Genshin Impact: Performance Forecast

Xueyan Hu

Central University of Finance and Economics (CUFE)
School of Finance

2022.12.28

Contents

1.1: Variables

There are six variables available in the dataset of Genshin Impact, which are Daily Active Users (DAU),Installs, and In-app Purchase (IAP) for users in Android and Apple respectively.
The data begins from 2020/2/1 and ends at 2021/1/31, daily.

1.1: Descriptive Statistics

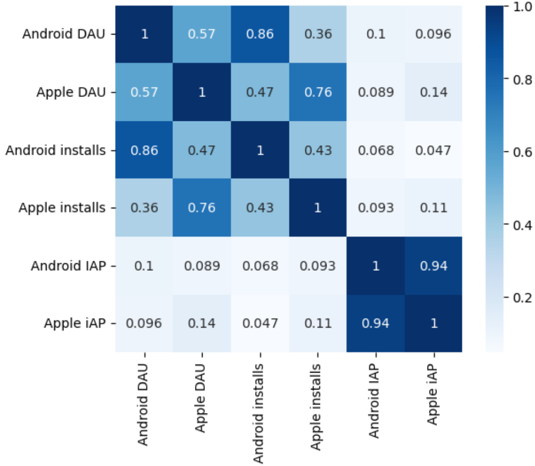
Descriptive statistics table is shown below. All the variables are Z-score standardized, hence the *Mean* and *Std* are omitted.

	Count	*Skew*	*Kurt*	*Min*	25%	50%	75%	*Max*
Android DAU	366.0	1.52	2.41	-1.44	-0.65	-0.27	0.35	3.76
Apple DAU	366.0	0.75	0.88	-1.53	-0.84	-0.03	0.64	4.57
Android installs	366.0	3.05	11.29	-0.97	-0.53	-0.26	0.13	5.88
Apple installs	366.0	3.71	30.66	-1.34	-0.62	-0.14	0.35	10.10
Android IAP	366.0	2.67	10.63	-1.12	-0.63	-0.27	0.31	6.42
Apple IAP	366.0	2.95	13.45	-1.13	-0.66	-0.26	0.33	6.70

1.1: Pearson Correlation Matrix

Implications:

- DAU and installs of same platform is strongly related;
- IAP seems somehow unrelated with DAU and installs.



1.2: Resampling: from Daily to Monthly

It's difficult to forecast in a high frequency, i.e. daily. We hence transform the data from daily into monthly.

- For DAU, we calculate the mean for the month;
- for installs and IAP, we calculate the sum for the month.

After resampling, we get 12 lines of data, from 2020/2 to 2021/1, inclusively.

2.1: ADF-test: Theory

Before building models, we should try to make the data a stationary process. Otherwise, the performance of time-series models is bad. Fortunately, Augmented Dickey-Fuller test (ADF-test) tests whether a data series is stable.
Take AR process as example. An AR(p) process can be written as equation (1).

$$y_t = c + a_1 y_{t-1} + ... + a_p y_{t-p} + \epsilon_t \tag{1}$$

Furthermore, equation (1) can be re-written into equation (2).

$$\Delta y_t = c + \rho y_{t-1} + \sum_{i=2}^{p} \phi_{i\,t-(i-1)} + \epsilon_t \tag{2}$$

And we test: $H_0 : \rho = 0 \quad H_1 : \rho < 0$.
If H_0 fails to hold, then the process is stationary.

2.1: ADF-test: Results

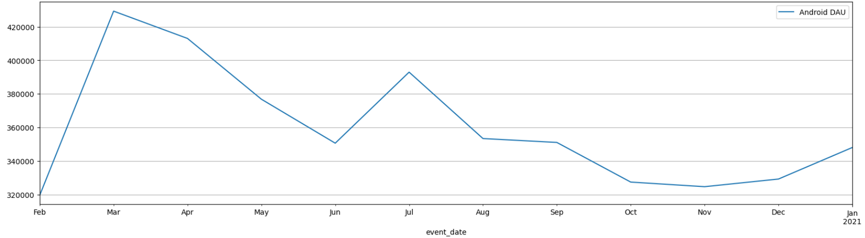
The ADF-test of variables are shown below.

Variable	ADF-statistics	p-value	Variable	ADF-statistics	p-value
Android DAU	-2.54	0.10	Apple DAU	-1.07	0.73
Android installs	-3.47	0.01	Apple installs	-2.88	0.05
Android IAP	-6.56	0.00	Apple IAP	-6.44	0.00

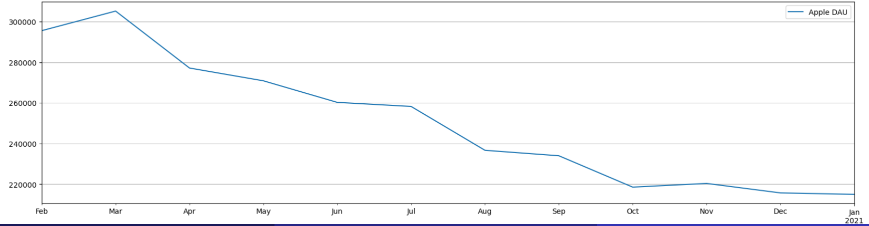
We notice that Android DAU and Apple DAU are not a stationary process (under a 10% significance).

The result of ADF-test is resonable.

See Android DAU line-chart.



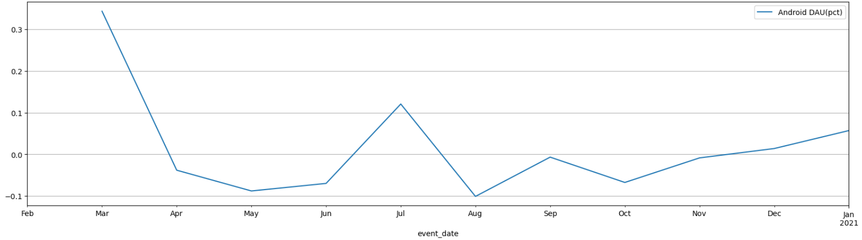
See Apple DAU line-chart.



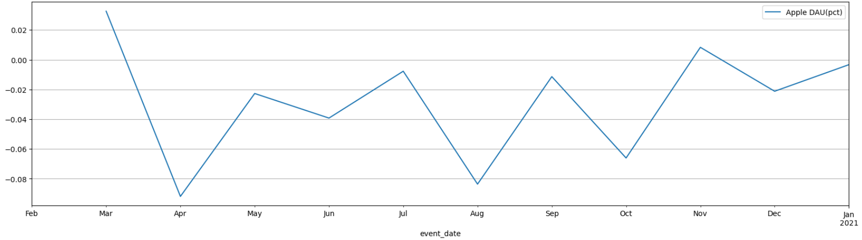
2.1: Differentiate DAU

- We calculate the percent change of DAUs.
- The ADF-statistics of the differences of DAUs are -6.30 (p-value=0.00) and -6.72 (p-value=0.00) for Android DAU and Apple DAU respectively.
- The differences pass the ADF-test, which means using the decay/growth rate to model is necessary.

See Android DAU(pct) line-chart.



See Apple DAU(pct) line-chart.



2.2 Forecasting with AR(p)

In this session, we take Android DAU as example to introduce how variables are predicted.
The rules is pretty simple here, in order to avoid black boxes.
We assume that the variable is linearly determined by it's previous observations, that is

$$Y_t = a_1 Y_{t-1} + a_2 Y_{t-2} + ... + a_p Y_{t-p} + c + \epsilon_t \tag{3}$$

where Y is the variable we focuses (Android DAU here), ϵ_t is the random shock at period t. Because of the ADF-test, we assure that ϵ is a white noise.
This assumption is basic and reasonable: people's behaviors are usually consistent. If people play for previous several days, they probably play for the next day.

2.2 Forecasting with AR(p)

- Because our sample is quite small, only 12 lines of data, so the number of lags, i.e. p shouldn't be too large.
- We consider the AIC criterion, BIC criterion and significance of each regressor comprehensively. In short, we believe AR(3) model fits the percent change of Android DAU.

2.2 Forecasting with AR(p)

Fitting the Android DAU(pct) with Newey-West adjustment, in a AR(3) model.
After fitting the model, we can use it to forecast. We use the percent change implement: $\prod_{i=t+1}^{t+k}(1+\hat{Y}_i)$ multiplied by the last available Android DAU.

	Dependent Variable
	Android DAU(pct)
const	-0.04 (-2.412)
AR(1)	-0.86 (-2.983)
AR(2)	-0.56 (-2.577)
AR(3)	-0.38 (-6.989)
AIC	-17.45
BIC	-17.06

2.2: Forecasting with AR(p)

Actually, it's simpler when forecasting installs and IAP because they don't require difference.

	Dependent Variable					
	Android DAU(pct)	Apple DAU(pct)	Android installs	Apple installs	Android IAP	Apple IAP
const	-0.04 (-2.412)	-0.05 (-4.87)	1.54×10^6 (6.44)	2.078×10^5 (4.51)	2.255×10^6 (6.36)	6.703×10^6 (4.66)
AR(1)	-0.86 (-2.983)	-0.57 (-7.35)	-0.21 (-1.79)	0.57 (5.82)	0.68 (3.72)	0.58 (6.64)
AR(2)	-0.56 (-2.577)		0.18 (5.03)		-0.50 (-3.15)	-0.42 (-2.37)
AR(3)	-0.38 (-6.989)		0.0028 (0.044)			-0.16 (-2.03)
AIC	-17.45	-39.24	258.90	279.65	285.27	275.10
BIC	-17.06	-38.33	259.89	278.90	286.48	276.09

2.3 Forecasting Results

Forecasting results are shown below.

	Android DAU	Apple DAU	Android_installs	Apple_installs	Android_IAP	Apple_IAP
2021-02-28	314304.13	204375.41	1459324.07	468633.04	2725699.93	6226565.92
2021-03-31	349070.76	210047.44	1556604.65	476956.96	2804281.05	6870181.70
2021-04-30	343461.30	206795.86	1497907.04	481738.60	2775799.98	7153577.02
2021-05-31	349218.70	208659.88	1526881.73	484485.39	2716991.41	6986026.43
2021-06-30	334194.86	207591.30	1510656.38	486063.27	2691630.27	6664903.72
2021-07-31	346027.30	208203.88	1519040.81	486969.68	2704127.04	6501609.23
2021-08-31	342065.08	207852.71	1514482.55	487490.37	2725338.18	6568060.28
2021-09-30	344564.10	208054.02	1516882.05	487789.47	2733365.20	6727445.87
2021-10-31	340134.97	207938.62	1515592.89	487961.29	2728101.48	6819174.74
2021-11-30	344052.04	208004.78	1516276.59	488059.99	2720504.58	6795141.69
2021-12-31	342210.59	207966.85	1515911.00	488116.69	2718026.51	6716781.79
2022-01-31	343286.64	207988.59	1516105.47	488149.26	2720179.69	6666205.76

If we have the true value, we can calculate statistics such as MSFE, DW to show how good our forecasts are.
If we have data of longer periods, we can use in-sample to train the AR and test how good it performs, and do out-of-sample forecast individually.

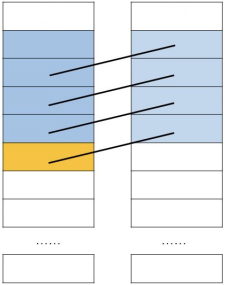
Auto-regressive distributed lag (ARDL) model is a transform of AR by including variables other than dependent variables.
Simplified ARDL can be written as:

$$Y_t = \sum_{p=1}^{p} \phi_p Y_{t-p} + \sum_{l=1}^{k} \sum_{j=1}^{j} \beta_{l,j} X_{l,t-j} + \epsilon_t \tag{4}$$

In this session, Y represents DAU, and Xs are installs and IAP.

3.2: ARDL Fitting

Before modeling, we notice the Pearson Correlation Matrix showed ahead. The relationship between DAU and IAP is quite week, implying the DAU is driven by lags of installs and itself.
We fit equation for Android DAU(pct) with value one period afterwards, i.e. predictive regression.



3.3: ARDL Result

- fortunately, the result is satisfying.
- Although the coefficient of variables other than the dependent variable is extremely low (in 10^-8 scale), it's due to the scale of the data. We should focus on significance level.

	Dependent Variable	
	Android DAU(pct)	Apple DAU(pct)
const	0.55 (4.29)	0.050 (0.77)
AR(1)	0.92 (5.90)	-0.356 (-2.35)
installs_{t-1}	-3.21×10^{-7} (-6.75)	-2.4×10^{-7} (-4.58)
installs_{t-2}	7.212×10^{-8} (3.56)	
IAP_{t-1}	7.212×10^{-8} (3.56)	-4.61×10^{-9} (-0.96)
AIC	-22.69	-42.48
BIC	-20.87	-40.97

References

Newey, W. K. and West, K. D. (1987).
Hypothesis testing with efficient method of moments estimation.
International Economic Review, pages 777–787.

3.3: ARDL Result

The DAU forecast are shown below.

	Android DAU	Apple DAU
2021-02-28	401307.97	196979.44
2021-03-31	266185.11	204502.07
2021-04-30	253933.88	207728.2
2021-05-31	265908.91	204724.55
2021-06-30	300174.86	207039.82
2021-07-31	241366.62	204232.93
2021-08-31	294655.83	211377.7
2021-09-30	297820.85	207065.16
2021-10-31	314064.31	216344.98
2021-11-30	326374.86	209193.03
2021-12-31	347474.55	210301.95
2022-01-31	334952.91	209624.07

Thank You!