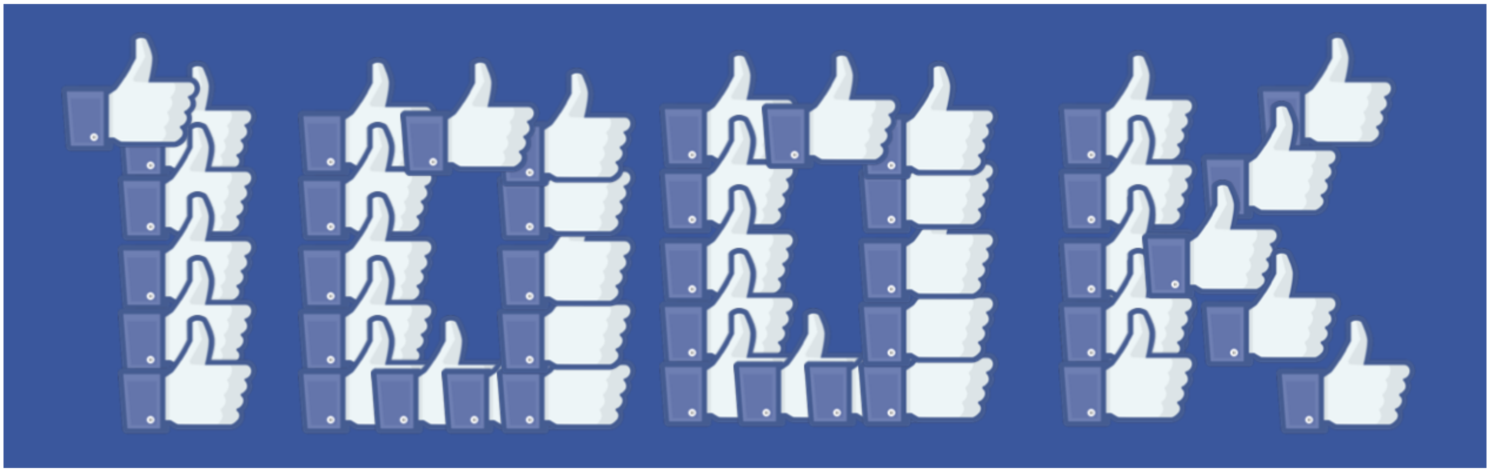


Photo by Author

Forecast the Likes on my Facebook Fan Page — Can I Reach 100k Likes in 6 months?

Using time-series predictions to set data-driven goals for social media influencers.

Eric Weng
May 11 · 7 min read ★

I have a Facebook Fan Page with 96k+ Likes.

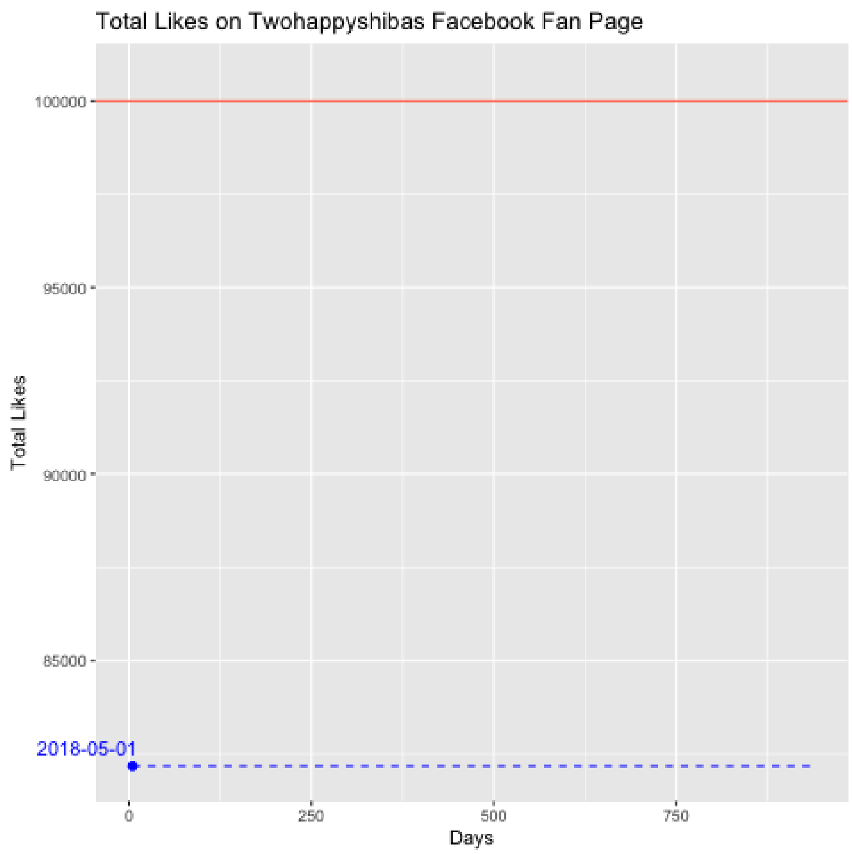
```
www.facebook.com/TwoHappyShibas/
```

> "How many new Likes should I get in this month?"
>
> "How to establish time-bounded goals?"
>
> "I have 96k Likes now, when should I reach 100k?"

Before I became a data analyst, these are some top questions that I asked myself but never answered.

Now, as a data analyst, I can make meaningful insights from data.

I applied the Time Series Analysis to my Facebook Fan Page's two-year historical data, forecasting the number of Likes in the next six months:



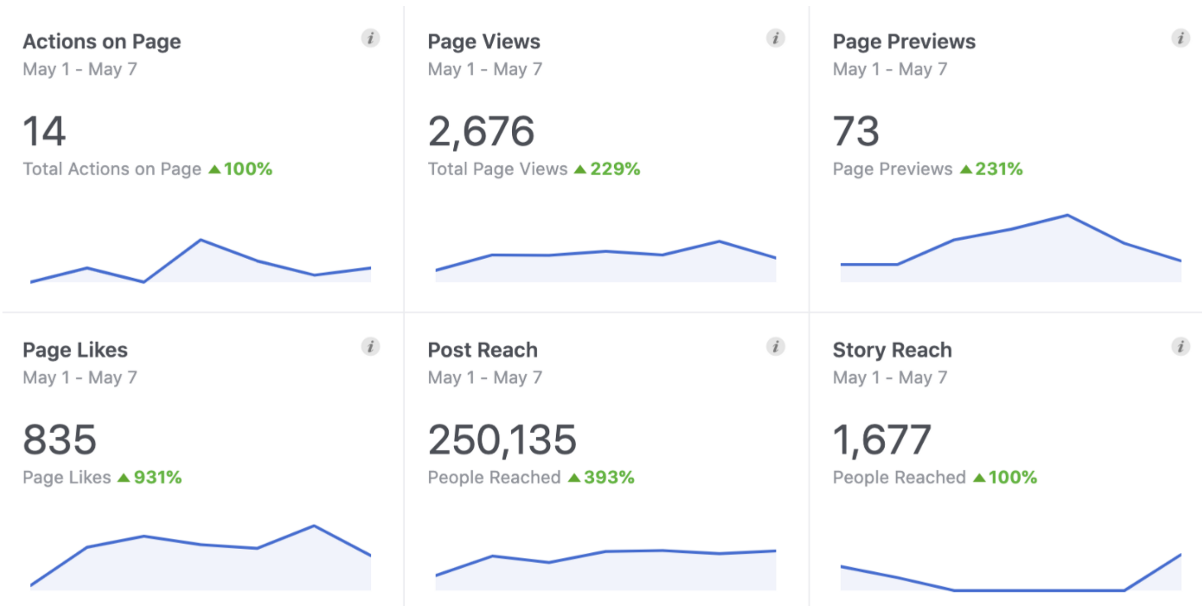
Forecasting Total Likes on https://www.facebook.com/TwoHappyShibas/

With the forecast, I can directly answer the three questions mentioned above:

This data analysis application is extremely helpful for me because now I can gradually make fewer decisions by intuition.

Previously, I can only measure my performance using the *Facebook analysis tool* to compare with the last period, but never know what kind of return is enough and what is not.

But, **after I did this analysis, setting goals becomes more accessible, data-oriented, and realistic.** The forecast can serve as a benchmark, assisting me to create a data-driven timeline to follow and continue making my influence on social media.



Facebook Analysis Tool: showing periodic performance comparisons

Analysis Approaches

1. Define Objectives
2. Data Collection & Preparation
3. Data Exploratory Analysis
4. Time Series Analysis
5. Application & Insights

Define Objectives

> *Problem:*

When operating a Fan Page, setting a goal is always hard when I don't have a reliable benchmark. For instance, I don't know how many new Likes to acquire in a month would be considered a good return.

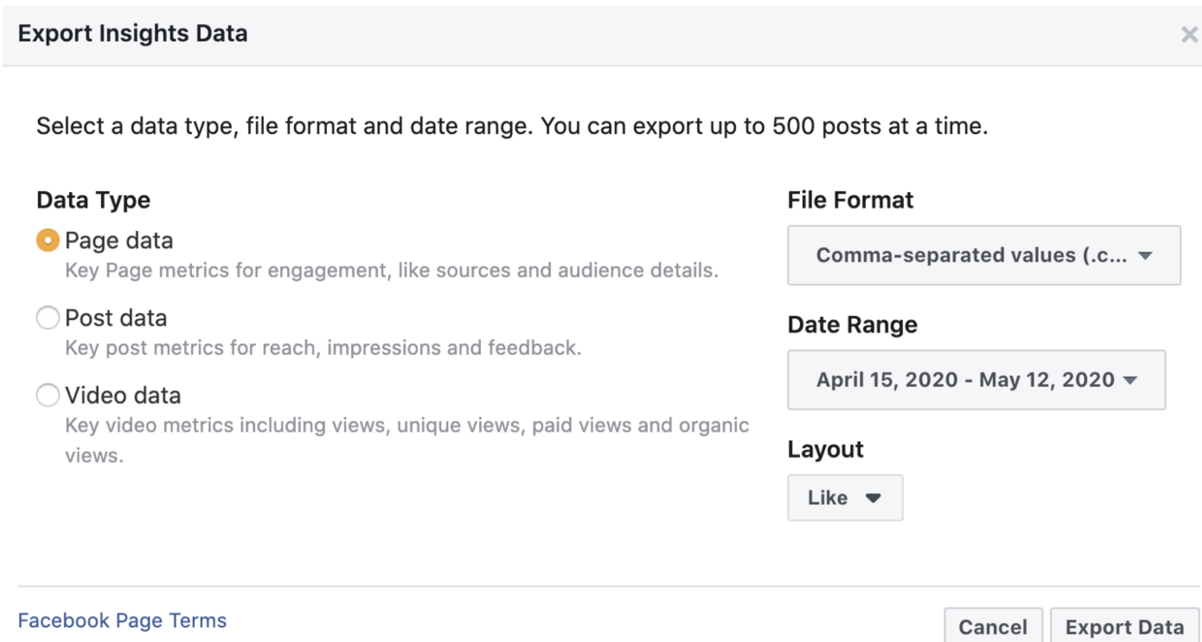
> *Main Objective:*

Predict the number of total Likes on my Facebook Fan Page, helping me to set attainable, realistic data-driven goals in the future.

Data Collection & Preparation

The data is collected on my Facebook Fan Page: www.facebook.com/TwoHappyShibas/

With the export insights data function on Facebook, I am able to get varied data within two years.



Facebook lets the users export data within two years(Screenshot by Author)

Date	Total_Likes
4/27/18	82161
4/28/18	82171
4/29/18	82171
4/30/18	82178
5/1/18	82174
5/2/18	82180

Daily Total Likes (2018/04/27–2020/05/03)

This time I am using the Daily Total Likes Data, it's really suitable with Time Series analysis since it provides daily numbers with Date.

```
> periodicity(like_xts)
Daily periodicity from 2018-04-27 to 2020-05-03
```

Preparation:

The major part of data preparation is to transform the data into a time-series data type.

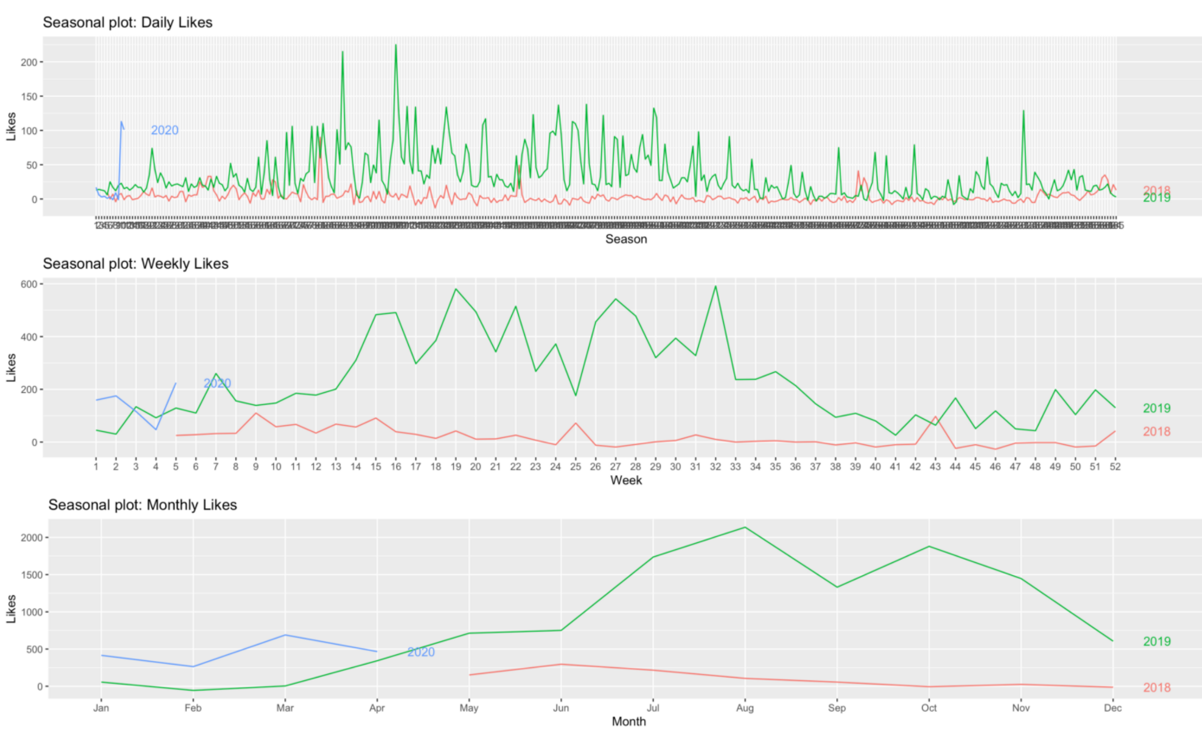
```
#Transform data into time series data

$like  = data$Total_Likes%>%
  ts(start=c(2018,4,27) ,frequency=365)
```

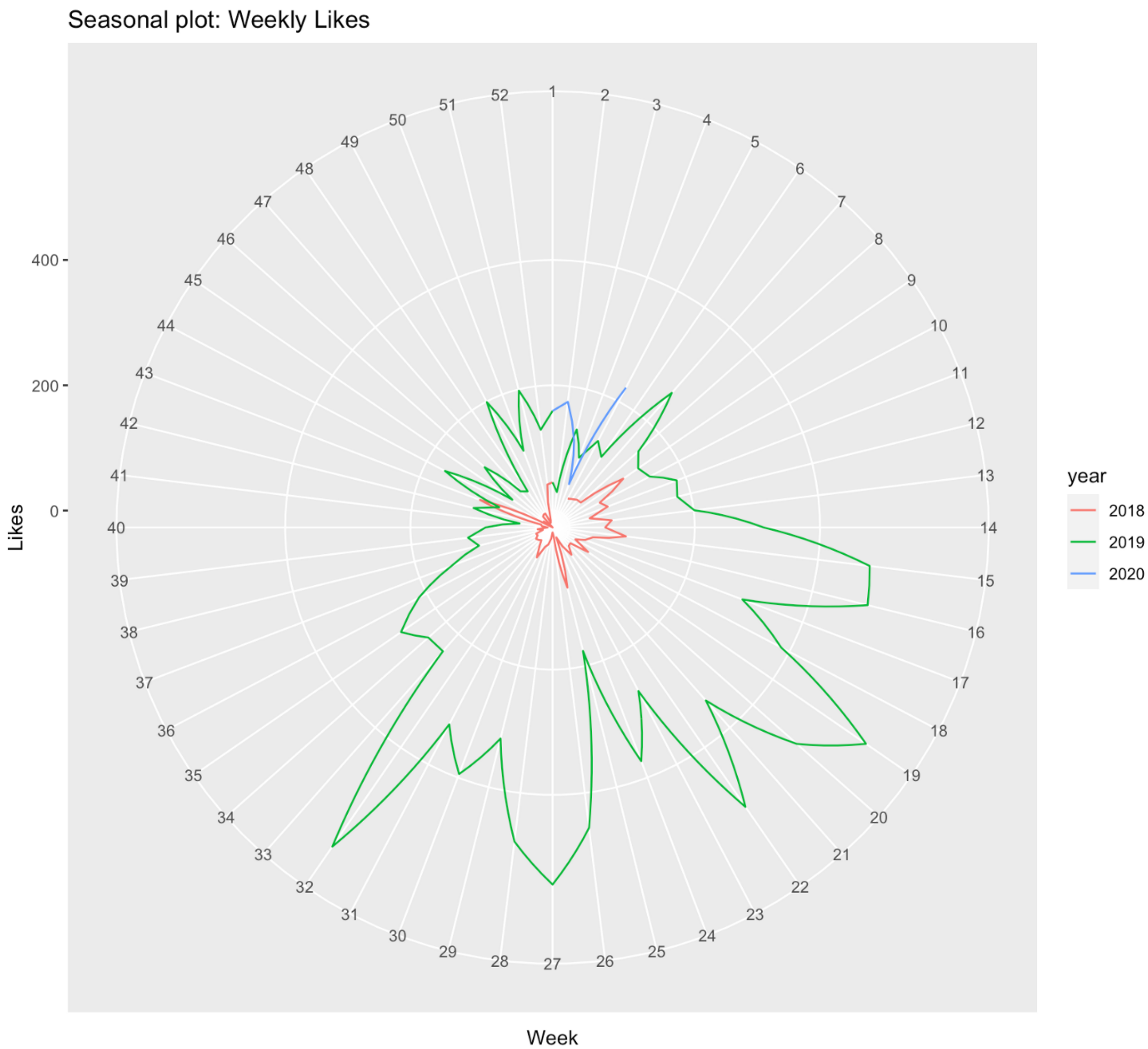
Data Exploratory Analysis

In a time series analysis, it's essential to observe and handle the **seasonal effect**, **trend,** and **stationarity(stability)** of the data.

First, I took a look at the **seasonality** of the increased number of Likes from 2018–2020, breaking down the data to daily, weekly, and monthly total Likes increase:



Daily, weekly, and monthly Likes

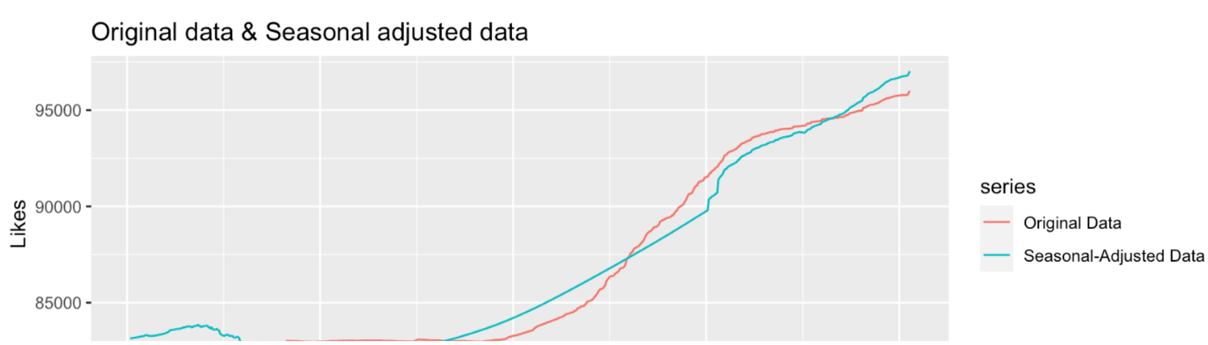


Polar seasonal plot of weekly increased Likes

Based on my 5-year experience with social media, the graphs indicate that there is no significant seasonal effect in the data.

We can reduce the seasonality by decomposing the data:

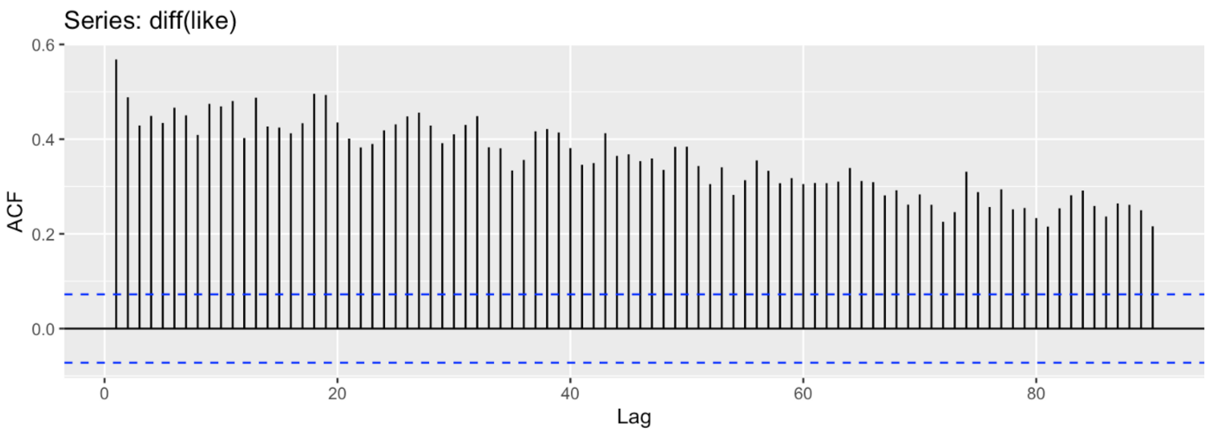
```
#Generate seasonal–adjusted data
seasonal_adjusted = decomposedlike$x–decomposedlike$seasonal
```



As this graph shows, there's no significant difference between the original data and the seasonally adjusted data.

Second, we can observe the **trend** by looking at Autocorrelation plot:

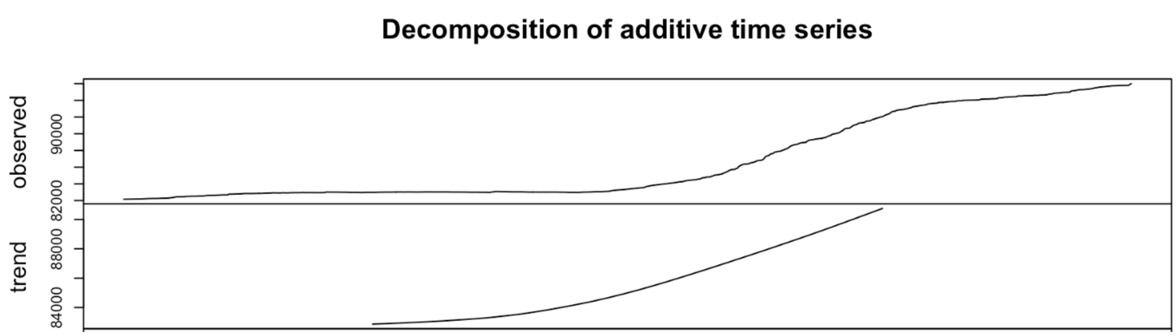
```
ggAcf(diff(like),lag =90)
```



Autocorrelation plot

The plot shows that the autocorrelations for small lags tend to be significant and positive; meanwhile, the values are decreasing slowly as the lags increase.
In short, **there's a trend in this data, and the seasonal lags are not significant.**

This also shows in the decomposition plot that there's a clear increasing trend:



Third, we use the Dickey-Fuller Test to examine the **stationarity**:

```
> adf.test(like, alternative = "stationary")
```

```
Augmented Dickey-Fuller Test

data:  like
Dickey-Fuller = -1.4016, Lag order = 9, p-value = 0.8316
alternative hypothesis: stationary
```

Given the high p-value, we fail to reject the null ➡ **data is non-stationary.**

Then I take a difference of the data, helping to stabilize the mean by removing changes in the level of a time series:

```
> adf.test(diff(like), alternative = "stationary")

Augmented Dickey-Fuller Test

data:  diff(like)
Dickey-Fuller = -3.4011, Lag order = 9, p-value = 0.05326
alternative hypothesis: stationary
```

The p-value is pretty close to what we want, but still, fail to reject the null.

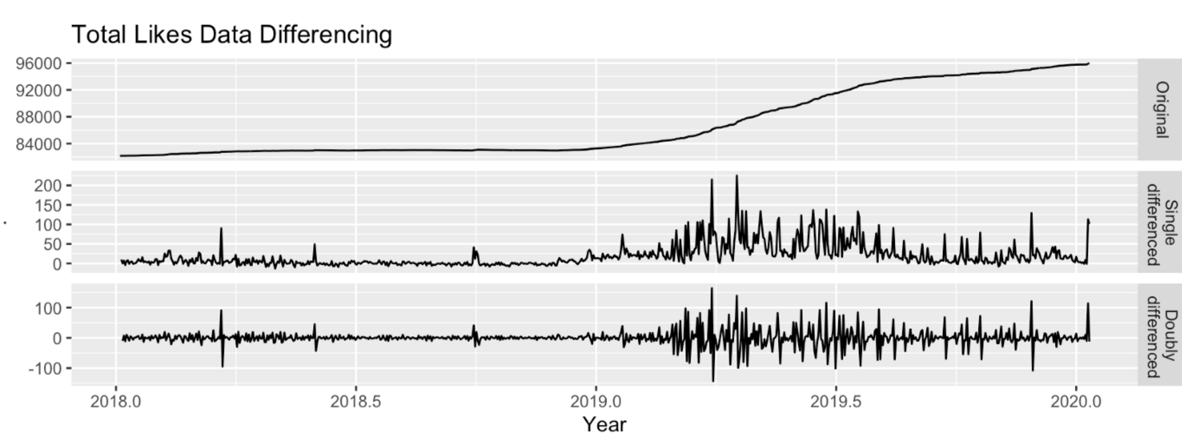
Then I take a second difference.

```
> adf.test(diff(diff(like)), alternative = "stationary")

Augmented Dickey-Fuller Test

data:  diff(diff(like))
Dickey-Fuller = -14.821, Lag order = 9, p-value = 0.01
alternative hypothesis: stationary
```

Given the low p-value, we reject the null➡ **data is stationary.** ✅



Data Differencing

After exploring and get a better understanding of the data:

1. Seasonality — not significant
2. Trend — a clear increasing trend
3. Stationarity — take doubly-differenced data to achieve stationary.

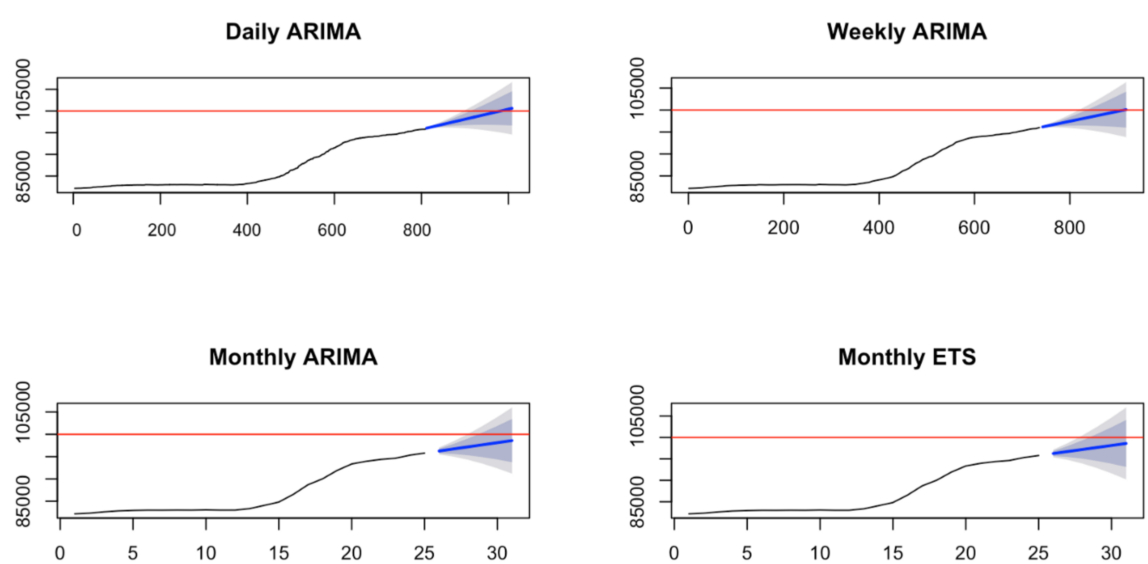
We are good to go for modeling!

Time Series Analysis

In the modeling process, I'm going to compare four different models:

1. Daily data with ARIMA model
2. Weekly data with ARIMA model
3. Monthly data with ARIMA model
4. Monthly data with ETS model(ETS has a better fit with monthly data)

Using the four models to do a half-year forecast, we can see only the Daily ARIMA and Weekly ARIMA predict that **my Fan Page will reach 100k in 6 months.**



Next, we want to test which models are more reliable. We will use the following "training and testing" method to evaluate the models.

Evaluating Forecast Accuracy in 3 steps

I split the data into 21 months as training data and three months as testing data to test the accuracy.

After we test the accuracy for the four models:

		ME	RMSE	MAE	MPE	MAPE
Daily	**ARIMA**	**3.55192**	**101.0099**	**87.395**	**0.0032841335**	**0.0918075**
Weekly	ARIMA	219.605	287.5174	228.14	0.229669962	0.2386927
Monthly	ARIMA	48.6667	150.7824	149.33	0.05037247	0.1567438
Monthly	ETS	122.588	219.0160	207.47	0.127768354	0.2174692

Result: **The Daily ARIMA model has the best performance**(accuracy).

Next, we'll dive deeper to see if the **Daily ARIMA model** is reliable. If it isn't, we'll then test the second-best model.

First, we want to see if the auto.ARIMA() function picks a good fit:

```
> fit
Series: like
ARIMA(0,2,3)

> test$ttable
    Estimate     SE  t.value p.value
ma1  -0.7280 0.0374 -19.4414  0.0000
```

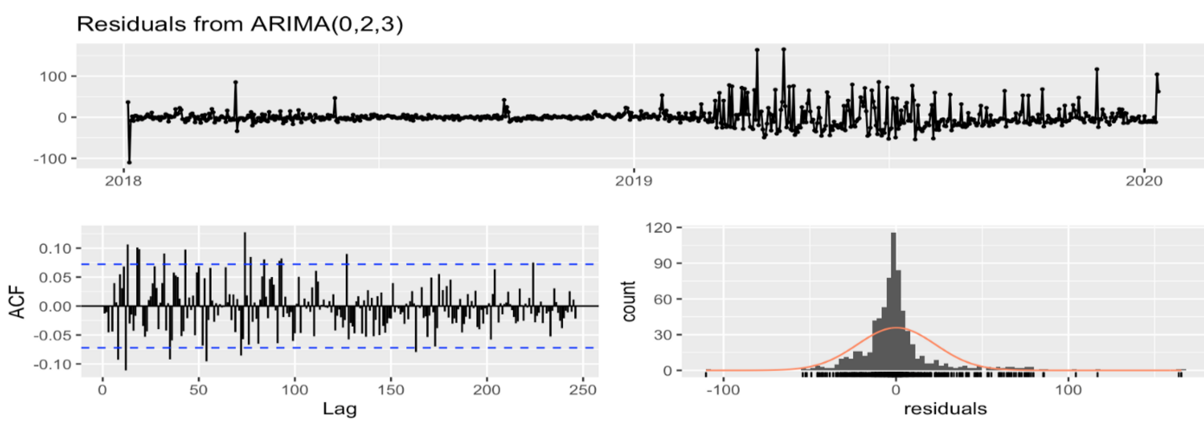
```
ma2   −0.1076 0.0434  −2.4808  0.0133
ma3   −0.0774 0.0370  −2.0945  0.0366
```

1. When picking the lowest AICc fit, it also shows doubly-differenced data is the best fit.✅

2. It shows the ma1, ma2, and ma3 are all good fit(low p-values).✅

Second, we need to check the **residuals**:

```
> fit %>% residuals()%>% mean()
[1] 0.2174377
#the mean of risiduals is close to zero✅
```



1. Residuals look like a white noise series.✅

2. 93.6% of the autocorrelations are between the blue lines, it’s a little less than 95% but is good enough.✅

3. The residuals look like a normal distribution.✅

```
qqnorm(residuals(fit)); qqline(residuals(fit))
```

The Q-Q plot shows it’s a reasonably good distribution.✅

Third, we can take a look at the difference between training and testing data:

With the three-month training and testing, the MAPE(mean absolute percentage error) about 9%. It’s accurate enough for me to set up data-driven goals!

Limitation

> How often should I update the data?

```
#Cross-validation
fc_function <- function(x, h){forecast(Arima(x, order=c(0,2,3)),
h=h)}
e <- tsCV(like_xts, forecastfunction=fc_function, h=180)
mse <- colMeans(e^2, na.rm = T)
data.frame(h = 1:180, MSE = mse) %>%
  ggplot(aes(x = h, y = MSE)) +
  geom_point()+
  ggtitle("Cross-Validation")
```

From this cross-validation graph, we can tell that the prediction within 60 days is reasonably accurate. Therefore, when using this model, it's better to update the data at least once a month.

Conclusions & Insights

> *We can finally let the old approach stay in the past, embracing the beauty of data analysis!*

> How can a social media influencer implement the analysis to practical uses?

Based on the prediction, my Fan Page will grow 950 Likes in May 2020 if I remain a performance similar to the past pattern. Therefore, when I am drawing up a plan for this month, I'll know if I want to put a lot of effort into this, growing 950 Likes is not enough. Instead, I should set up a higher goal, for instance, acquiring 1500 new Likes.

Also, this is excellent assistance for me to set up quarterly goals. I can build a six months timeline for me to follow. **Setting up unrealistic goals will never happen again!**

Before, setting up goals for social media accounts is highly rely on intuition and experience. Now, with the help of the analysis, we can finally let the old approach stay in the past, embracing the beauty of data analysis!

If you have any ideas or feedback you would like to share with me, feel free to send a message to me on Linkedin!

➡️ https://www.linkedin.com/in/kuanchengw/