Asking Python Questions on Stack Overflow

Time Series Analysis and Forecasting

Stella Jia

3 June 2022

Contents

Abstract

This report presents a time series analysis of Python-related questions on Stack Overflow, aiming to investigate the growth patterns and potential forecasting of Python's popularity as a programming language. By applying the Box-Jenkins methodology, we conduct a comprehensive time series analysis and utilize the obtained insights to forecast the Python question trends for 2019. While the actual data lies outside our predicted interval, this outcome highlights the nonlinearity and unpredictability inherent in programming languages and their association with evolving technological paradigms.

Introduction

With AI development rising in the past decade, the most popular programming language individuals use to explore and build in this area is Python. In this project, I wanted to explore how the questions around Python have grown over time given it's popularity throughout different technology paradigms. Specifically, is there a pattern to Python questions being asked and can we **forecast** it?

The dataset I've chosen is from Kaggle and records the number of Stack Overflow questions under the Python tag from 2009 to 2019. Using the Box-Jenkins methodology, we will perform a time series analysis on the dataset and forecast the year of 2019 based on the years before. Although the true data was not within our forecasted prediction interval, this reveals the unpredictable and nonlinear nature of programming languages and tracking technological paradigms from object-oriented programming to generative AI. In the future, it would be fruitful to explore nonlinear time series models.

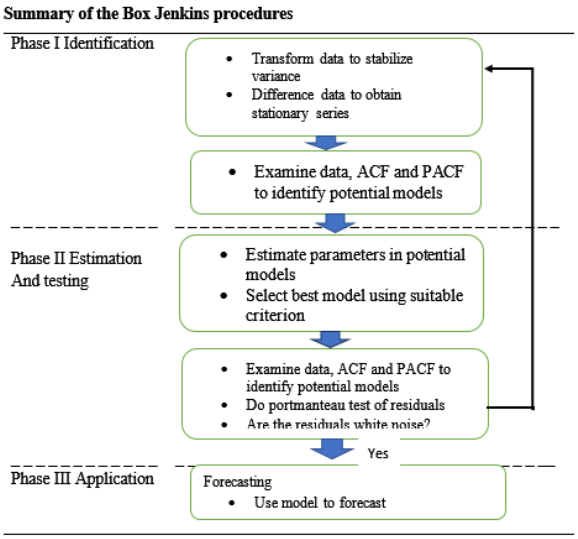


Figure 1: Box-Jenkins Method

Data Preprocessing

Cleaning

First, the data has been split into a training (up until 2019) and testing (2019 itself). We will use the testing set to assess the accuracy of our forecast. In Figure 2 Stage 2, the data before 2012 appears to be inconsistent with the rest of the data. Thus, data before 2010 will be removed to ensure accurate forecasting while still

having enough data to forecast. We will be moving forward with the data presented in Stage 3. Further characteristics of our final data can be seen in Figure 3.

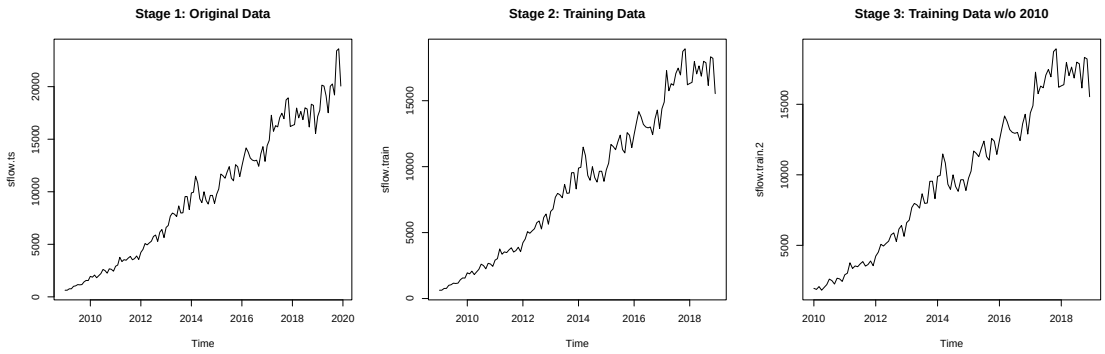


Figure 2: Stages of Cleaning Data

Based on the following observations, it is clear that our data is **not stationary.**

(i) The data has a clear **linear trend** as shown by the red line on the time series plot.
(ii) There may also be some **light seasonality** as seen in the subtle periodic behavior in the time series plot.

(iii) After removing the data before 2010, the apparent changes throughout the data has lessened however the data still **does not have a constant variance.**
(iv) The ACF plot decays very slowly which means future values are heavily correlated with the past and thus, our data **does not have a constant mean.**

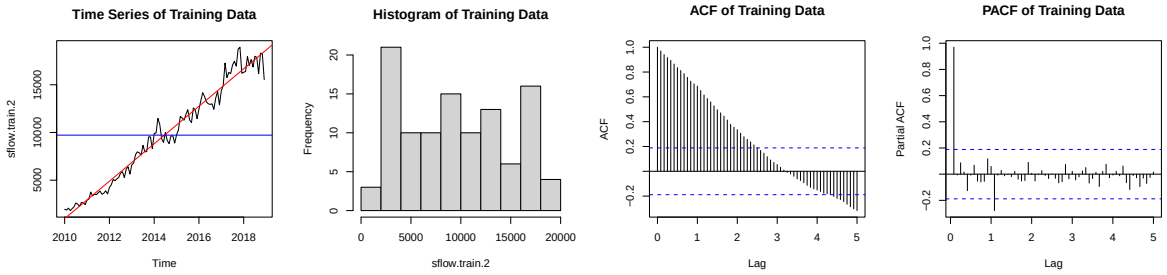


Figure 3: Characteristics of Training Data

Transforming

Given that our data is not stationary, we must apply transformations to make it stationary. First, lets try a Box-Cox transformation. Since our 95% confidence interval for lambda does not include zero, we can conclude that a log transform wouldn't be significant.

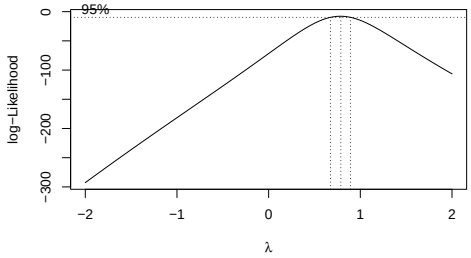


Figure 4: Log-Likelihood of Lambda for Box-Cox

Based on Figure 5, log transform appears to have an exponential trend and the histogram is skewed to the left making it a poor option. While the data for Box-Cox isn't completely Gaussian, there is some symmetry which makes it closer to Gaussian than the log transform. Given this advantage, we will proceed with the Box-Cox transformation.

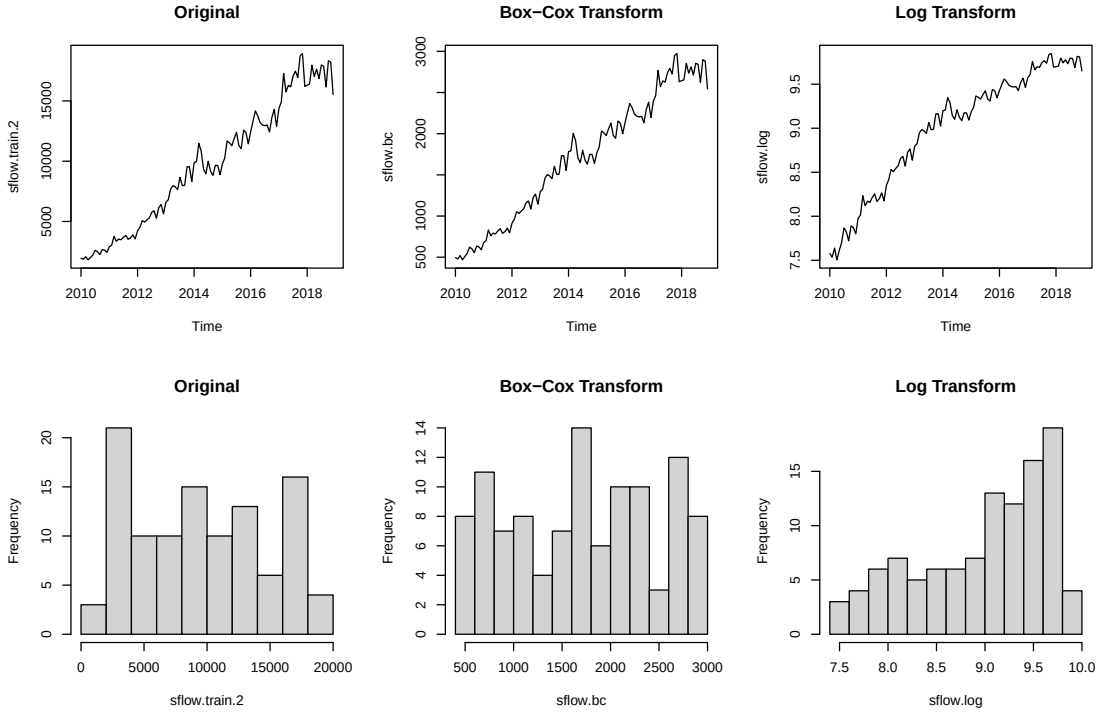


Figure 5: Characteristics of Different Transformations

Differencing

The final step in making our data stationary is differencing. Based on the time series plot of our Box-Cox transform in Figure 5, it's clear that we have a linear trend. Thus, we can begin by differencing at lag 1. After differencing at lag 1 we have the following plots:

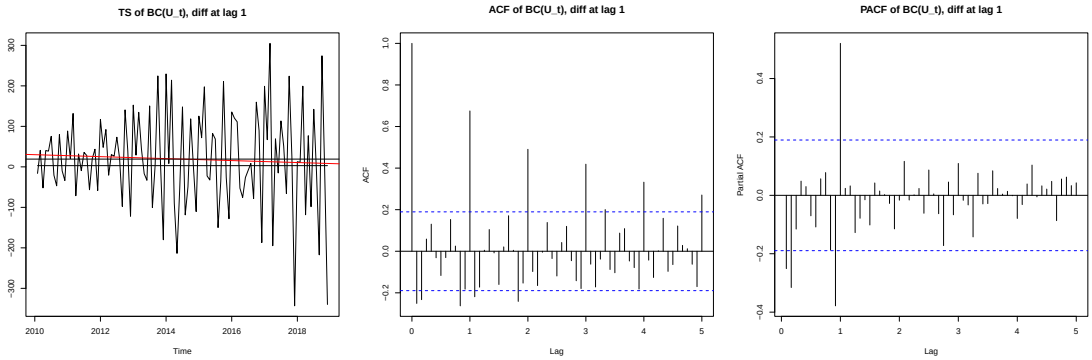


Figure 6: Difference at Lag 1

From Figure 6, we can see that the trend is mostly removed where the red line of the time series plot is closer to a horizontal line (no trend) rather than a diagnoal line (indicating trend). However, the ACF now displays significance at a seasonality of 12 so we will difference once at lag 12 to remove seasonality. After differencing at lag 12 we have these plots:

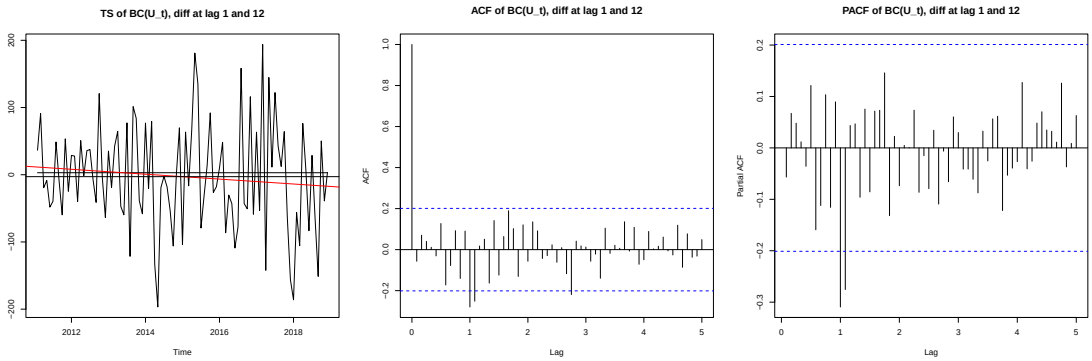


Figure 7: Difference at Lag 12

The red line of the trend seems to have gotten steeper however the variance appears to be a bit more stable. The ACF plot decay now resembles a stationary process and has no seasonality. Before finalizing our model, lets look at the variance of all these differences:

difference	variance
No Difference	574990.568
De-trended	13865.344
Seasonally Differenced + De-trended	6279.487
Seasonally Differenced + Den-trended twice	13397.245

Based on the table, the difference with the lowest variance is the seasonally differenced + de-trended ($\nabla_1\nabla_{12}\text{bc}(U_t)$ where U_t is our original data from 2010 to 2019). Since this model has the lowest variance, and appears stationary based on time series and P/ACF plots, we will move forward with this model. When comparing the histogram of our original data in Figure 7 with the differenced data, it also looks more Gaussian which is advantageous for the Box Jenkins method.

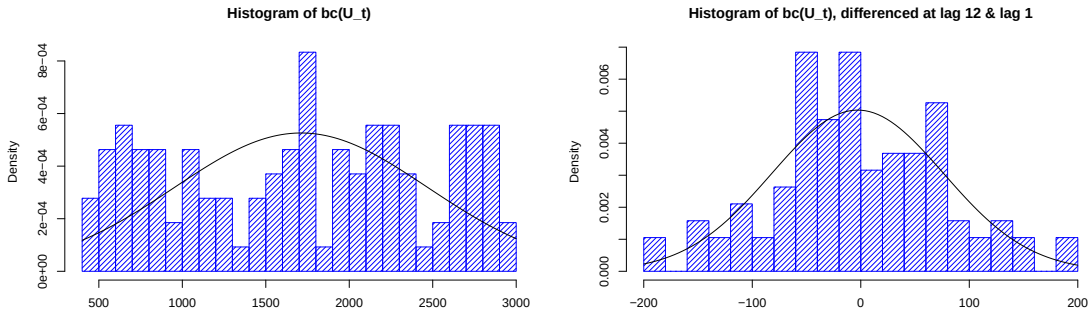


Figure 8: Comparing Histogram of Transformed Data w/ Tranformed + Differenced Data

Model Identification and Estimation

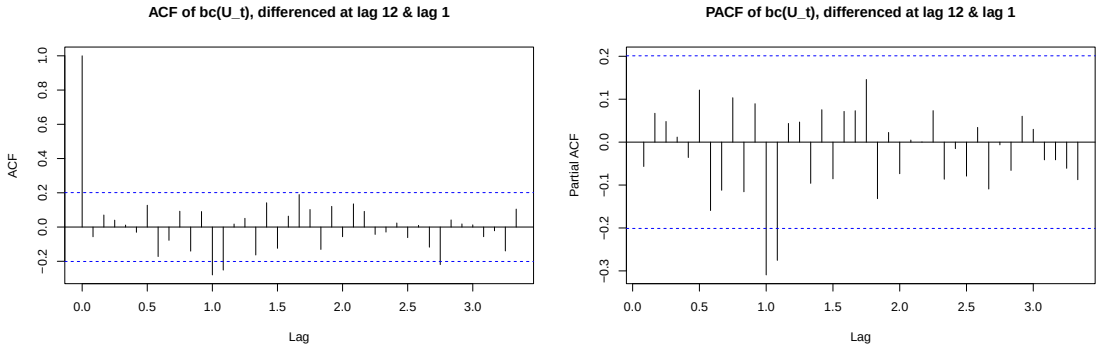


Figure 9: ACF and PACF of Transformed Data

The ACF and PACF both show a significant spike at lag 12 and 13. All lags k>13 in ACF and PACF appear to be insignificant. ACF at lag 33 goes over the CI, but we can disregard that because R underestimates Bartlett's formula. Given that behavior, here are some potential models:

Note: Table identifies what the significant coefficients are in the third column. All coefficient and standard error values can be found in the Appendix.

Model	AICc_Value	Significant_Coefficients
A. SMA(1)_12	1092.392	SMA1
B. SAR(1)_12	1095.171	SAR1
C. SARIMA(0,1,0)x(1,1,1)_12	1093.737	SMA1
D. SARIMA(0,1,1)x(0,1,1)_12	1093.069	SMA1
E. SARIMA(0,1,1)x(0,1,2)_12	1094.896	SMA1
F. MA(12)	1105.787	MA12
G. AR(12)	1109.074	AR12

Based on our summary table above, the models with the lowest AICc values are model A, D, C, and E. Models A-C in the table were determined based on the ACF and PACF plots of our stationary data. Models D and E were chosen after experimenting with diagnostic checking. Models F and G were chosen to show how straying away from the rule of parsimony will increase the AICc drastically.

The models C, D, and E suggest that adding any component aside from SMA1 is insignificant. Although it appears that SMA1 is the only significant coefficient, it still may be useful to see how the other models perform in diagnostic checking. If we set all insignificant coefficients to 0, we would end up with multiple SMA1 models. Thus, I chose to move forward with model A, D, and E while keeping the other coefficients.

Model A: $\nabla_1\nabla_{12}bc(U_t) = (1 - 0.4149_{0.1198}B^{12})Z_t, \quad \hat{\sigma}_Z^2 = 5397$

Model D: $\nabla_1\nabla_{12}bc(U_t) = (1 - 0.1231_{0.1006}B)(1 - 0.4407_{0.1194}B^{12})Z_t, \quad \hat{\sigma}_Z^2 = 5296$

Model E: $\nabla_1\nabla_{12}bc(U_t) = (1 - 0.1055_{0.1056}B)(1 - 0.4232_{0.1181}B^{12} - 0.0757_{0.1284}B^{24})Z_t, \quad \hat{\sigma}_Z^2 = 5296$

To properly forecast, we need to prove that our models are stationary and invertible. All of our models only contain moving average components so it is stationary by default. Model A is invertible because $|\Theta_1| = 0.4149 < 1$ and there is only one order. Model D is invertible because $|\theta_1| = 0.1231 < 1$ and $|\Theta_1| = 0.4407 < 1$. Model E is invertible because $|\theta_1| = 0.106 < 1$ and all seasonal roots are outside the unit circle.

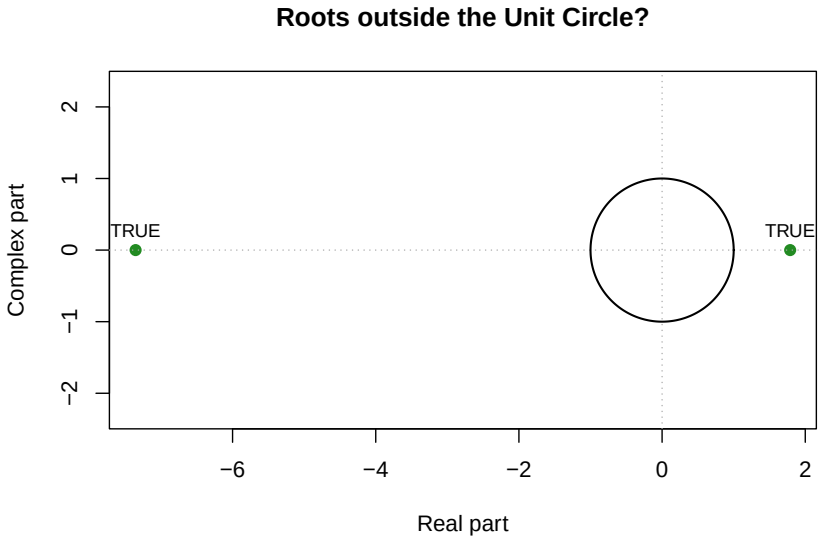


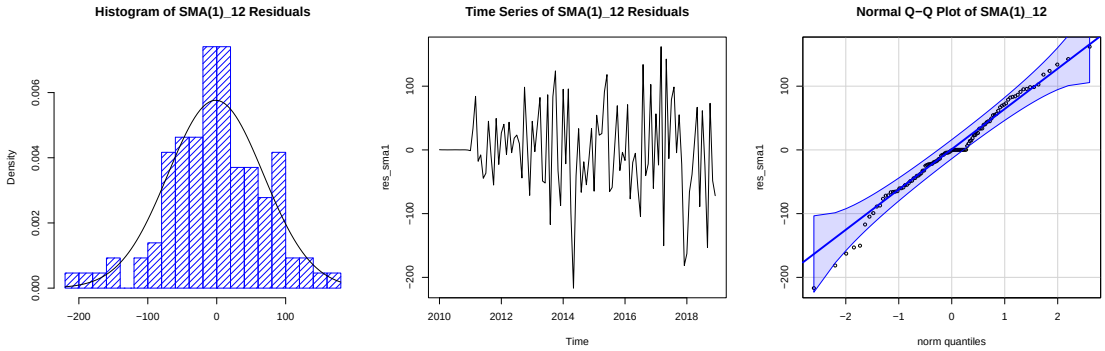
Figure 10: Model E Roots of MA Part, Seasonal

Diagnostic Checking

In addition to checking roots, we also want to perform a diagnostic check on the residuals to ensure it is ready for forecasting. An appropriate model should have resiudals resembling normality, and pass all the diagnostic tests (no linear and no nonlinear dependence). To perform diagnostic tests we will need to calculate the lag and degrees of freedom (fitdf) for each model.

Model A: SMA(1)_12

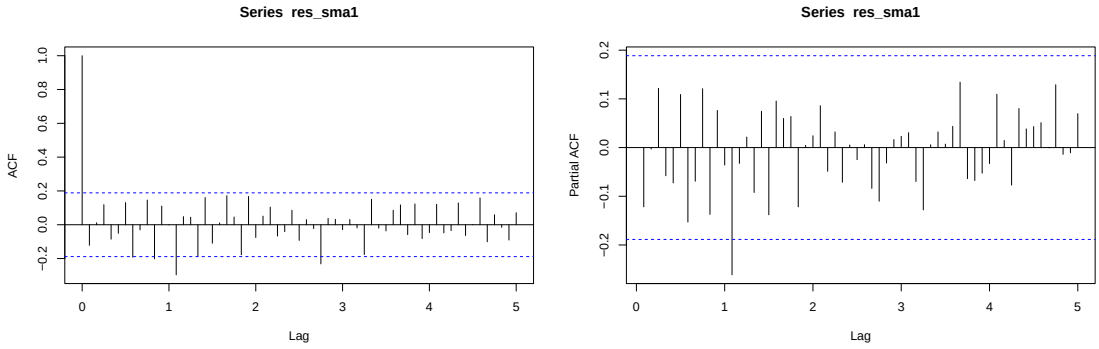
Although the normal QQ plot displays that $SMA(1)_{12}$ deviates from normality (heavy tails), it passes the Shapiro-Wilk normality test so we can conclude that it is normal.



We have a lag of $\sqrt{n} = \sqrt{120} \approx 11$ and fitdf of 1. This model passes Shapiro-Wilk and Box-Pierce test but fails both Box-Ljung tests. We can conclude that there is linear and nonlinear dependence which makes $SMA(1)_{12}$ model not ideal.

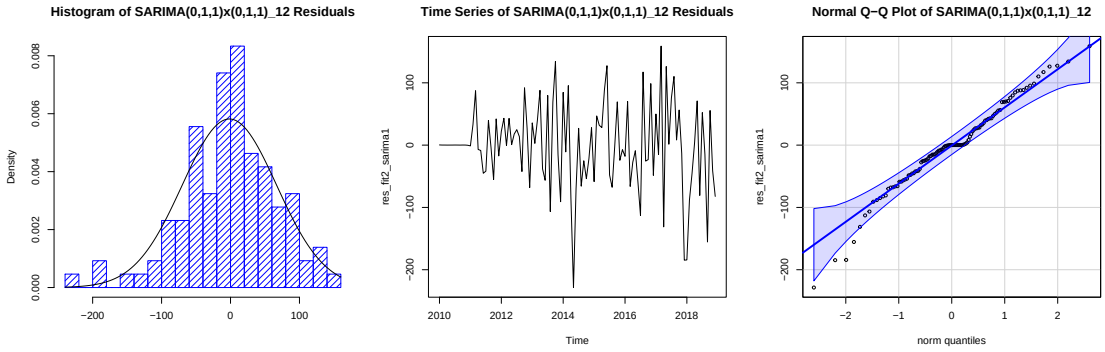
```
##
##  Shapiro-Wilk normality test
##
## data:  res_sma1
## W = 0.98246, p-value = 0.1664

##
##  Box-Pierce test
##
## data:  res_sma1
## X-squared = 18.148, df = 10, p-value = 0.05251

##
##  Box-Ljung test
##
## data:  res_sma1
## X-squared = 19.811, df = 10, p-value = 0.0311

##
##  Box-Ljung test
##
## data:  (res_sma1)^2
## X-squared = 46.388, df = 11, p-value = 2.761e-06
```

The ACF and PACF of $SMA(1)_{12}$ show significance at lag 13. This observation led to the formation of model D where we tried increasing the non-seasonal part to q=1: $SARIMA(0,1,1) \times (0,1,1)_{12}$.

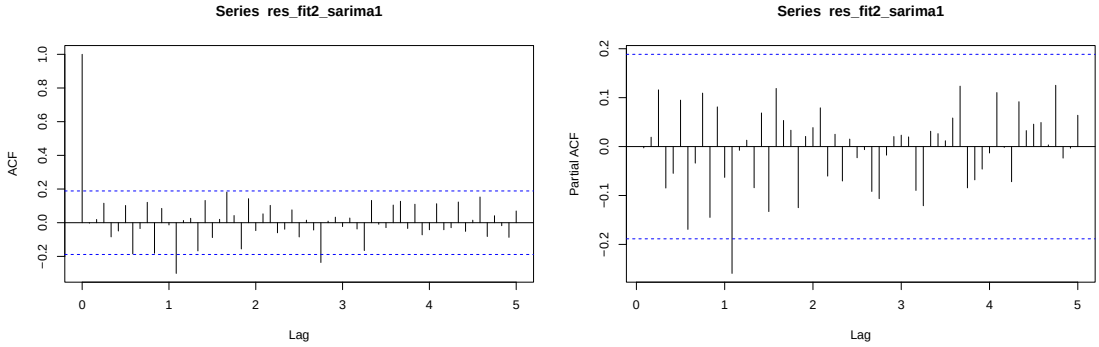


Model D: SARIMA(0,1,1)x(0,1,1)_12

Based on the figure below, Model D seems to have some heavy tails which deviates from normality.



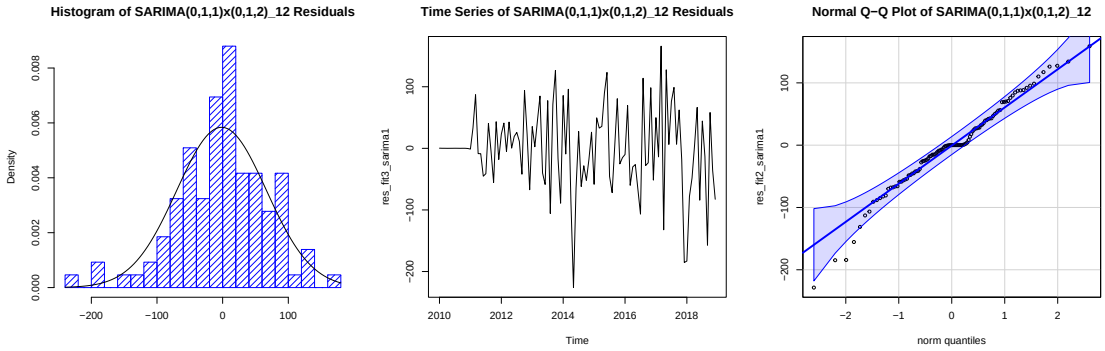
Increasing to q=1 still results in signifiance at lag 13 in the ACF and PACF of our residuals. Due to this abnormal behavior, I originally tested models that set p=1 or P=1 however both still had signifance at lag 13 and having an AR component seemed to greatly increase the AICc value. Thus, I stuck with altering MA components which led me to the formulation of Model E setting Q=2.



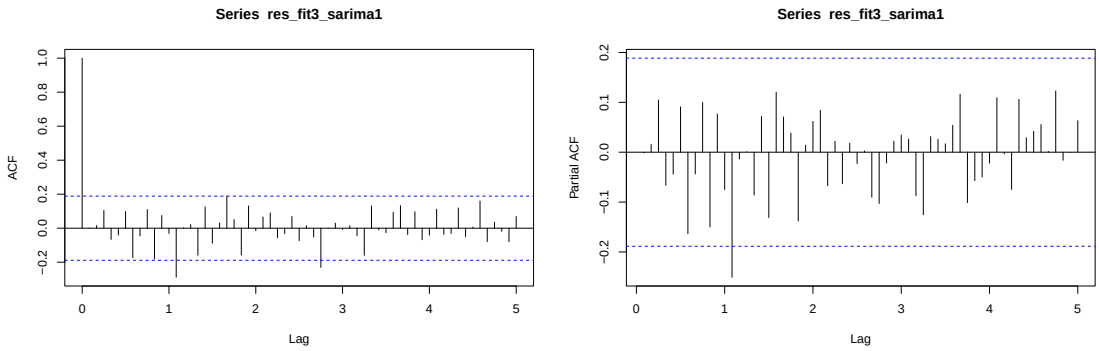
For testing, we still have lag as 11 but now fitdf is 2. Model D fails the Shapiro-Wilk test, and McLeod-Li test. Thus, residuals do not have normality and exhibits nonlinear dependence. Failure of both these tests makes it a not ideal option for modeling.

```
##
##  Shapiro-Wilk normality test
##
## data:  res_fit2_sarima1
## W = 0.97575, p-value = 0.04553

##
##  Box-Pierce test
##
## data:  res_fit2_sarima1
## X-squared = 13.394, df = 9, p-value = 0.1456

##
##  Box-Ljung test
##
## data:  res_fit2_sarima1
## X-squared = 14.673, df = 9, p-value = 0.1003

##
##  Box-Ljung test
##
## data:  (res_fit2_sarima1)^2
## X-squared = 32.801, df = 11, p-value = 0.0005667
```

Model E: SARIMA(0,1,1)x(0,1,2)_12



While our ACF and PACF of residuals indicate significance at lag 13, I have attempted to address the issue by exploring multiple different models and the significance at lag 13 still remains. This is an indication that SARIMA may not be the best method for modeling our data however it can still be a beneficial tool to understand our data.



For testing, we still have lag as 11 but now fitdf is 3. Model E passes all tests but the McLeod-Li test. Since Model E only fails the McLeod-Li test, we can conclude there is nonlinear dependence. Thus, our data is not necessarily suitable for SARIMA model and would likely need to look into a nonlinear model for more accuracy.

```
##
##  Shapiro-Wilk normality test
##
## data:  res_fit3_sarima1
## W = 0.97687, p-value = 0.05648

##
##  Box-Pierce test
##
## data:  res_fit3_sarima1
## X-squared = 11.795, df = 8, p-value = 0.1606

##
##  Box-Ljung test
##
## data:  res_fit3_sarima1
## X-squared = 12.941, df = 8, p-value = 0.1139

##
##  Box-Ljung test
##
## data:  (res_fit3_sarima1)^2
## X-squared = 33.667, df = 11, p-value = 0.0004099
```

Forecasting

Model E passes the most tests while Model A has the lowest AICc. Given that the AICc difference among the three models is minimal, we will value passing residual tests over the low AICc value. In this case, **Model E passes the most tests so we will choose it as our final model.** While the model is not completely satisfactory according to our residual analysis, it is the best option given our limitations to SARIMA models.

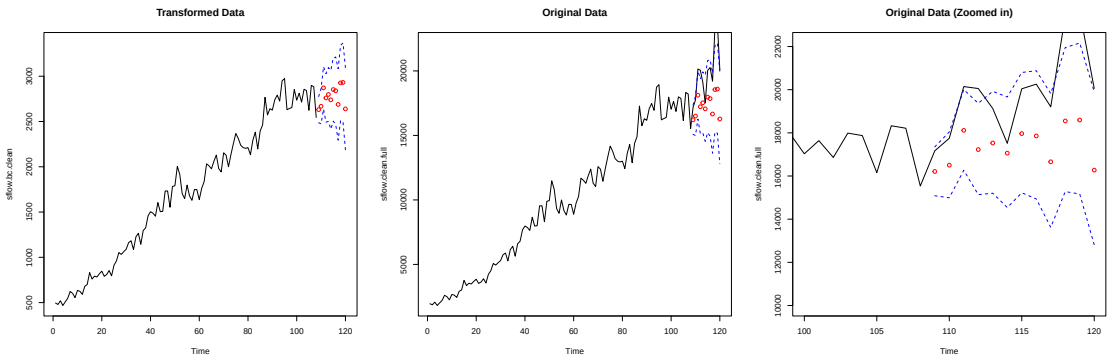


Figure 11: Forecasting on Transformed and Original Data with Model E

Unfortunately, our test set is not within the prediction interval (in blue). Although the test set isn't in the P.I., the nature of the forecasted data matches the staggering pattern of our original dataset. As noted in our observations of nonlinear dependence, it is not surprising to see that Model E was not perfect in forecasting our data.

Conclusions

Through a tradeoff of AICc value versus residual analysis tests, we selected Model E as our final model to forecast the number of Python questions asked.

$$\text{Model E:} \quad \nabla_1\nabla_{12}bc(U_t) = (1 - 0.1055_{0.1056}B)(1 - 0.4232_{0.1181}B^{12} - 0.0757_{0.1284}B^{24})Z_t, \quad \hat{\sigma}_Z^2 = 5296$$

Model E ended up not being a suitable model for forecasting since our test set was outside the prediction interval. To more accurately forecast this problem, it would be useful to look into nonlinear models for time series. While the goal of this project to *accurately* forecast Python questions was not achieved, this result still captured the staggering nature of this dataset. With the growth in AI development, versatile programming language like Python will likely continue seeing an increase in popularity and different technological paradigms could facilitate it. This development will likely be unpredictable and beyond the scope of SARIMA models so more complex time series analysis would be beneficial.

Acknowledgements

Professor Raisa Feldman

TA Thiha Aung

TA Lihao Xiao

Student Jessie Zhou

Student Rebecca Chang

References

Data obtained from Kaggle: https://www.kaggle.com/datasets/aishu200023/stackindex/code

PSTAT 174 Lecture Slides

Appendix

All code used to create the figures and tables above are included below.

```
# LIBRARIES INSTALLED

library(kableExtra)
library(knitr)
library(tidyverse)
library(MASS)
library(MuMIn)
library(UnitCircle)
library(ggplot2)
library(lemon)
library(forecast)
library(car)

# Import data
sflow <- read.csv('stack-overflow.csv')
```

```
# CREATED FUNCTIONS

# outputs 3 visuals to check stationarity
check_stationary <- function(ts, string, lag_max=60) {
  # par(mfrow=c(2,2))
  par(mfrow=c(1,3))
  # Plot time series
  ts.plot(ts, main=paste("TS of", string), cex=3)

  # Add trend and mean line
  fit <- lm(ts ~ time(ts))
  abline(fit, col="red")
  abline(h=mean(ts), col="black")
  # Decomposition
  # plot(decompose(ts))

  # Plot ACF and PACF
  #op2= par(mfrow=c(1,2))
  acf(ts, lag.max=lag_max, main=paste("ACF of", string), cex=3)
  pacf(ts, lag.max=lag_max, main=paste("PACF of", string), cex=3)
}

# outputs histogram compared with normal dist
plot_hist <- function(data, plot_title) {
  hist(data,density=20,breaks=20, col="blue", xlab="", prob=TRUE, main=paste(plot_title))
  m <- mean(data)
  std <- sqrt(var(data))
  curve(dnorm(x,m,std), add=TRUE)
}

# reverses box cox transformation
reverse_bc <- function(data, lambda) {
  (data*lambda+1)^(1/lambda)
}
```

```r
# DATA CLEANING

# Edit data types
sflow$month <- as.Date(paste0("01-", sflow$month), format = "%d-%y-%b")
# Select only Python
sflow.ts <- ts(sflow$python, start=c(2009,1,1), frequency=12)
sflow.ts.cut <- ts(sflow.ts[13:132], start=c(2010,1,1), frequency=12)
# Split intro train vs test
sflow.train <-ts(sflow.ts[1:120], start=c(2009,1,1), frequency=12)
sflow.test <- ts(sflow.ts[121:132], start=c(2019,1,1), frequency=12)
# Cut off before 2010
sflow.train.2 <- ts(sflow.train[13:120], start=c(2010,1,1), frequency=12) # n=108
```

```r
# FIGURE 2
par(mfrow=c(1,3))
ts.plot(sflow.ts, main="Stage 1: Original Data")
ts.plot(sflow.train, main="Stage 2: Training Data")
ts.plot(sflow.train.2, main="Stage 3: Training Data w/o 2010")

# FIGURE 3
par(mfrow=c(1,4))
ts.plot(sflow.train.2, main="Time Series of Training Data")
# Add trend and mean line
fit <- lm(sflow.train.2 ~ time(sflow.train.2))
abline(fit, col="red")
abline(h=mean(sflow.train.2), col="blue")
# Plot histogram
hist(sflow.train.2, main="Histogram of Training Data")
# Plot acf and pacf
acf(sflow.train.2, main="ACF of Training Data", lag.max=60)
pacf(sflow.train.2, main="PACF of Training Data", lag.max=60)

# FIGURE 4
t = 1:length(sflow.train.2)
fit = lm(sflow.train.2 ~ t)
bcTransform = boxcox(sflow.train.2 ~ t, plotit = TRUE) # 0 is not in the CI
lambda = bcTransform$x[which(bcTransform$y == max(bcTransform$y))]

sflow.bc <- (1/lambda)*(sflow.train.2^lambda-1)
sflow.log <- log(sflow.train.2)

# FIGURE 5
par(mfrow=c(2,3))
ts.plot(sflow.train.2, main="Original")
ts.plot(sflow.bc, main="Box-Cox Transform")
ts.plot(sflow.log, main="Log Transform")

hist(sflow.train.2, main="Original", breaks=10)
hist(sflow.bc, main="Box-Cox Transform", breaks=10)
hist(sflow.log, main="Log Transform", breaks=10)

# FIGURE 6
sflow.bc1 <- diff(sflow.bc, difference=1, lag=1)
```

```r
check_stationary(sflow.bc1, "BC(U_t), diff at lag 1")

# FIGURE 7
sflow.bc.1.12 <- diff(sflow.bc1, difference=1, lag=12)
check_stationary(sflow.bc.1.12, "BC(U_t), diff at lag 1 and 12")
sflow.bc.1.12.1 <- diff(sflow.bc.1.12, difference=1, lag=1)

# Table of Differences
difference <- c("No Difference","De-trended", "Seasonally Differenced + De-trended", "Seasonally Differ
variance <- c(var(sflow.bc), var(sflow.bc1), var(sflow.bc.1.12), var(sflow.bc.1.12.1))
vars_df <- data.frame(difference, variance)
kable(vars_df) %>% kable_styling(position = "center") # best one is seasonally differenced and de-trend

# FIGURE 8
par(mfrow=c(1,2))
plot_hist(sflow.bc, "Histogram of bc(U_t)")
plot_hist(sflow.bc.1.12, "Histogram of bc(U_t), differenced at lag 12 & lag 1")

# FIGURE 9
par(mfrow=c(1,2))
acf(sflow.bc.1.12, lag.max = 40, main="ACF of bc(U_t), differenced at lag 12 & lag 1")
pacf(sflow.bc.1.12, lag.max=40, main="PACF of bc(U_t), differenced at lag 12 & lag 1")
```

```r
# Fit models
## SMA1
fit1_sma1 <- arima(sflow.bc, order=c(0,1,0), seasonal=list(order=c(0,1,1), period=12), method = "ML")
## SAR1
fit1_sar1 <- arima(sflow.bc, order=c(0,1,0), seasonal=list(order=c(1,1,0), period=12), method = "ML")
## SARIMA(0,1,0)x(1,1,1)_12
fit1_sarima1 <- arima(sflow.bc, order=c(0,1,0), seasonal=list(order=c(1,1,1), period=12), method = "ML"
## SARIMA(0,1,1)x(0,1,1)_12
fit2_sarima1 <- arima(sflow.bc, order=c(0,1,1), seasonal=list(order=c(0,1,1), period=12), method = "ML"
## SARIMA(0,1,1)x(0,1,2)_12
fit3_sarima1 <- arima(sflow.bc, order=c(0,1,1), seasonal=list(order=c(0,1,2), period=12), method = "ML"
## MA(12)
fit_ma12 <- arima(sflow.bc, order=c(0,1,12), seasonal=list(order=c(0,1,0), period=12), method = "ML")
## AR (12)
fit_ar12 <- arima(sflow.bc, order=c(12,1,0), seasonal=list(order=c(0,1,0), period=12), method = "ML")
```

```
##
## Call:
## arima(x = sflow.bc, order = c(0, 1, 0), seasonal = list(order = c(0, 1, 1),
##     period = 12), method = "ML")
##
## Coefficients:
##          sma1
##       -0.4149
## s.e.   0.1198
##
## sigma^2 estimated as 5397:  log likelihood = -544.13,  aic = 1092.26

##
## Call:
```

```
## arima(x = sflow.bc, order = c(0, 1, 0), seasonal = list(order = c(1, 1, 0),
##     period = 12), method = "ML")
##
## Coefficients:
##          sar1
##       -0.3024
## s.e.   0.1002
##
## sigma^2 estimated as 5623:  log likelihood = -545.52,  aic = 1095.04

##
## Call:
## arima(x = sflow.bc, order = c(0, 1, 0), seasonal = list(order = c(1, 1, 1),
##     period = 12), method = "ML")
##
## Coefficients:
##         sar1     sma1
##       0.2303  -0.6298
## s.e.  0.2426   0.2278
##
## sigma^2 estimated as 5314:  log likelihood = -543.74,  aic = 1093.47

##
## Call:
## arima(x = sflow.bc, order = c(0, 1, 1), seasonal = list(order = c(0, 1, 1),
##     period = 12), method = "ML")
##
## Coefficients:
##           ma1     sma1
##       -0.1231  -0.4407
## s.e.   0.1006   0.1194
##
## sigma^2 estimated as 5296:  log likelihood = -543.4,  aic = 1092.81

##
## Call:
## arima(x = sflow.bc, order = c(0, 1, 1), seasonal = list(order = c(0, 1, 2),
##     period = 12), method = "ML")
##
## Coefficients:
##           ma1     sma1     sma2
##       -0.1055  -0.4232  -0.0757
## s.e.   0.1056   0.1181   0.1284
##
## sigma^2 estimated as 5256:  log likelihood = -543.23,  aic = 1094.45

##
## Call:
## arima(x = sflow.bc, order = c(0, 1, 12), seasonal = list(order = c(0, 1, 0),
##     period = 12), method = "ML")
##
## Coefficients:
##           ma1     ma2     ma3      ma4      ma5      ma6      ma7      ma8
```

```
##       -0.1798  0.0110  0.0092  -0.1173  -0.0375  -0.0340  -0.0997  -0.0575
## s.e.   0.1251  0.1172  0.1141   0.1109   0.1159   0.1201   0.1282   0.1301
##          ma9     ma10    ma11     ma12
##       0.0522  -0.1964  0.1656  -0.5159
## s.e.  0.1186   0.1092  0.1516   0.1365
##
## sigma^2 estimated as 4516:  log likelihood = -537.65,  aic = 1101.29

##
## Call:
## arima(x = sflow.bc, order = c(12, 1, 0), seasonal = list(order = c(0, 1, 0),
##     period = 12), method = "ML")
##
## Coefficients:
##          ar1     ar2     ar3      ar4      ar5     ar6      ar7      ar8
##       0.0315  0.0177  0.0595  -0.0181  -0.0997  0.1610  -0.2016  -0.0616
## s.e.  0.0952  0.0952  0.0930   0.0952   0.0952  0.0944   0.0923   0.0938
##          ar9     ar10    ar11     ar12
##       0.1058  -0.1090  0.0818  -0.3518
## s.e.  0.0941   0.0952  0.0958   0.1001
##
## sigma^2 estimated as 4869:  log likelihood = -539.29,  aic = 1104.58
```

```
# Table of AICc values
Model <- c("A. SMA(1)_12","B. SAR(1)_12", "C. SARIMA(0,1,0)x(1,1,1)_12", "D. SARIMA(0,1,1)x(0,1,1)_12",
AICc_Value <- c(AICc(fit1_sma1), AICc(fit1_sar1), AICc(fit1_sarima1), AICc(fit2_sarima1), AICc(fit3_sari
Significant_Coefficients <- c("SMA1", "SAR1", "SMA1", "SMA1", "SMA1", "MA12", "AR12")
aicc_summary <- data.frame(Model, AICc_Value, Significant_Coefficients)
kable(aicc_summary) %>% kable_styling(position = "center") # best one is seasonally differenced and de-

# FIGURE 10
uc.check(c(1, -0.423, -0.076), plot_output=T, print_output=F)

# Diagnostic Checking
## Model A
par(mfrow=c(1,3))
res_sma1 <- residuals(fit1_sma1)
plot_hist(res_sma1, "Histogram of SMA(1)_12 Residuals")
plot.ts(res_sma1, main="Time Series of SMA(1)_12 Residuals")
qqPlot(res_sma1, main="Normal Q-Q Plot of SMA(1)_12", id=FALSE)

shapiro.test(res_sma1)
Box.test(res_sma1, lag =11, type = c("Box-Pierce"), fitdf = 1)
Box.test(res_sma1, lag = 11, type = c("Ljung-Box"), fitdf = 1)
Box.test((res_sma1)^2, lag = 11, type = c("Ljung-Box"), fitdf = 0)

par(mfrow=c(1,2))
acf(res_sma1, lag.max=60)
pacf(res_sma1, lag.max=60)

## Model D
par(mfrow=c(1,3))
res_fit2_sarima1 <- residuals(fit2_sarima1)
plot_hist(res_fit2_sarima1, "Histogram of SARIMA(0,1,1)x(0,1,1)_12 Residuals")
```

```r
plot.ts(res_fit2_sarima1, main="Time Series of SARIMA(0,1,1)x(0,1,1)_12 Residuals")
qqPlot(res_fit2_sarima1, main="Normal Q-Q Plot of SARIMA(0,1,1)x(0,1,1)_12", id=FALSE)

par(mfrow=c(1,2))
acf(res_fit2_sarima1, lag.max=60)
pacf(res_fit2_sarima1, lag.max=60)

shapiro.test(res_fit2_sarima1)
Box.test(res_fit2_sarima1, lag =11, type = c("Box-Pierce"), fitdf = 2)
Box.test(res_fit2_sarima1, lag = 11, type = c("Ljung-Box"), fitdf = 2)
Box.test((res_fit2_sarima1)^2, lag = 11, type = c("Ljung-Box"), fitdf = 0)

## Model E
par(mfrow=c(1,3))
res_fit3_sarima1 <- residuals(fit3_sarima1)
plot_hist(res_fit3_sarima1, "Histogram of SARIMA(0,1,1)x(0,1,2)_12 Residuals")
plot.ts(res_fit3_sarima1, main="Time Series of SARIMA(0,1,1)x(0,1,2)_12 Residuals")
qqPlot(res_fit2_sarima1, main="Normal Q-Q Plot of SARIMA(0,1,1)x(0,1,2)_12", id=FALSE)

par(mfrow=c(1,2))
acf(res_fit3_sarima1, lag.max=60)
pacf(res_fit3_sarima1, lag.max=60)

shapiro.test(res_fit3_sarima1)
Box.test(res_fit3_sarima1, lag =11, type = c("Box-Pierce"), fitdf = 3)
Box.test(res_fit3_sarima1, lag = 11, type = c("Ljung-Box"), fitdf = 3)
Box.test((res_fit3_sarima1)^2, lag = 11, type = c("Ljung-Box"), fitdf = 0)

# FORECASTING
par(mfrow=c(1,2))

## forecast on transformed data
sflow.bc.clean <- as.vector(sflow.bc)
fit.E <- arima(sflow.bc.clean, order=c(0,1,1), seasonal = list(order = c(0,1,2), period = 12), fixed=NU
pred.tr <- predict(fit.E, n.ahead = 12)
U.tr <- pred.tr$pred+2*pred.tr$se # upper bound of PI
L.tr <- pred.tr$pred-2*pred.tr$se # lower bound of PI

ts.plot(sflow.bc.clean, xlim=c(1,length(sflow.bc.clean)+12), ylim = c(min(sflow.bc.clean),max(U.tr)), ma
lines(U.tr, col="blue", lty="dashed")
lines(L.tr, col="blue", lty="dashed")
points((length(sflow.bc.clean)+1):(length(sflow.bc.clean)+12), pred.tr$pred, col="red")

## forecast on original data
sflow.clean <- as.vector(sflow.train.2)
pred.orig <- reverse_bc(pred.tr$pred, lambda)
U <- reverse_bc(U.tr, lambda)
L <- reverse_bc(L.tr, lambda)

sflow.clean.full <- as.vector(sflow.ts.cut)

ts.plot(sflow.clean.full, xlim=c(1,length(sflow.clean)+12), ylim = c(min(sflow.clean),max(U)), main="Or
lines(U, col="blue", lty="dashed")
```

```
lines(L, col="blue", lty="dashed")
points((length(sflow.clean)+1):(length(sflow.clean)+12), pred.orig, col="red")
```