge role in its historical stock price, and contribute to the unique stock price trend pattern. The approaches we have used in the article is a purely technical analysis to predict the stock price based on historical performance. It does not interpret the fundamentals of the stocks. Therefore, it is a good starting point but we shall consider adding more fundamental features into the predictive models.

The stock price of both Apple and Lockheed Martin was obtained from Yahoo Finance through yfinance. If you would like to learn how to obtain stock price data, you may learn more from my post on how to obtain the data from Yahoo Finance. If you would like to look at the full code, you may visit my Github page where you can find the code related to this post.

**Jacques Sham - San Francisco Bay Area | Professional Profile | LinkedIn**

View Jacques Sham's profile on LinkedIn, the world's largest professional community. Jacques has 4 jobs listed on their...

www.linkedin.com



## Reference

My Github page on this Post:

**jacquessham/StockAnalytics**

This folder consists the supplemental files for my Medium Post of Predict Stock Price with Time-Series Statistical…

github.com

My Github page on my Stock Analytics Project:

**jacquessham/StockAnalytics**

The goal of this project is to make analysis and develop tools to make investment analysis. In this project, there are…

github.com

My Github page on S&P 500 Prediction:

**jacquessham/StockAnalytics**

S&P 500 is a benchmark index in the United States and it is a significant stock market indicator to the US stock…

github.com

Facebook Prophet Documentation:

**Prophet**

Prophet is a forecasting procedure implemented in R and Python. It is fast and provides completely facebook.github.io

pmdarima Documentation:

**pmdarima: ARIMA estimators for Python - pmdarima 1.6.1 documentation**

brings R's beloved to Python, making an even stronger case for why you don't need R for data science. is 100% Python +...

alkaline-ml.com

Statsmodel Holt-winters Documentation:

**alkaline-ml/pmdarima**

Pmdarima (originally pyramid-arima, for the anagram of 'py' + 'arima') is a statistical library designed to fill the...

github.com

Investment Finance Data Science Predictive Modeling Time Series Forecasting

13

WRITTEN BY

**Jacques Sham**

Follow

A former economist, data scientist in-training, a whisky-lover, an aviation enthusiast, and a gamer. Concern how to use data science to answer questions.

**Data Driven Investor**

Follow

from confusion to clarity not insanity

## More From Medium

**Business Manager: How to Launch a Machine Learning Project?**

Brigitte Maillère in The Startup

**Fine Tuning GPT-2 for Magic the Gathering Flavour Text Generation**

Richard Bownes, PhD in The Startup

**Neural Networks and Image Recognition (CNN's)**

Justin Tennenbaum in The Startup

**Viewing the world through a straw, Part 2**

Nick Weir in The DownLinQ

**Applying Context Aware Spell Checking in Spark NLP**

Alberto Andreotti in spark-nlp

**How fast is fast.ai? Building a world-class pathologist in an evening**

Alex Federation in The Startup

**Feature Selection in Machine Learning**

Shaurya Lalwani in Towards AI — Multidisciplinary Science Journal

**Using Transfer Learning to Detect Malaria Diseases**

Satsawat Natakarnkitkul in Towards AI — Multidisciplinary Science Journal