DSA301 Time Series Analysis

Professor Benjamin Ee

Project Report

Done by:
Bharat Gangwani
Heng Hon Lon Jaden
Rajvardhan Bhartia
Sim Wei Jie
Zhou Min Yuan

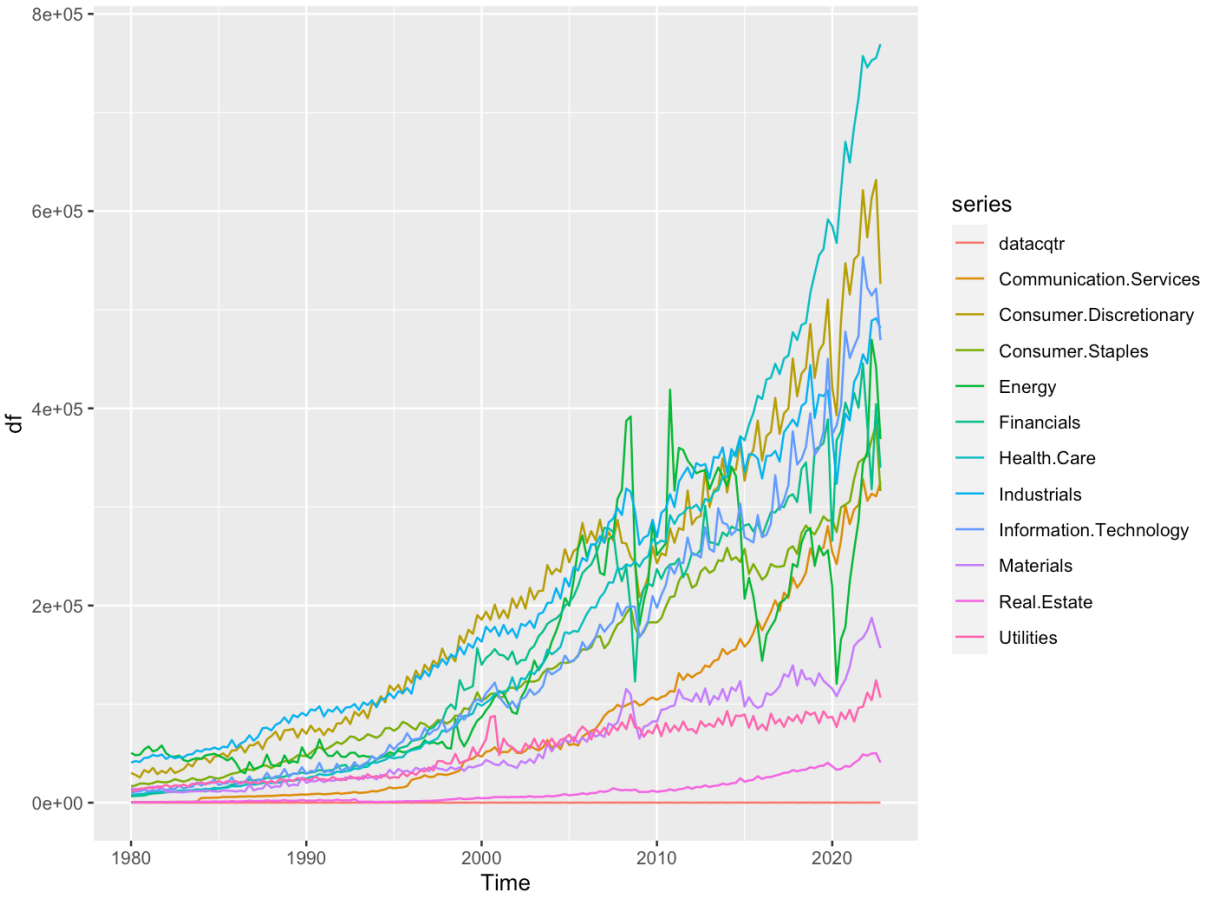
Content Page

1. Introduction

This project forecasts aggregate corporate revenues whilst accounting for general and idiosyncratic characteristics that might be observed across industries through the help of various time series forecasting methods (i.e., SARIMA, ARIMA-X and VAR). We account for inter-industry correlation, intra-industry seasonality, black swan events, other macro variables such as interest rates etc., in our analysis to come up with a robust and accurate model. After basic exploratory data analysis on our entire dataset of eleven industries, we narrow down on five (Information Technology, Healthcare, Consumer Discretionary, Financials, Utilities) and try to predict their future revenues.

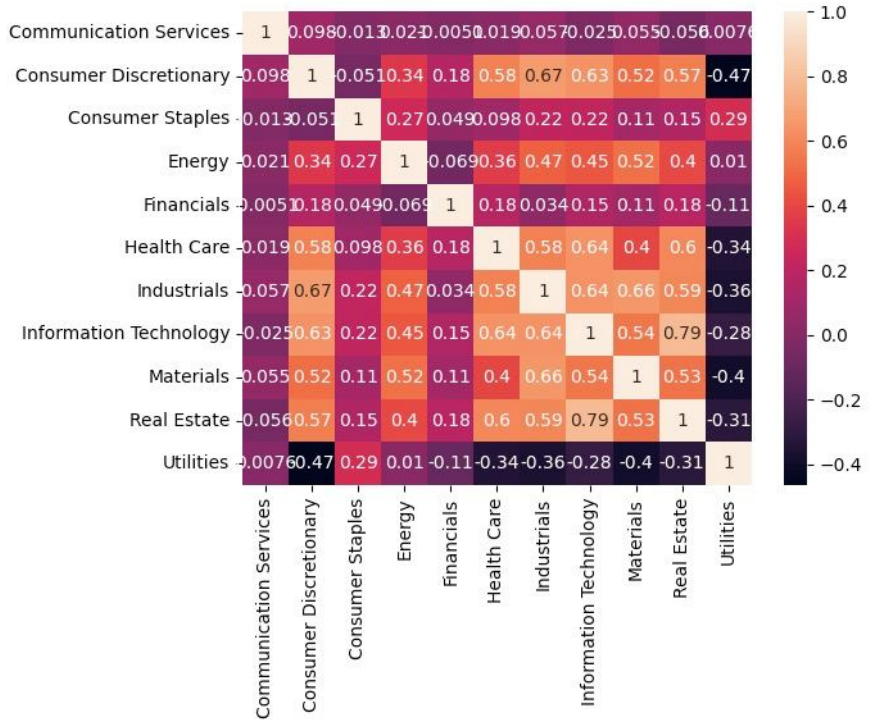
2. EDA

After successfully collecting the data, before deep-diving into the models, we conducted intermediate exploratory data analysis which involved: plotting raw data, understanding correlation plots and summary statistics, decomposing time series' using various techniques etc., to form basic hypotheses about the data we were working with. Apart from the general plots, we conducted the same analysis for every industry looking for some idiosyncratic characteristic in the form of seasonality, cycle or trend. Moving forward, however, we discuss the utilities industry in detail as it had prominent results.

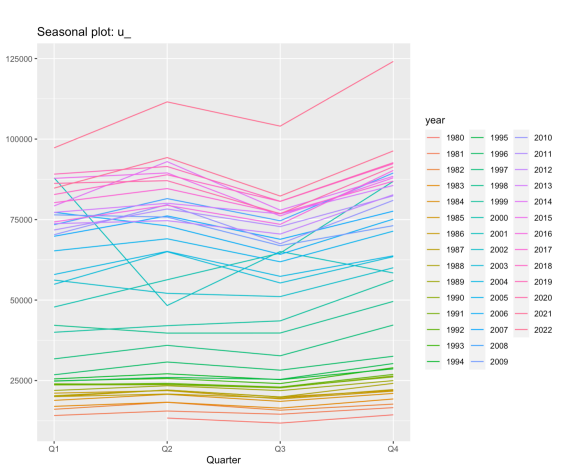


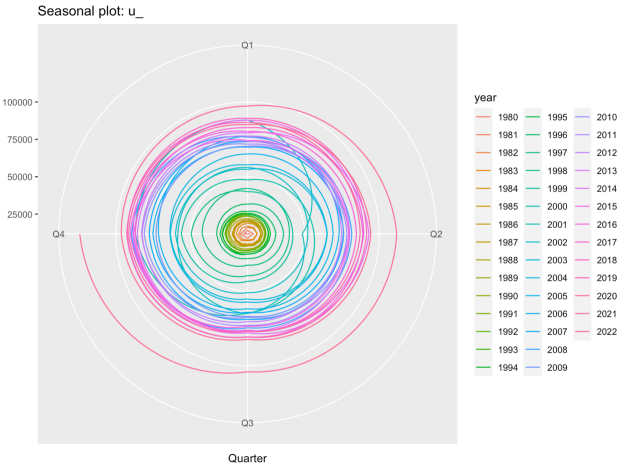
The only visually observable characteristic from the entire plot above, is simply an overall increase in the revenue of each industry over time, however, at different rates. The

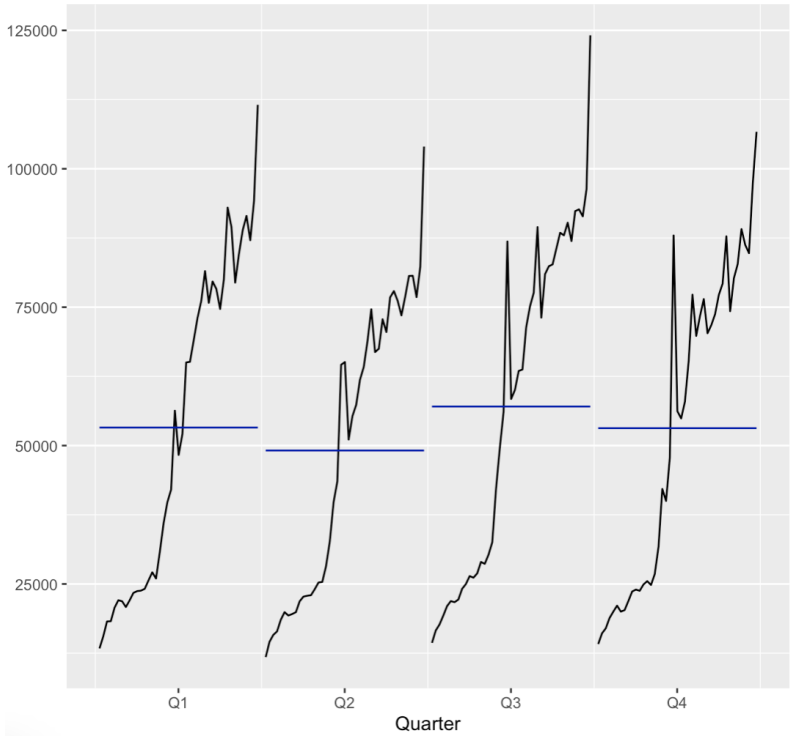
correlation plot shows the percentage change in growth across a large range for different industries, some being negatively correlated with most industries (Utilities) wherein some having almost no correlation with other industries (Communication services).



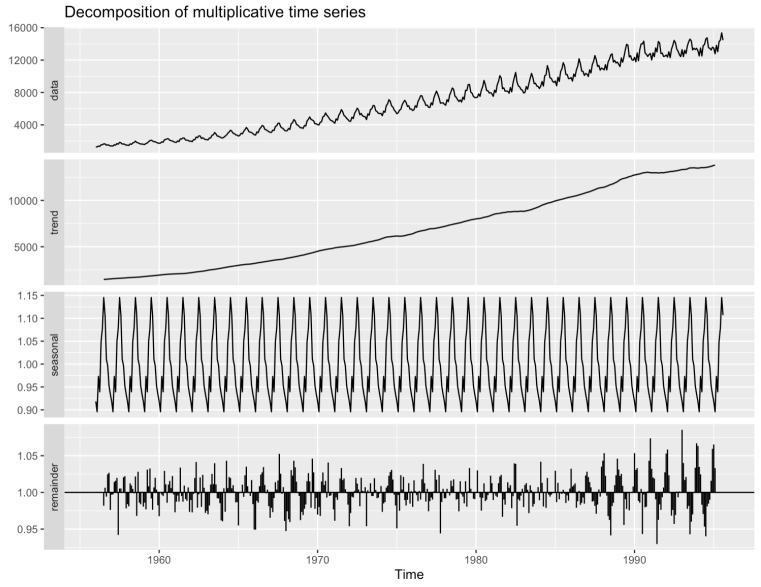
To reiterate, we display the graphs corresponding to the utilities industry below, however we conducted the same for all. Apart from the minimal wave-like pattern in the first plot on the left below, the symmetry in the seasonal and subseries plots across all industries was quite high, i.e., there was very little seasonal variation across quarters for any industry in particular.







Industry aggregate revenues do not display any significant seasonality. One possible reason could be diversification, i.e., some companies outperform others within the industry and thus the aggregate seasonality gets neutralised. From the earlier plots we could see that the seasonal variation, if any, was not constant over time and thus we did not use additive decomposition to aid our analysis. We perform a multiplicative time series decomposition on the utilities industry to try and isolate the effect of seasonality.



	df.Information.Technology	df.Health.Care	df.Consumer.Discretionary	df.Financials	df.Utilities	fedfunds	popgrowth	gdpchange
df.Information.Technology	NA	NA	NA	NA	NA	NA	NA	NA
df.Health.Care	NA	NA	NA	NA	NA	NA	NA	NA
df.Consumer.Discretionary	NA	NA	NA	NA	NA	NA	NA	NA
df.Financials	NA	NA	NA	NA	NA	NA	NA	NA
df.Utilities	NA	NA	NA	NA	NA	NA	NA	NA
fedfunds	NA	NA	NA	NA	NA	NA	NA	NA
popgrowth	NA	NA	NA	NA	NA	NA	NA	NA
gdpchange	NA	NA	NA	NA	NA	1	NA	NA

	df.Information.Technology	df.Health.Care	df.Consumer.Discretionary	df.Financials	df.Utilities	fedfunds	popgrowth	gdpchange
df.Information.Technology	NA	NA	NA	NA	1	NA	1	NA
df.Health.Care	1	NA	NA	NA	NA	NA	1	NA
df.Consumer.Discretionary	1	1	NA	NA	1	NA	1	NA
df.Financials	NA	NA	NA	NA	NA	1	NA	NA
df.Utilities	1	1	1	NA	NA	NA	NA	NA
fedfunds	1	1	1	NA	NA	NA	NA	NA
popgrowth	1	1	NA	NA	1	NA	NA	NA
gdpchange	NA	NA	NA	NA	NA	NA	NA	NA

The outcome of this EDA gets achieved when we look at the seasonal component of the various plots. We see that there is seasonality amongst the revenues of the different industries, in this case, the utilities industry. The seasonality component, if narrowed into each year, shows a cycle which peaks mid year and sees a trough during the beginning and the end. Each industry has their own idiosyncratic reasons, i.e., for the utilities industry it is far more obvious how during the summer (mid year) we face higher temperatures and thus our utility bills are higher due to the greater usage of our home appliances.

We run the granger causality test to understand whether any industry granger causes the other, however, we do not see any significant results. Having formed an initial hypothesis about our data we narrow down on five industries namely Information Technology, Healthcare, Consumer Discretionary, Financials, Utilities to identify the most efficient and accurate model using the entire kitchen sink of tools we have been taught in class.

3. ARIMA-X

Best model tables summary

Industry	Model	MAE	RMSE
Information Technology	ARIMA(1,1,1) with lagged Real Estate Revenues	33,438.30	37,688.75
Healthcare	ARMA(1,1) w/ deterministic trend	14,175.89	18,546.779
Consumer Discretionary	ARIMA(5, 1, 2) w/ %GDP change as X regressor	20,829.17	32,118
Financials	ARIMA(1, 1, 0) w/ %GDP change & healthcare as X regressor	32,571.68	50,689.07
Utilities	ARIMA(1, 1, 0) w/ materials & real estate as X regressor	3,550.41	4,413.066

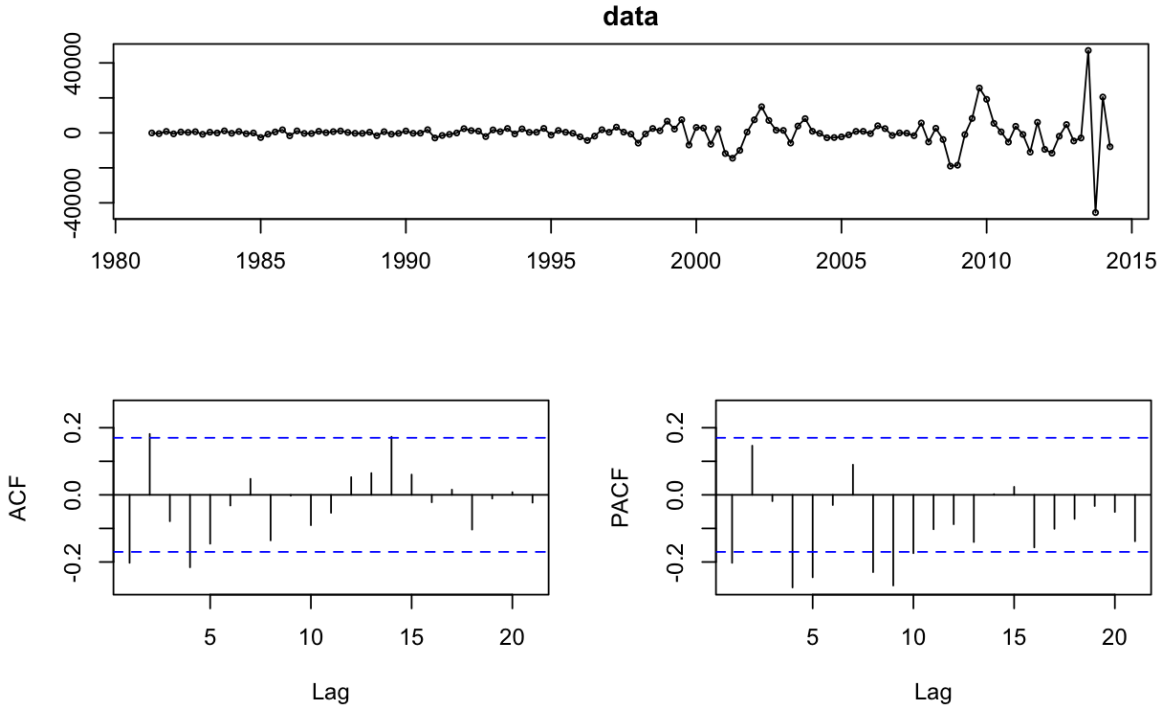
3.1 Information Technology

The benchmark model selected for our analysis was a seasonal naive model with drift, which gave an out-of-sample RMSE = 97697.53 (Appendix A3.1.1)

3.1.1 SARIMA

The first model we tried for information technology revenue was a seasonal ARIMA (SARIMA) model. We note that we could have used SARIMA to model just the seasonally adjusted component of the data, but we wanted to see the performance of SARIMA without seasonally adjusting the data. To form hypotheses on the orders of the SARIMA model, we plotted the ACF & PACF of the differenced Information Technology (IT) series.

```
# Run seasonal differences and normal differences on data to retrieve
differenced data

# diffData and diagnostics are custom functions used to difference data
# and checking if data is stationary respectively
# Function is defined in the appendix 3.1.2
finalDiffData <- diffData(ts_it_split$train)[[1]]

# Check nsdiffs and ndiffs
diffData(ts_it_split$train)[2:3] # 1 seasonal, 1 non-seasonal

# Run KPSS test and tsdisplay residuals to confirm that data is
stationary
diagnostics(finalDiffData)
```



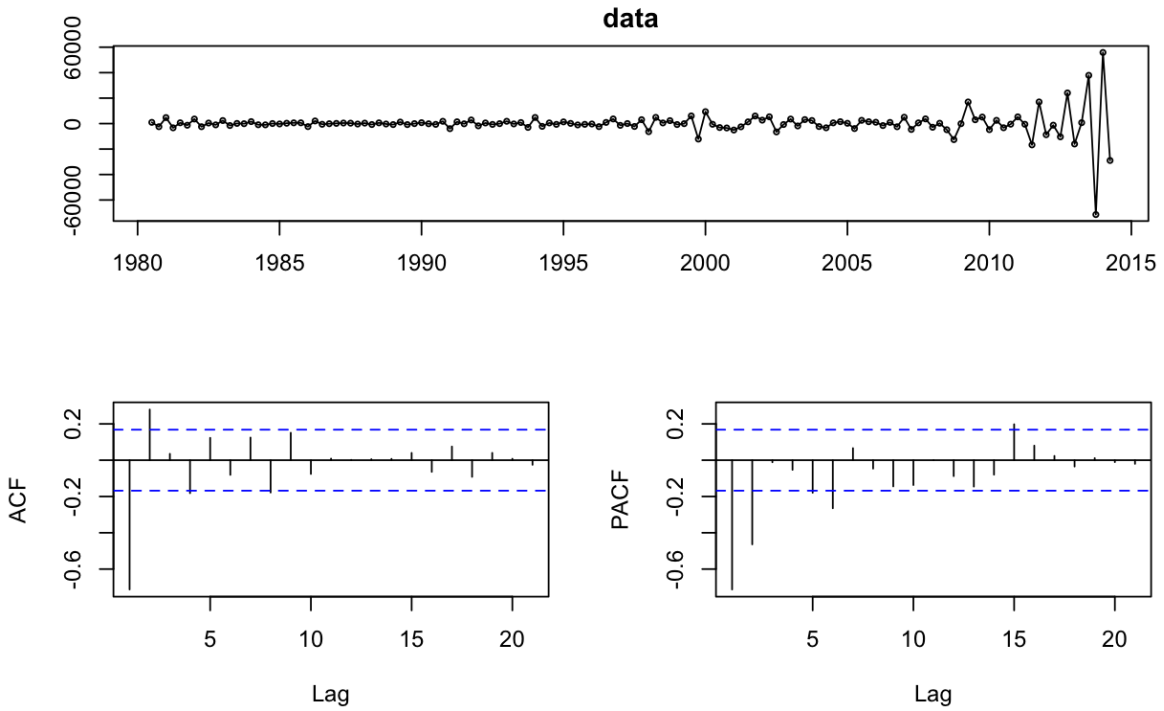
From the PACF plot, it looks like there is a sharp drop at seasonal lag 1, which suggests a seasonal AR1 component. There also seems to be a sharp drop in PACF at non-seasonal lag 4 suggesting a non-seasonal AR4 component. The ACF plot shows a significant spike at non-seasonal lag 4 suggesting a non-seasonal MA4 component. It does not look like there is a seasonal MA component. Since our data requires 1 seasonal difference and 1 non-seasonal difference, we will run a SARIMA(4,1,4)(1,1,0) model. This results in an AICc=654.99, and a p-value=0.9137 for the Ljung-Box test, meaning that we cannot reject

the null hypothesis that the residuals of our model are not autocorrelated (Appendix 3.1.3). The resulting test set RMSE = 38659.999 (Appendix 3.1.4).

3.1.2 STL Decomposition

Next, we tried a Seasonal and Trend decomposition using Loess (STL). We plotted the ACF & PACF of the seasonally adjusted Information Technology series for an estimation of the order of ARIMA to fit.

```
seasadj_it <- seasadj(mstl(ts_it))
seasonal_it <- seasonal(mstl(ts_it))
oos <- round(length(ts_it)*0.20)
seasadj_it_split <- ts_split(seasadj_it, sample.out = oos)
seasonal_it_split <- ts_split(seasonal_it, sample.out = oos)
diffData(seasadj_it_split$train)[2:3]
diagnostics(diffData(seasadj_it_split$train)[[1]])
```

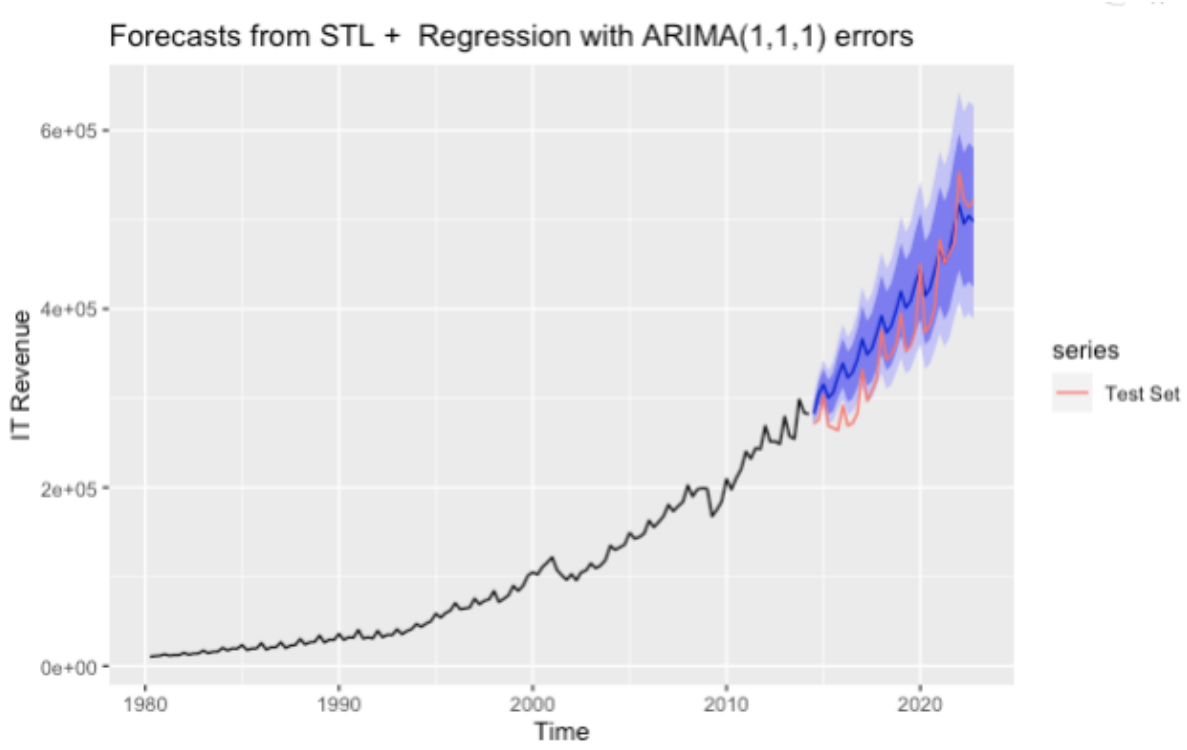


From the ACF plot, there is a significant spike at lag 1 suggesting a MA(1) component. However, it also looks like the spike at lag 2 is significant, so we'll try a MA(2) component as well. PACF plot shows significant spike at lag 2. Hence we will test ARIMA(2, 2, 1) and ARIMA(2, 2, 2).

Since ARIMA(2, 2, 1) has a lower AICc of 62.16 compared to ARIMA(2, 2, 2) with an AICc of 62.31, we will proceed with ARIMA(2, 2, 1) (Appendix 3.1.5). ARIMA(2, 2, 1) also passes the Ljung-Box test (Appendix 3.1.6) meaning there is no time series information present in the residuals of our model. This results in an out-of-sample RMSE of 58779.62 (Appendix 3.1.7).

3.1.2 ARIMA-X with Real Estate Revenues

From our exploratory data analysis, we noticed that differenced Information Technology Revenues had the highest correlation with differenced Real Estate Revenues. One possible reason for this could be the high correlation of the Real Estate sector with the US economy, making it a good proxy indicator for the health of the US economy. Given how most industries, including Information Technology, are highly correlated with the performance of the US economy, we decided to explore if using Real Estate revenues as a regressor in an ARIMA-X model would help with our prediction of Information Technology Revenues. To avoid any look-ahead bias possible, we lagged Real Estate Revenues by 1 quarter. Appendix 3.1.8 shows that the accuracy of our ARIMA-X model has the lowest RMSE thus far for Information Technology of 37688.75.

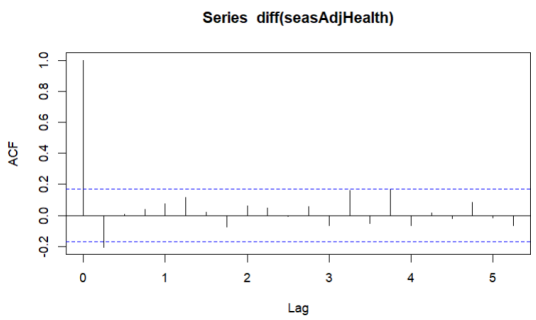


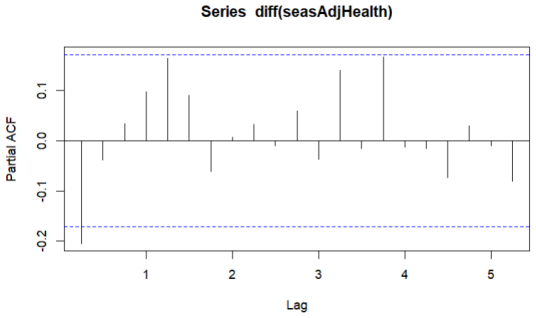
In conclusion, the best model for Information Technology is an STL Decomposed ARIMA-X model with lagged Real Estate Revenues as the X regressor.

3.2 Healthcare

Healthcare is an unbelievably well-behaved series and quite easy to model. A Box-Cox transformation converts it into a nearly linear trend. All models are estimated on the Box-Cox transformed seasonally-adjusted series after decomposition. The RMSEs reported are calculated after adding seasonality to the model forecasts using snaive on the seasonal averages estimated during the decomposition.

The out-of-sample RMSE for a random walk forecast (A3.2.1), which we use as a benchmark, is 20,045.671. It should be noted that due to the well-behaved nature of the series, the ARIMA(0,1,0) is itself a good model with the Ljung-Box test being unable to deliver evidence for autocorrelation among the residuals till lag 12.

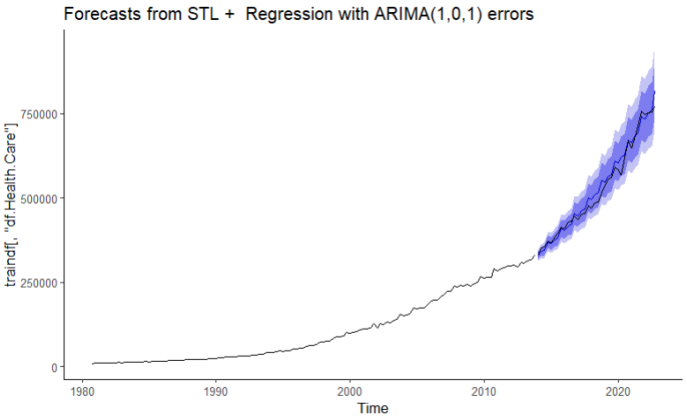




Observing the ACF and PACF, we hypothesise an ARIMA(1,1,1) model to be a good fit for the series. Indeed, the RMSE for the model (A3.2.2) is lower at 19,400.848. We try adding population growth rate and gdp growth rate to the model as external regressors but witness an increase in the out-of-sample RMSE (A3.2.3). Hence, we leave them out in our final model.

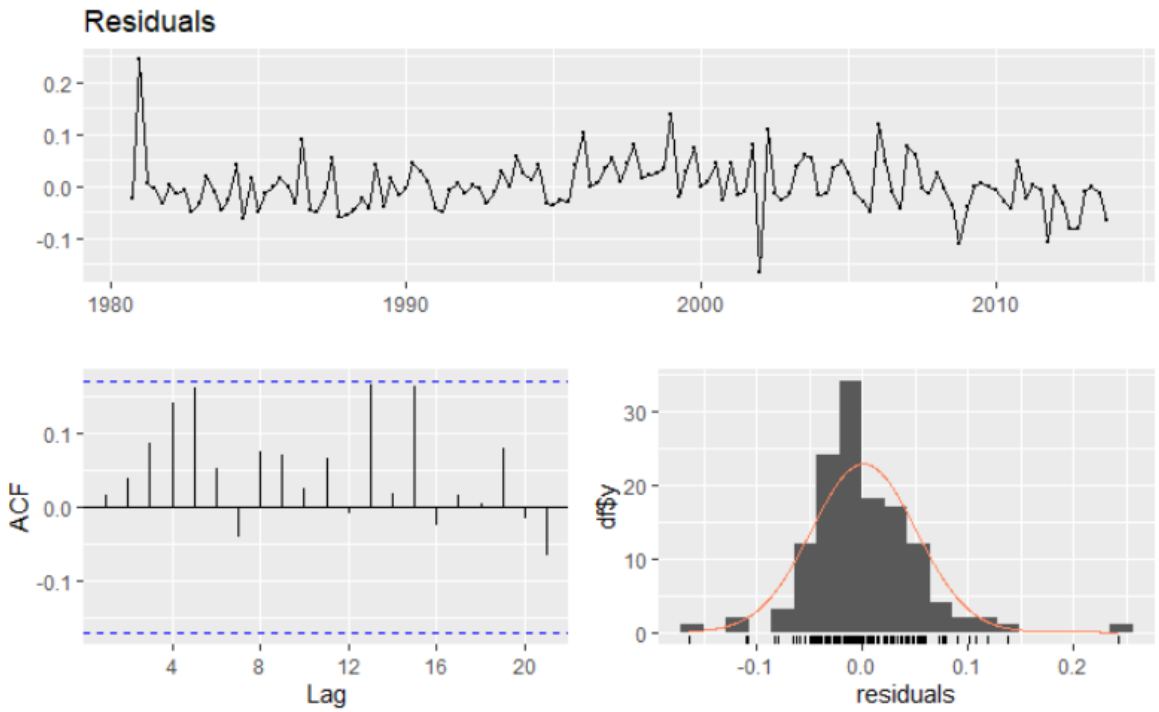
The last and best model we fit is an ARMA(1,1) with a deterministic linear trend (A3.3.4). The model delivers the lowest out-of-sample RMSE at 18,546.779. The Ljung-Box test is unable to detect autocorrelation among the residuals as well and hence we find the following model as the best fit for the series:

```
Series: x
Regression with ARIMA(1,0,1) errors

Coefficients:
         ar1      ma1  intercept    xreg
      0.9814  -0.2329    12.0716  0.0521
s.e.  0.0144   0.0858     0.1907  0.0021

sigma^2 = 0.002523:  log likelihood = 209.71
AIC=-409.42   AICc=-408.95   BIC=-394.97
```



It should also be noted that the residuals from the model are safely stationary, further suggesting that it is a good fit.

```
Warning: p-value greater than printed p-value
        KPSS Test for Level Stationarity

data:  md14f$residuals
KPSS Level = 0.27449, Truncation lag parameter = 4, p-value = 0.1
```



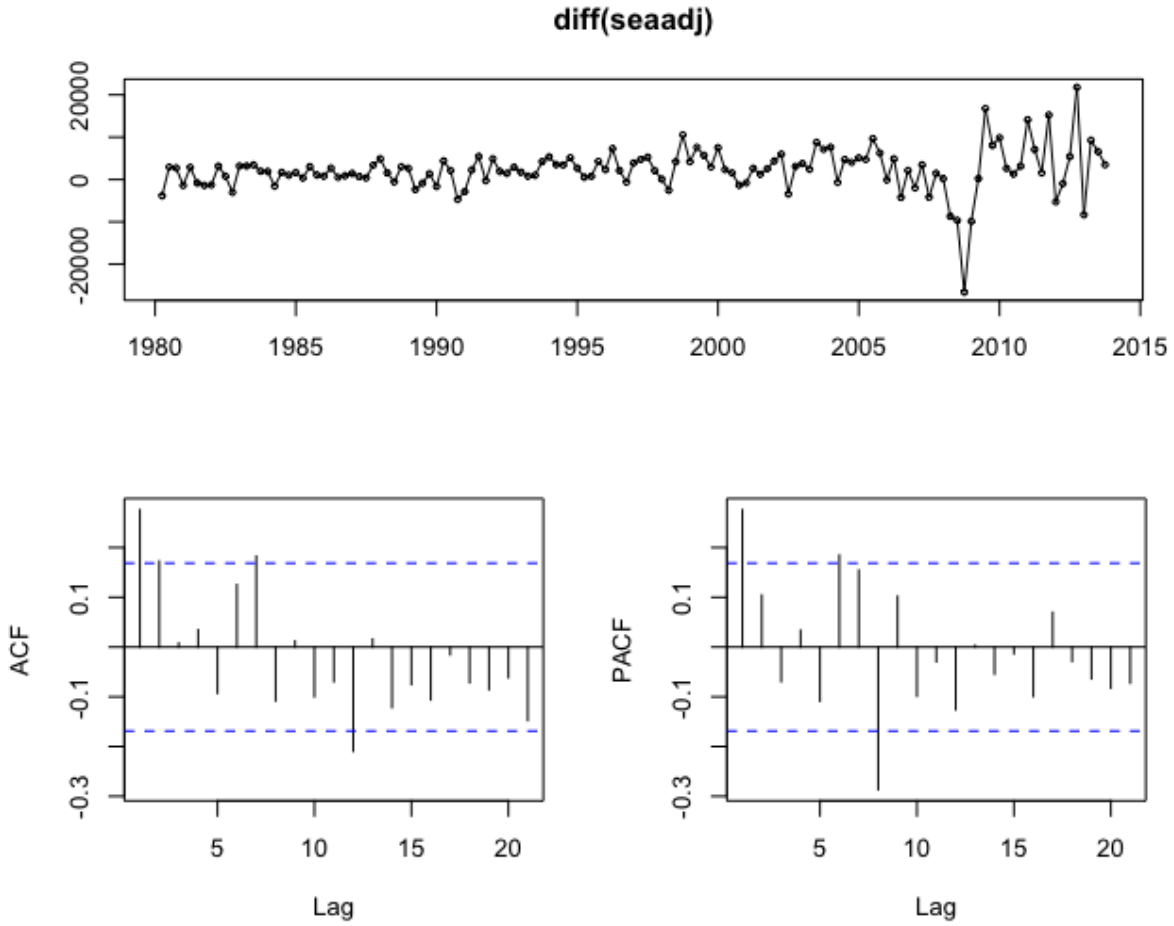
On an industry level, we can observe that COVID does not seem to have positively or negatively shocked corporate revenues to a significant degree given that a linear forecast with ARMA(1,1) components delivers a good fit for the out-of-sample series.

3.3 Consumer Discretionary

Firstly, we generated the seasonal naive model with drift as a benchmark for our analysis. The test set RMSE = 80282.27 (Appendix A3.3.1).

Then, we plotted the ACF & PACF of the seasonally adjusted Consumer Discretionary series for an estimation of the order of ARMA to fit it to.

```
seaadj = seasadj(mstl(cd_split$train)) #seasonally adjusted
season = seasonal(mstl(cd_split$train)) #seasonal
ndiffs(seaadj) #1
nsdiffs(seaadj) #0
tsdisplay(diff(seaadj))
```



From the plots, we can tell that without X-regressors, ARIMA (1, 1, 1) or ARIMA (2, 1, 2) could be suitable. We first tried the ARIMA (1, 1, 1) model, but it failed the Ljung Box test at 5% level of significance (Appendix A3.3.2). The ARIMA (2, 1, 2) passed the Ljung Box test and has AICC = -47.16217 (Appendix A3.3.3). Just to be sure, we tried ARIMA (3, 1, 2) which gave a larger AICC = -45.11857 (Appendix A3.3.4). This leads us to conclude that when X regressors are not added, ARIMA (2, 1, 2) is the best with the test set RMSE = 32513.82.

Next, we then tried to add the X regressors. The category of Consumer Discretionary includes companies such as Amazon, Tesla, Louis Vuitton and McDonalds, mostly companies that have higher income elasticity and perform better when the economy is booming and drops during a crisis or recession. Hence, we will be adding various X regressors that might be reflective of GDP performance, namely the seasonally adjusted percentage change in GDP (data from FRED) and corporate revenues of the Financial and Information Technology industries. All the X regressors will be lagged by 1 quarter to ensure that we do not have the lookahead bias.

We started off by using Information Technology as a X regressor. However, it actually made the performance on the test set worse, with RMSE = 36786.94 (Appendix A3.3.5). The use of Financials as an X regressor proved to be slightly better; it lowered RMSE to 32438.5 (Appendix A3.3.6). However, both of these variables were unable to capture the fluctuations

in corporate revenue due to COVID. This is so as both data series are undifferenced, and are probably more useful in capturing the overall trend instead. However, since ARIMA (2, 1, 2) is able to capture the general trend, the improvements from these two X regressors are marginal.

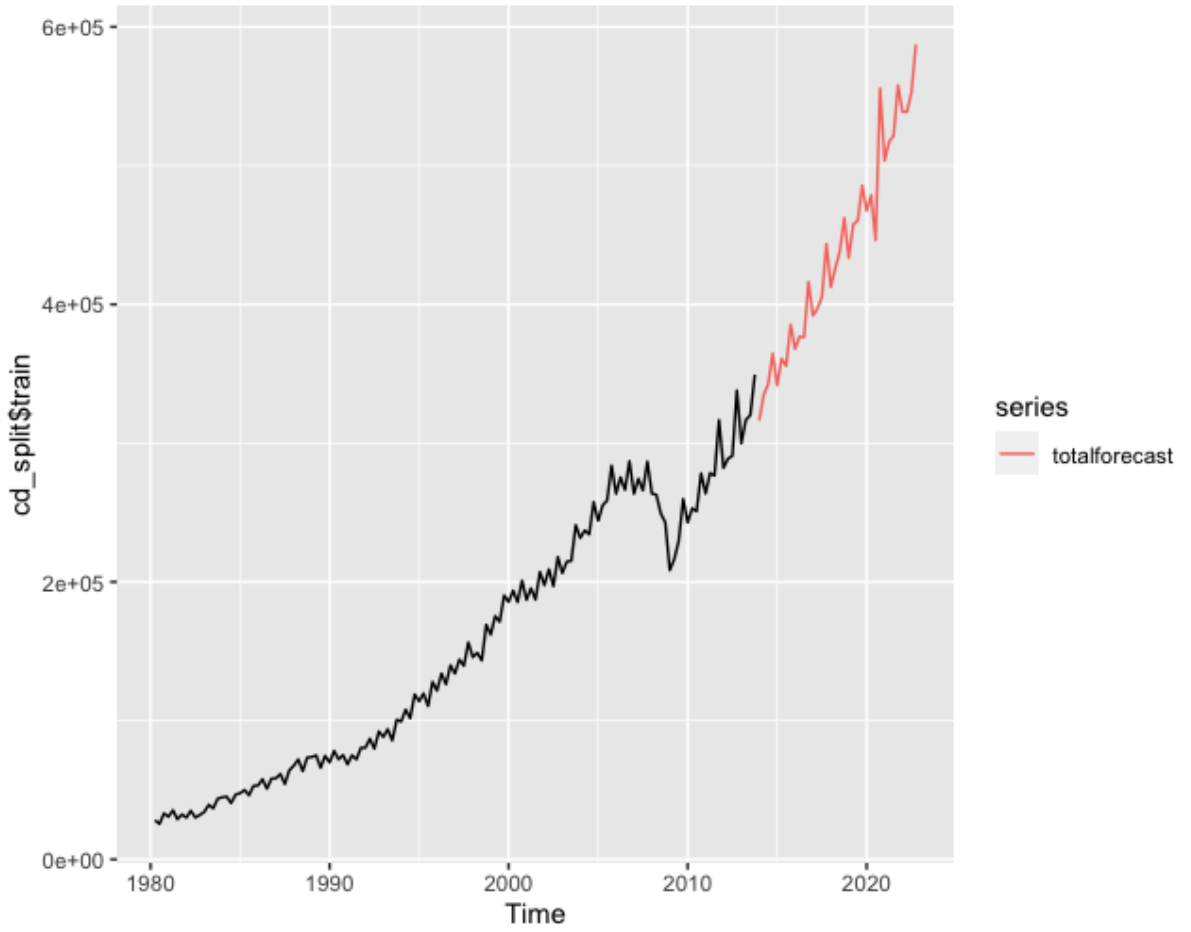
Hence, we decided to try using the lagged % change in seasonally adjusted US GDP for the same time period as an X regressor. We feel that this will help the model learn about the small fluctuations due to COVID. As the addition of this to an ARIMA (2, 1, 2) model generates a model that fails the Ljung Box test (Appendix A3.3.7), we increased the AR value until the Ljung Box test is passed, and obtained a better model using ARIMA (5, 1, 2) with RMSE = 32118.

```
arima1 <- Arima(seaadj, order=c(5,1,2),
                xreg=cbind(us_lag_split$train), lambda="auto",
include.drift = TRUE)
totalforecast = (forecast(arima1, xreg=cbind(us_lag_split$test),h = m,
lambda=arima1$lambda))$mean + (snaive(season, h = m))$mean
arima1$coef
checkresiduals(arima1)
accuracy(totalforecast, cd_split$test)
```

```
> arima1$coef
        ar1         ar2         ar3         ar4         ar5         ma1         ma2       drift
-0.44090541 -0.56825565  0.10267076  0.10453568 -0.25239925  0.64462798  0.67512332  0.11516577
       xreg
 0.01671349
> checkresiduals(arima1)

        Ljung-Box test

data:  Residuals from Regression with ARIMA(5,1,2) errors
Q* = 5.5991, df = 3, p-value = 0.1328

Model df: 9.   Total lags used: 12

> accuracy(totalforecast, cd_split$test)
              ME  RMSE      MAE       MPE    MAPE      ACF1 Theil's U
Test set 4396.89 32118 20829.17 0.2593105 4.27128 0.2109561 0.6890888
```

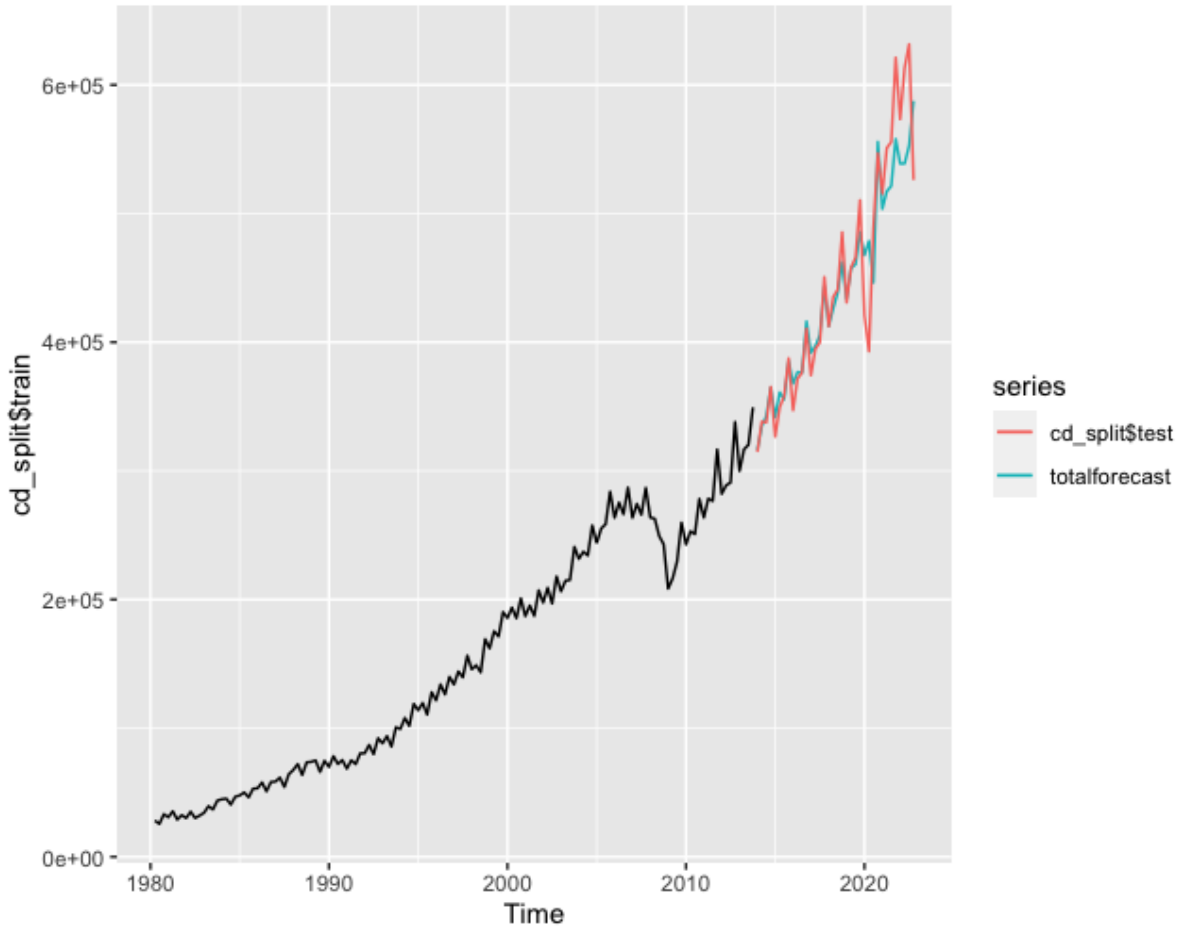
From the plot of the forecast done by the ARIMA model below, we can tell that there is a slight dip during COVID, which other models were not able to capture:

```
autoplot(cd_split$train) + autolayer(totalforecast)
```



It also overlays with the actual realised data quite well:

```
autoplot(cd_split$train) + autolayer(totalforecast) +
autolayer(cd_split$test)
```

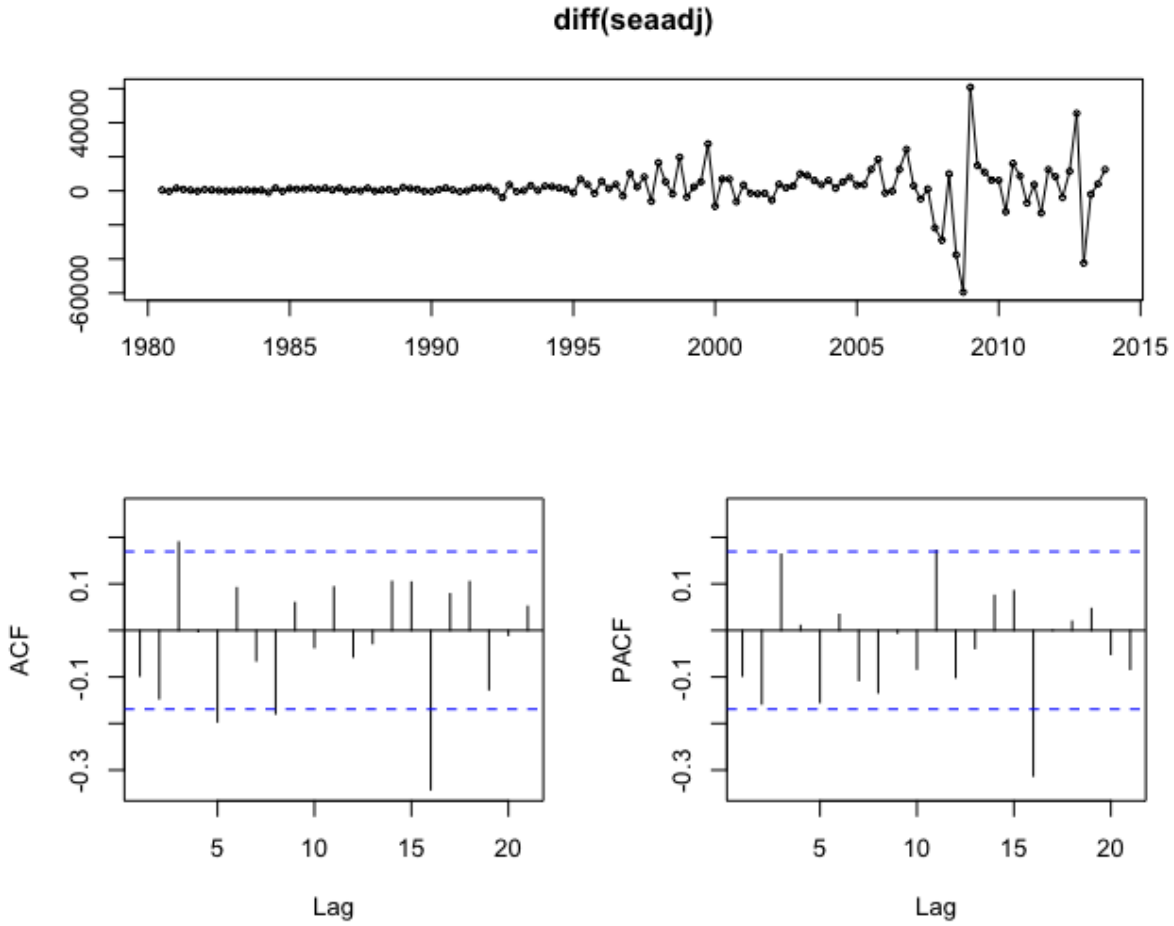


In conclusion, the best model for the Consumer Discretionary is an ARIMA-X model with % change in GDP as the X regressor and an ARIMA (5, 1, 2) model.

3.4 Financials

To start, we found the RMSE of the snaive with drift model = 40507.86 (Appendix A3.4.1). Then, we plotted the ACF & PACF of the seasonally adjusted Financial series for an estimation of the order of ARMA to fit it to.

```
seaadj = seasadj(mstl(f_split$train)) #seasonally adjusted
season = seasonal(mstl(f_split$train)) #seasonal
ndiffs(seaadj) #1
nsdiffs(seaadj) #0
tsdisplay(diff(seaadj))
```

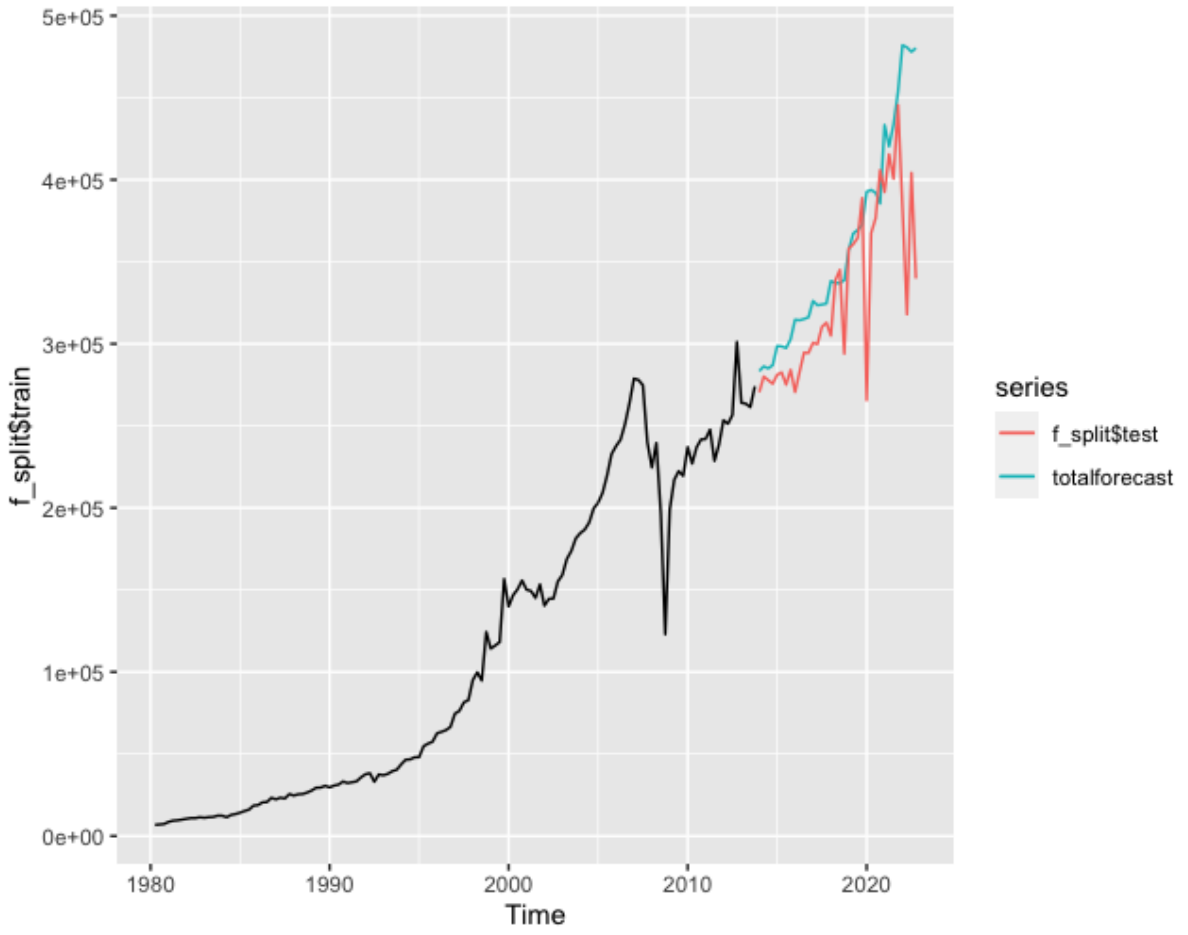


From here, it seems as though this is likely an ARIMA (0, 1, 0) model, which passes the Ljung Box test and has AICC = 959.2007 (Appendix A3.4.2). We also tried ARIMA (1, 1, 0), and it gave a slightly better AICC = -960.1954 (Appendix A3.4.3). To be sure, we tried ARIMA (2, 1, 0), which gave a worse AICC = -958.3833 (Appendix A3.4.4). However, none of these pure ARIMA models gave any good forecasts as they all tend to project an exponential growth, hence the best RMSE from ARIMA (1, 1, 0) is 577451.5.

As the ARIMA models do not properly capture information about the trend, choosing a X regressors that is able to correctly reflect trend information is important for Financials. Hence, we decided to try Healthcare and Consumer Discretionary, which are 2 industries with the highest amount of growth in corporate revenue, as X regressors. To attempt to capture the smaller fluctuations, we also tried to add in the % change in seasonally adjusted GDP.

Using Consumer Discretionary as a X regressor produced a model with RMSE = 580321.8 that under forecasts during the test period as the prediction still seemed to be exponential (Appendix A3.4.5). Meanwhile, Healthcare produced a slightly better model with RMSE = 53763.04 that over forecasts during the test period, although the trend predicted is a little erratic (Appendix A3.4.6). Combining Healthcare and the % change in GDP yielded a model with the best RMSE = 50689.07, and the upward trend is more regular.

```
arima1 <- Arima(seaadj, order=c(1,1,0),
                xreg=cbind(us_lag_split$train, hc_lag_split$train),
lambda="auto")
totalforecast = (forecast(arima1, xreg=cbind(us_lag_split$test,
hc_lag_split$test),h = m, lambda=arima1$lambda))$mean + (snaive(season,
h = m))$mean
autoplot(f_split$train) + autolayer(totalforecast) +
autolayer(f_split$test)
```




```
arima1$coef
checkresiduals(arima1)
accuracy(totalforecast, f_split$test)
```

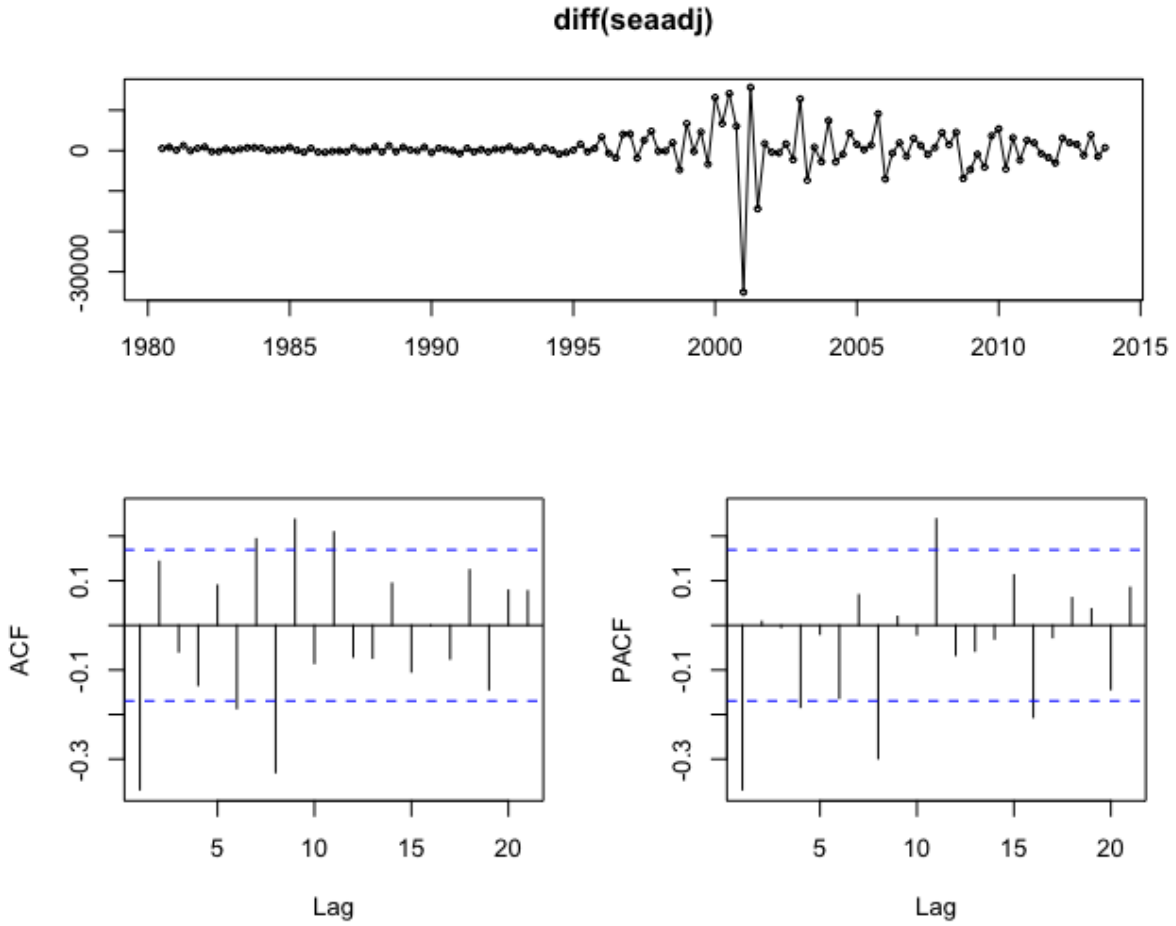
```
> arima1$coef
               ar1 us_lag_split$train hc_lag_split$train
      1.694001e-02      -7.501462e-05       8.068660e-08
> checkresiduals(arima1)

        Ljung-Box test

data:  Residuals from Regression with ARIMA(1,1,0) errors
Q* = 9.8612, df = 5, p-value = 0.07927

Model df: 3.   Total lags used: 8

> accuracy(totalforecast, f_split$test)
                ME     RMSE      MAE       MPE     MAPE      ACF1 Theil's U
Test set -29962.45 50689.07 32571.68 -9.472612 10.15041 0.3015856  1.129394
```

Hence, the best model is ARIMA-X with X regressors Healthcare, % change in US GDP with ARIMA (1, 1, 0). However, since the Financials data is highly volatile, we believe that even this best model is unable to capture the fluctuations in its corporate revenue. Since the series is not well behaved (unlike Healthcare in the previous section), forecasting Financial corporate revenue is not an easy task.

3.5 Utilities

Firstly, we found the snaive + drift model with RMSE = 7380.499 as the benchmark (Appendix A3.5.1). we plotted the ACF & PACF of the seasonally adjusted Utilities series for an estimation of the order of ARMA to fit it to.

```
seaadj = seasadj(mstl(u_split$train)) #seasonally adjusted
season = seasonal(mstl(u_split$train)) #seasonal
ndiffs(seaadj) #1
nsdiffs(seaadj) #0
tsdisplay(diff(seaadj))
```



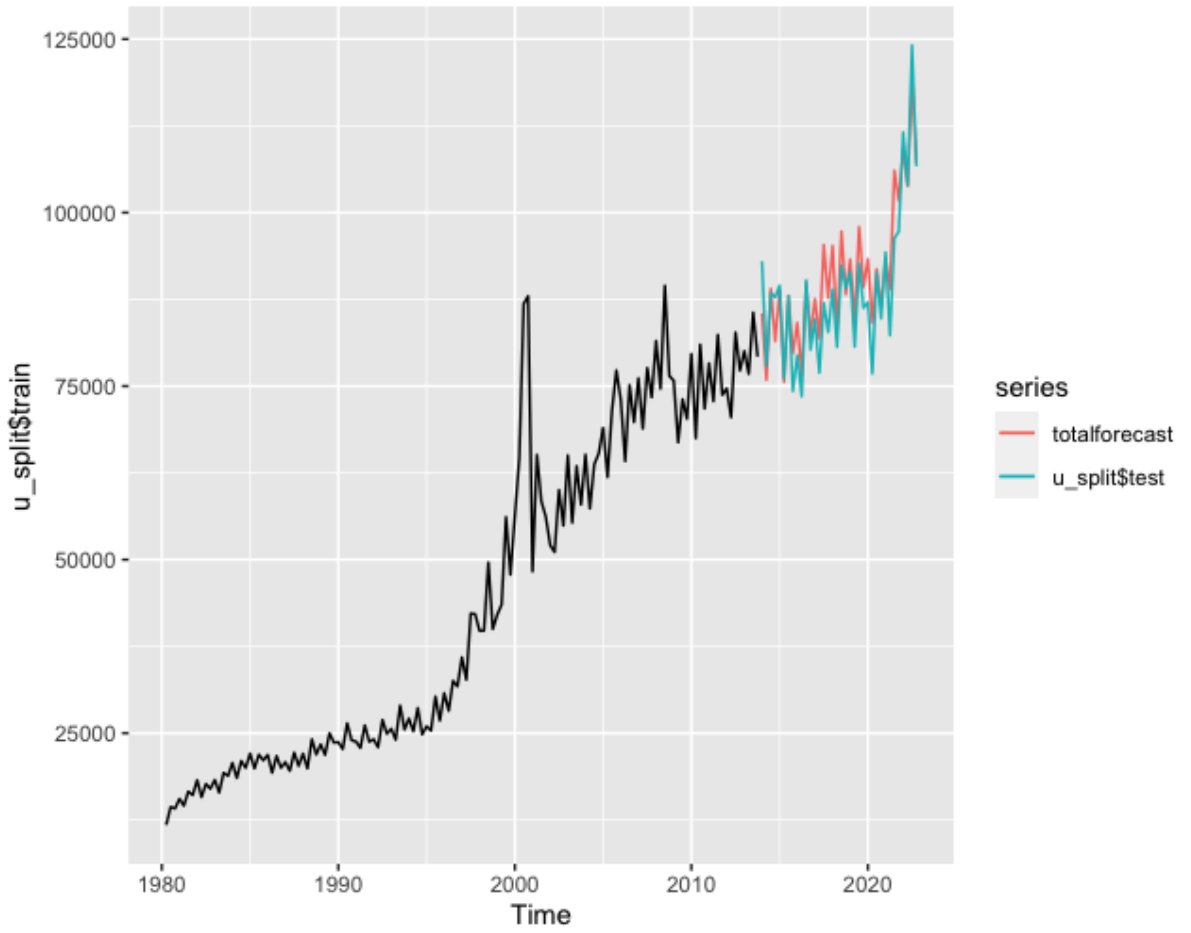
From the ACF and PACF plots, ARIMA (1, 1, 1) or ARIMA (1, 1, 0) seems plausible. The AICC value for ARIMA (1, 1, 1) is larger at -2532.444 (Appendix A3.5.2) while that for ARIMA (1, 1, 0) is -2534.286 (Appendix A3.5.3). When plotting the forecasted value, it gives an exponentially increasing trend when drift is included.

As the corporate revenue for Utilities is relatively flat since it is less elastic, we thought of using equally inelastic corporate revenue series such as Materials and Real Estate as X regressors. However, we were still getting exponential results, and we decided to remove the drift term for this case since Utilities is mostly flat, and the model would strictly perform better without a drift term.

After Materials was added as a X regressor, a model with test set RMSE = 5790.622 was obtained (Appendix A3.5.4). This improved after Real Estate was added as an X regressor as well, producing an RMSE = 4428.247 (Appendix A3.5.5). The best model was obtained after adding in the % change in US GDP to take care of the smaller fluctuations with RMSE = 4413.066.

```
arima1 <- Arima(seaadj, order=c(1,1,0),
                xreg=cbind(m_lag_split$train, re_lag_split$train,
us_lag_split$train), lambda="auto")
```

```
totalforecast = (forecast(arima1, xreg=cbind(m_lag_split$test,
re_lag_split$test, us_lag_split$test),h = m, lambda=arima1$lambda))$mean
+ (snaive(season, h = m))$mean
autoplot(u_split$train) + autolayer(totalforecast) +
autolayer(u_split$test)
```




```
arima1$coef
checkresiduals(arima1)
accuracy(totalforecast, u_split$test)
```

```
> arima1$coef
               ar1  m_lag_split$train re_lag_split$train us_lag_split$train
     -2.267599e-01       3.924105e-10       6.002882e-10      -1.636022e-08
> checkresiduals(arima1)

        Ljung-Box test

data:  Residuals from Regression with ARIMA(1,1,0) errors
Q* = 0.50211, df = 4, p-value = 0.9733

Model df: 4.   Total lags used: 8

> accuracy(totalforecast, u_split$test)
                ME     RMSE      MAE       MPE     MAPE      ACF1 Theil's U
Test set -2074.986 4413.066 3550.41 -2.542456 4.089698 0.4258793 0.4252494
```

4. VECM

4.1 Variable Selection

Based on our EDA, we have decided to explore VECMs and VARs should VECMs not be suited for our data, with granger-caused variables. We split the series into the training and test sets before model selection and parameter estimation. We will be focusing on Industrials, Healthcare, Consumer Discretionary, Utilities and Information Technology.

4.2 Data Preprocessing

To start off, we check the individual series if they are non-stationary or not. All variables require some degree of differencing to render stationary. We note that Healthcare, Information Technology and Industrials all require the same order of differencing, 1 seasonal and 1 non-seasonal, which allows us to check for cointegration to see if we are able to form a VECM with those Granger Caused Variables.

4.3 Cointegration Tests

Conducting a Johansen Test, we note that we fail to reject the null at r <= 1 at the 5% level of significance, hence we have 1 cointegrating relationship. We therefore can proceed on with building a VECM model using the Granger Caused Variables.

```
> summary(ca.jo(data.frame(it_train,hc_train,in_train)))

######################
# Johansen-Procedure #
######################

Test type: maximal eigenvalue statistic (lambda max) , with linear trend

Eigenvalues (lambda):
[1] 0.2622317 0.0918050 0.0213052

Values of teststatistic and critical values of test:

          test 10pct  5pct  1pct
r <= 2 |  2.91  6.50  8.18 11.65
r <= 1 | 13.00 12.91 14.90 19.19
r = 0  | 41.06 18.90 21.07 25.75

Eigenvectors, normalised to first column:
(These are the cointegration relations)

            it_train.l2 hc_train.l2 in_train.l2
it_train.l2   1.0000000  1.00000000    1.000000
hc_train.l2   0.2845497 -0.77597566    0.324517
in_train.l2  -1.6680556 -0.01626497   -1.107980

Weights W:
(This is the loading matrix)

            it_train.l2  hc_train.l2 in_train.l2
it_train.d -0.04715977 -0.228687606 0.009319877
hc_train.d -0.04413493 -0.004134812 0.003292089
in_train.d -0.03259281 -0.069068366 0.070003331
```

4.4 Model Selection

From here, we build a VECM model using VARselect(), which gave us the optimal lags of 5 based on the SC(n) statistic. From there, we build a VECM model of order 5, using ML estimation and generated our forecast

The Model Parameters are in Appendix 4.4.

4.5 Forecasting and Out of Sample Performance

A quick summary of the OOS Sample Performance based on RMSE, of which, only Healthcare was able to outperform the out-of-sample performance of ARIMA model from above, while the remaining VECM models are slightly underperforming.

GICS INDUSTRY	RMSE
Information Technology	51,731.23
Healthcare	131,423.2
Industrials	44,802.16
Consumer Discretionary	93,238.03
Utilities	17,419.05

5. Conclusion

Through this project, we briefly analysed the seasonality in corporate revenues, before a thorough analysis of forecasting models for 5 industries. We learnt that there is no significant seasonality observed in any industry due to the effect of aggregation, as individual companies in the same industry might still have slightly different seasonality characteristics, which will cancel out with aggregation. However, we did see some seasonality in the Utilities sector, which can be explained by the changes in demand and supply of energy during different seasons.

For the forecasting models, we tried many methods, including SARIMA, ARIMA-X and VECM on the Information Technology, Healthcare, Consumer Discretionary, Financials and Utilities industries. Out of the 5, healthcare is the most well behaved and did not experience large fluctuations due to COVID, which meant that a simple deterministic trend with ARMA(1, 1) model is sufficient in modelling the series. Information Technology, Consumer Discretionary and Utilities are also generally well behaved although COVID did cause some minor fluctuations, which are captured with the addition of X regressors in ARIMA-X. The financial industry is the most difficult to model due to the high volatility in corporate revenue, and only a general trend could be captured with ARIMA-X. We also tried VECMs, but they do not work well as the economic relationships between corporate revenues of these industries are not that strong. Hence, the best models we tried are mostly variations of ARIMA.

All in all, this project definitely helped us understand the trends of corporate revenues in different industries, and also honed our data analysis and forecasting skills. We also enjoyed the process of working through problems together, and are grateful for the experience!

Appendix

A2 EDA

Initial hypothesis

```
#Basic Exploratory Data Analysis, COrrelation Plots (seasonality)
#Autoplots
autoplot(df)
autoplot(cd_)
autoplot(u_)
autoplot(f_)
autoplot(it_)
autoplot(hc_)
ggseasonplot(diff(diff(cs_)))
ggseasonplot(diff(cs_), polar = TRUE)
ggsubseriesplot(diff(cs_))
#Naive and seasonal naive
checkresiduals(naive(diff(u_)))
checkresiduals(snaive(diff(u_)))
#decomposition
autoplot(decompose(u_))
autoplot(decompose(elec, type="multiplicative"))
```

Granger Causality

```
for (i in 1:ncol(traindf)){
  print(colnames(traindf)[i])
  print(ndiffs(diff(traindf[, i])))
}

for (i in 1:ncol(traindf)){
  for (j in 1:ncol(traindf)){
    if (i != j){
      pval = lmtest::grangertest(diff(adjusteddf[,i]),
diff(adjusteddf[,j]))$`Pr(>F)`[2]
      if (pval < 0.1){
        causalitymatrix[i,j] = 1
      }
    }
  }
}

for (i in 1:ncol(traindf)){
  for (j in 1:ncol(traindf)){
    if (i != j){
      pval = Box.test(lm(adjusteddf[,i] ~ adjusteddf[,j])$resid, lag =
12)$p.value
```

```
      if (pval > 0.1){
        cointegrationmatrix[i,j] = 1
      }
    }
  }
}

cointegrationmatrix
causalitymatrix
```

A3.1 Information Technology

A3.1.1 - snaive + drift benchmark

```
snaive_series <- snaive(ts_it_split$train, h=oos)$mean
totalforecast <- rwf(ts_it_split$train, drift=TRUE, h=oos)$mean +
snaive_series - mean(snaive_series)
accuracy(totalforecast, ts_it_split$test)
```

```
                ME     RMSE      MAE      MPE     MAPE      ACF1 Theil's U
Test set 69132.83 97697.53 74086.66 15.35594 17.18131 0.7897738  2.330587
```

A3.1.2 - Custom Functions diffData and diagnostics

```
diffData <- function(data){
  requiredSDiffs <- nsdiffs(data)
  sDiffData <- data
  if (requiredSDiffs!=0){
    for (i in 1:requiredSDiffs) {
      sDiffData <- diff(sDiffData, 4)
    }
    # sDiffData <- diff(data, requiredSDiffs*4)
  } else{
    sDiffData <- data
  }

  requiredDiffs <- ndiffs(sDiffData)
  finalDiffData <- sDiffData
  if (requiredDiffs!=0){
    for (i in 1:requiredDiffs) {
      finalDiffData <- diff(finalDiffData)
    }
    # finalDiffData <- diff(sDiffData, requiredDiffs)
  } else{
    finalDiffData <- sDiffData
```

```
  }
  return(list(finalDiffData, requiredSDiffs, requiredDiffs))
}

diagnostics <- function(data){
  tsdisplay(data)
  return(summary(ur.kpss(data)))
}
```

A3.1.3 - Ljung-Box Test for SARIMA(4,1,4)(1,1,0)

```
# Manual Arima model
model1_it <- Arima(ts_it_split$train, order = c(4,1,4), seasonal =
c(1,1,0), lambda="auto", include.drift = TRUE)

# In-sample evaluation
print("Model 1")
model1_it
checkresiduals(resid(model1_it))
Box.test(resid(model1_it), type="Ljung-Box")
```

```
No drift term fitted as the order of difference is 2 or more.[1] "Model 1"
Series: ts_it_split$train
ARIMA(4,1,4)(1,1,0)[4]
Box Cox transformation: lambda= 0.356793

Coefficients:
        ar1      ar2     ar3      ar4      ma1     ma2      ma3     ma4      sar1
      1.541  -1.1406  0.8595  -0.4359  -1.7864  1.7894  -1.7877  0.8006  -0.2085
s.e.  0.148   0.1724  0.1723   0.0994   0.1467  0.1568   0.1562  0.1406   0.1368

sigma^2 = 6.835:  log likelihood = -316.59
AIC=653.18   AICc=654.99   BIC=682.09

        Box-Ljung test

data:  resid(model1_it)
X-squared = 0.011753, df = 1, p-value = 0.9137
```

A3.1.4 - SARIMA(4,1,4)(1,1,0) Accuracy

```
# Manual arima
accuracy(x=ts_it_split$test, forecast(model1_it, h=oos))
```

```
                    ME      RMSE       MAE        MPE     MAPE      MASE       ACF1 Theil's U
Training set   313.977  5870.958  3010.679  0.3063235 3.022430 0.3009534 -0.2065524        NA
Test set     -5177.764 38659.999 32860.644 -3.3630912 8.999058 3.2848147  0.4954839   1.12032
```

A3.1.5 ARIMA(2,2,1) and ARIMA(2,2,2) AICc

```
> manual_arima_1_it
Series: seasadj_it_split$train
ARIMA(2,2,1)
Box Cox transformation: lambda= 0.1704223

Coefficients:
          ar1     ar2      ma1
      -0.1771  0.2030  -1.0000
s.e.   0.0846  0.0853   0.0203

sigma^2 = 0.08575:  log likelihood = -26.93
AIC=61.85   AICc=62.16   BIC=73.5
> manual_arima_2_it
Series: seasadj_it_split$train
ARIMA(2,2,2)
Box Cox transformation: lambda= 0.1704223

Coefficients:
          ar1     ar2      ma1      ma2
      -0.7016  0.0999  -0.4518  -0.5482
s.e.   0.2390  0.1154   0.2290   0.2284

sigma^2 = 0.085:  log likelihood = -25.93
AIC=61.85   AICc=62.31   BIC=76.42
```

A3.1.6 ARIMA(2,2,1) Ljung-Box Test

```
checkresiduals(manual_arima_1_it)
```

```
        Ljung-Box test

data:  Residuals from ARIMA(2,2,1)
Q* = 3.5584, df = 5, p-value = 0.6146

Model df: 3.   Total lags used: 8
```

A3.1.7 ARIMA(2,2,1) Accuracy

```
# Forecast with stl decomposition
seasadj_it_model <- Arima(seasadj_it_split$train, order=c(2,2,1),
include.drift = TRUE)
seasonal_it_model <- snaive(seasonal_it_split$train, h=oos)
totalforecast <- (forecast(seasadj_it_model, h=oos))$mean +
seasonal_it_model$mean
accuracy(x = ts_it_split$test, totalforecast)
```

```
                ME     RMSE      MAE      MPE     MAPE      ACF1 Theil's U
Test set 26836.2 58779.62 43010.19 4.53069 10.32357 0.8726822  1.396131
```

A3.1.7 ARIMA(2,2,1) ARIMA-X with Real Estate Revenues

```
ts_it <- ts(data=data[2:nrow(data),"Information.Technology"],
start=c(1980, 2), end=c(2022, 4), deltat = 1/4)
oos <- round(length(ts_it)*0.20)
ts_it_split <- ts_split(ts_it, sample.out = oos)

# Real estate industry revenues have the highest correlation IT
ts_re <- ts(data=data[3:nrow(data),"Real.Estate"], start=c(1980, 2),
end=c(2022, 4), deltat = 1/4) # LAGGED
ts_re_split <- ts_split(ts_re, sample.out = oos)

accuracy(x=ts_it_split$test,
         stlf(ts_it_split$train, h=oos, method="arima",
              lambda="auto",
              xreg=ts_re_split$train,
              newxreg=ts_re_split$test))
```

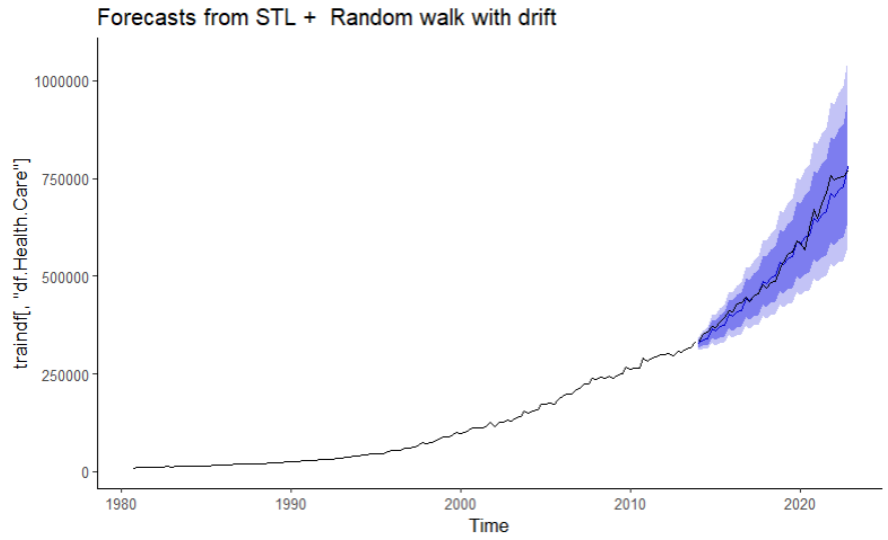
	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1	Theil's U
Training set	120.3894	5514.39	2918.97	-0.1803374	2.879083	0.2933347	-0.2036025	NA
Test set	-26803.0420	37688.75	33438.30	-8.8607140	10.130544	3.3603000	0.5793512	1.295997

A3.2 Healthcare

A3.2.1 - RWF w/ drift

```
# RW w/ drift
mdl1f = stlf(traindf[, "df.Health.Care"], s.window = "periodic", lambda
= "auto", robust = T, method = "rwdrift", h = nrow(testdf))
mdl1f %>% autoplot() + autolayer(testdf[, "df.Health.Care"], color =
"black") + theme_classic() + theme(legend.position = "none")
Box.test(mdl1f$residuals, lag = 12, type = "Ljung-Box")
accuracy(mdl1f, testdf[, "df.Health.Care"])
```

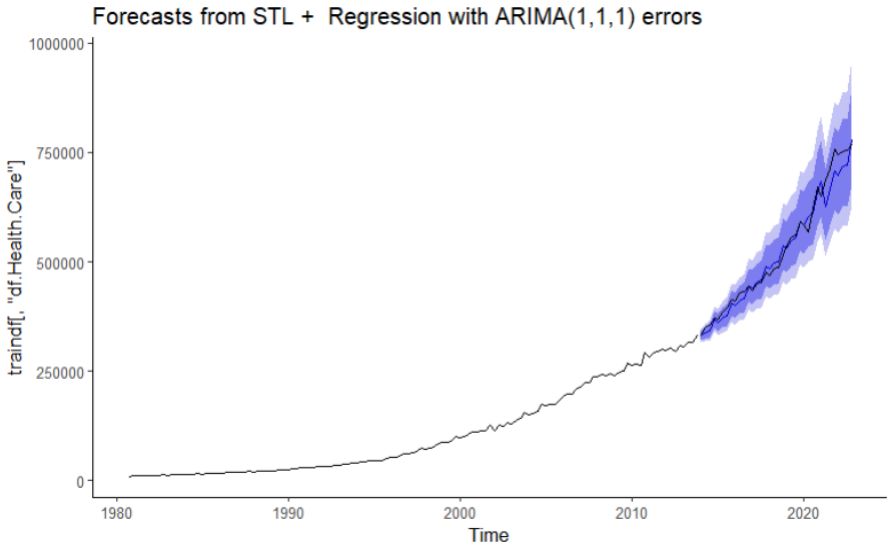
```
        Box-Ljung test

data:  md11f$residuals
X-squared = 11.493, df = 12, p-value = 0.4872

                     ME      RMSE       MAE          MPE      MAPE      MASE        ACF1 Theil's U
Training set -372.3055  3484.941  1993.63 -0.02083508 1.891118 0.2095132 -0.04218628        NA
Test set     9839.5162 20045.671 15805.76  1.74229061 2.872467 1.6610489  0.63940980  0.876119
```



A3.2.2 - ARIMA(1,1,1)

```r
# ARIMA(1,1,1)
md12 = stlm(traindf[, "df.Health.Care"], s.window = "periodic", lambda =
"auto", robust = T, modelfunction = Arima, order = c(1, 1, 1),
include.drift = T)
md12f = forecast(md12,  h = nrow(testdf))
md12f%>% autoplot() + autolayer(testdf[, "df.Health.Care"], color =
"black") + theme_classic() + theme(legend.position = "none")
Box.test(md12$residuals, lag = 12, type = "Ljung-Box")
accuracy(md12f, testdf[, "df.Health.Care"])
```

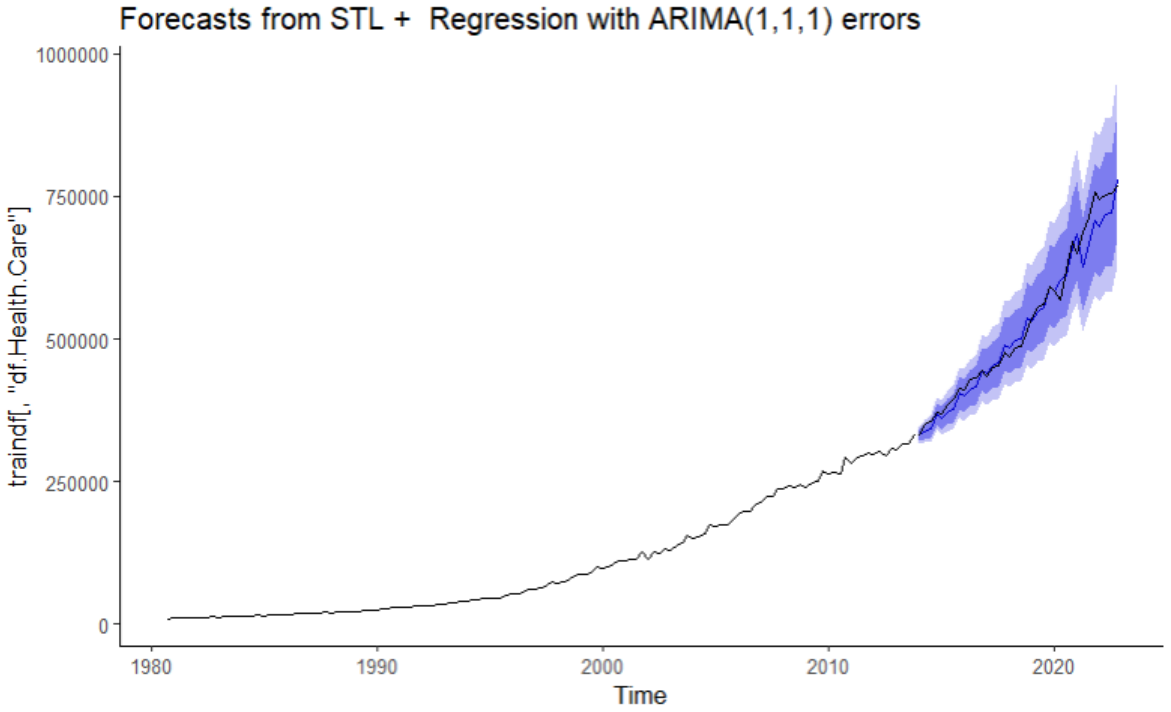
```
        Box-Ljung test

data:  md12$residuals
X-squared = 9.0713, df = 12, p-value = 0.6968

                     ME      RMSE       MAE         MPE      MAPE      MASE      ACF1 Theil's U
Training set -430.1668  3552.032  1989.48 0.01374107 1.851118 0.2090772 0.1910410        NA
Test set     8116.0686 19400.848 14984.82 1.37558031 2.690201 1.5747744 0.6462012 0.8348113
```



A3.2.3 - ARIMA(1,1,1) with external regressors

```
# ARIMA(1,1,1) with xreg
ArimaWxReg = function(x, ...){Arima(x, xreg = traindf[, c("gdpchange",
"popgrowth")], ...)}
# Function definition since xreg is already an argument in stlm

mdl3 = stlm(traindf[, "df.Health.Care"], s.window = "periodic", lambda =
"auto", robust = T, modelfunction = ArimaWxReg, order = c(1, 1, 1),
include.drift = T)
mdl3f = forecast(mdl3,  h = nrow(testdf), newxreg = testdf[,
c("gdpchange", "popgrowth")])
mdl3f %>% autoplot() + autolayer(testdf[, "df.Health.Care"], color =
"black") + theme_classic() + theme(legend.position = "none")
Box.test(mdl3$residuals, lag = 12, type = "Ljung-Box")
accuracy(mdl3f, testdf[, "df.Health.Care"])
```

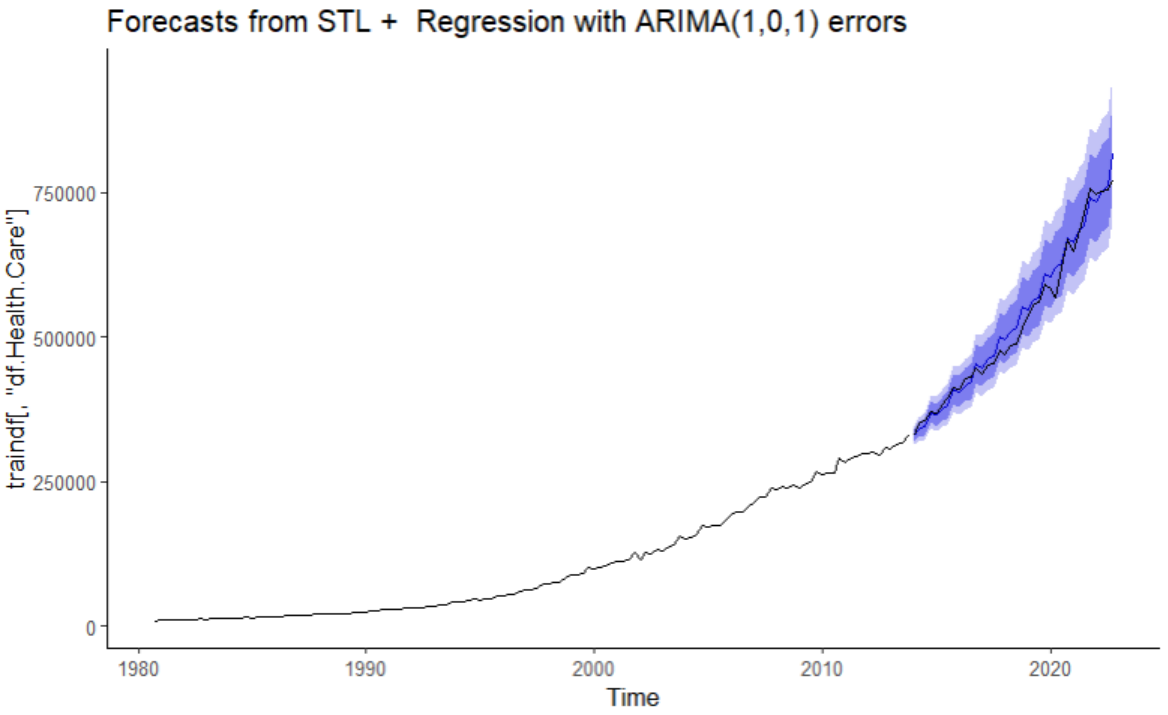
```
        Box-Ljung test

data:  mdl3$residuals
X-squared = 7.6727, df = 12, p-value = 0.8102

                     ME       RMSE        MAE          MPE      MAPE      MASE      ACF1 Theil's U
Training set -411.2877  3430.046  1913.973 0.006290419 1.871249 0.2011419 0.1480747        NA
Test set     7289.9131 22908.453 16724.622 1.194590413 2.941247 1.7576129 0.4471550 0.9483825
```



A3.3. ARMA(1,1) with deterministic linear trend

```
# Trend Estimate w/ ARMA(1,1) instead of Random Walk/Differencing
ArimaWTrend = function(x, ...){Arima(x, xreg = 1:nrow(traindf), ...)}
# Function definition since xreg is already an argument in stlm

mdl4 = stlm(traindf[, "df.Health.Care"], s.window = "periodic", lambda =
"auto", robust = T, modelfunction = ArimaWTrend, order = c(1, 0, 1),
include.constant = T)
mdl4f = forecast(mdl4,  h = nrow(testdf), newxreg =
(nrow(traindf)+1):nrow(dfAggNoVix))
mdl4f %>% autoplot() + autolayer(testdf[, "df.Health.Care"], color =
"black") + theme_classic() + theme(legend.position = "none")
Box.test(mdl4$residuals, lag = 12, type = "Ljung-Box")
accuracy(mdl4f, testdf[, "df.Health.Care"])
```

```
        Box-Ljung test

data:  mdl4$residuals
X-squared = 10.603, df = 12, p-value = 0.5632

                     ME      RMSE       MAE          MPE     MAPE      MASE      ACF1 Theil's U
Training set  -253.0982  3505.977  1925.075  0.02260349 1.831260 0.2023087 0.1821777        NA
Test set     -6770.1323 18546.779 14175.891 -1.14927489 2.701514 1.4897634 0.5615586  0.873205
```



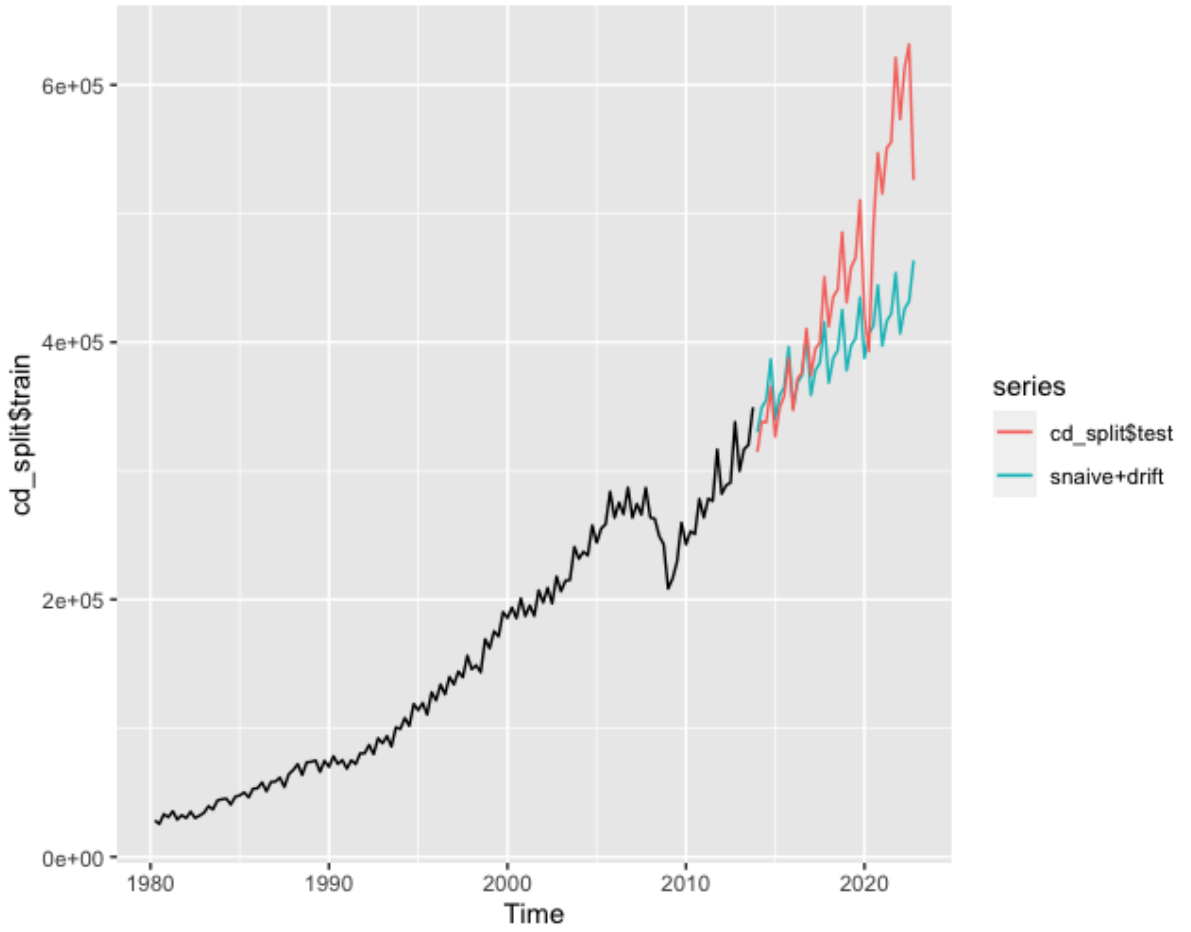
A3.3 Consumer Discretionary

A3.3.1 - snaive + drift benchmark

```
snaive_series <- snaive(cd_split$train, h=m)$mean
totalforecast <- rwf(cd_split$train, drift=TRUE, h=m)$mean +
snaive_series - mean(snaive_series)
accuracy(totalforecast, cd_split$test)
```

```
> accuracy(totalforecast, cd_split$test)
                ME     RMSE      MAE      MPE     MAPE      ACF1 Theil's U
Test set 50177.31 80282.27 56831.23 9.182781 11.08896 0.8597995  1.620818
```

```
autoplot(cd_split$train) + autolayer(totalforecast,
series="snaive+drift") + autolayer(cd_split$test)
```



A3.3.2 - ARIMA (1, 1, 1) model

```
arima1 <- Arima(seaadj, order=c(1,1,1), lambda="auto")
checkresiduals(arima1)
```

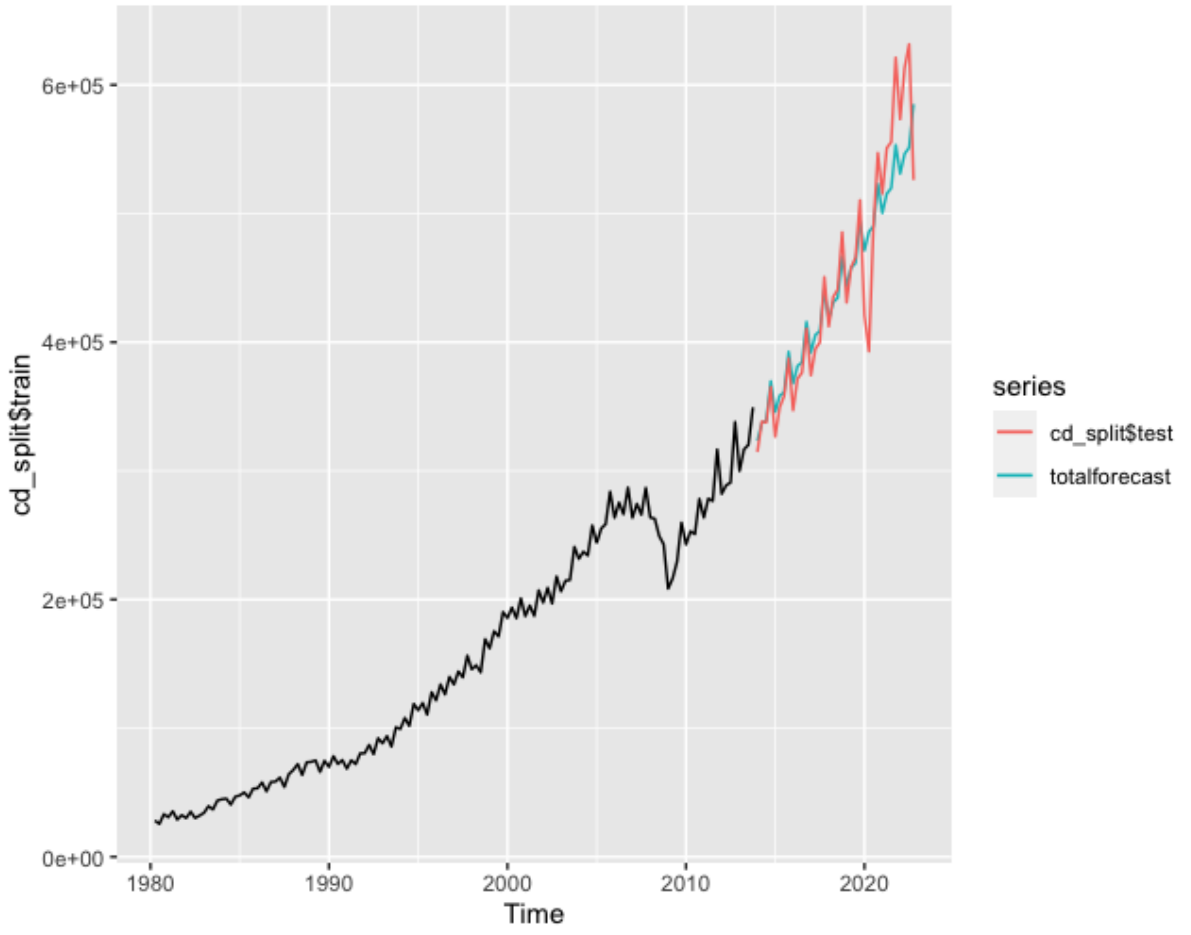
```
> checkresiduals(arima1)

        Ljung-Box test

data:  Residuals from ARIMA(1,1,1)
Q* = 15.877, df = 6, p-value = 0.01443

Model df: 2.   Total lags used: 8
```

A3.3.3 - ARIMA (2, 1, 2) model

```
arima1 <- Arima(seaadj, order=c(2,1,2), lambda="auto", include.drift =
TRUE)
checkresiduals(arima1)
arima1$aicc
```

```
> checkresiduals(arima1)

        Ljung-Box test

data:  Residuals from ARIMA(2,1,2) with drift
Q* = 4.567, df = 3, p-value = 0.2064

Model df: 5.   Total lags used: 8

> arima1$aicc
[1] -47.16217
```

```
totalforecast = (forecast(arima1,h = m, lambda=arima1$lambda))$mean +
(snaive(season, h = m))$mean
autoplot(cd_split$train) + autolayer(totalforecast) +
autolayer(cd_split$test)
```




```
accuracy(model, cd_split$test)
```

```
> accuracy(totalforecast, cd_split$test)
                ME     RMSE      MAE        MPE     MAPE      ACF1 Theil's U
Test set 2104.846 32513.82 21739.21 -0.3583493 4.532432 0.4101034 0.6930361
```

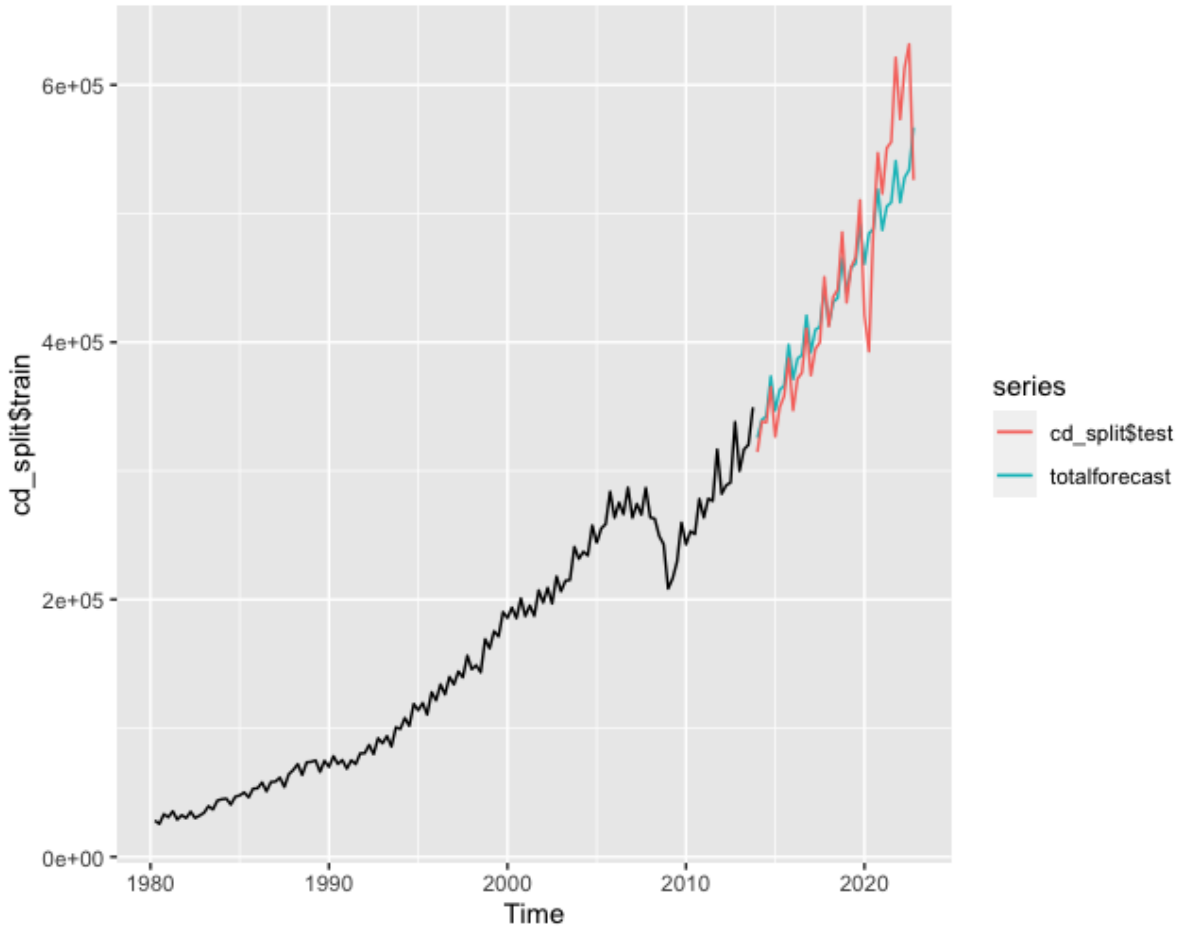
A3.3.4 - ARIMA (3, 1, 2) model

```
arima1 <- Arima(seaadj, order=c(3,1,2), lambda="auto", include.drift =
TRUE)
checkresiduals(arima1)
arima1$aicc
```

```
> checkresiduals(arima1)

        Ljung-Box test

data:  Residuals from ARIMA(3,1,2) with drift
Q* = 5.3759, df = 3, p-value = 0.1463

Model df: 6.   Total lags used: 9

> arima1$aicc
[1] -45.11857
```

A3.3.5 - Information Technology + ARIMA (2, 1, 2) model

```
arima1 <- Arima(seaadj, order=c(2,1,2),
                xreg=cbind(it_lag_split$train), lambda="auto",
include.drift = TRUE)
totalforecast = (forecast(arima1, xreg=cbind(it_lag_split$test),h = m,
lambda=arima1$lambda))$mean + (snaive(season, h = m))$mean
autoplot(cd_split$train) + autolayer(totalforecast) +
autolayer(cd_split$test)
```



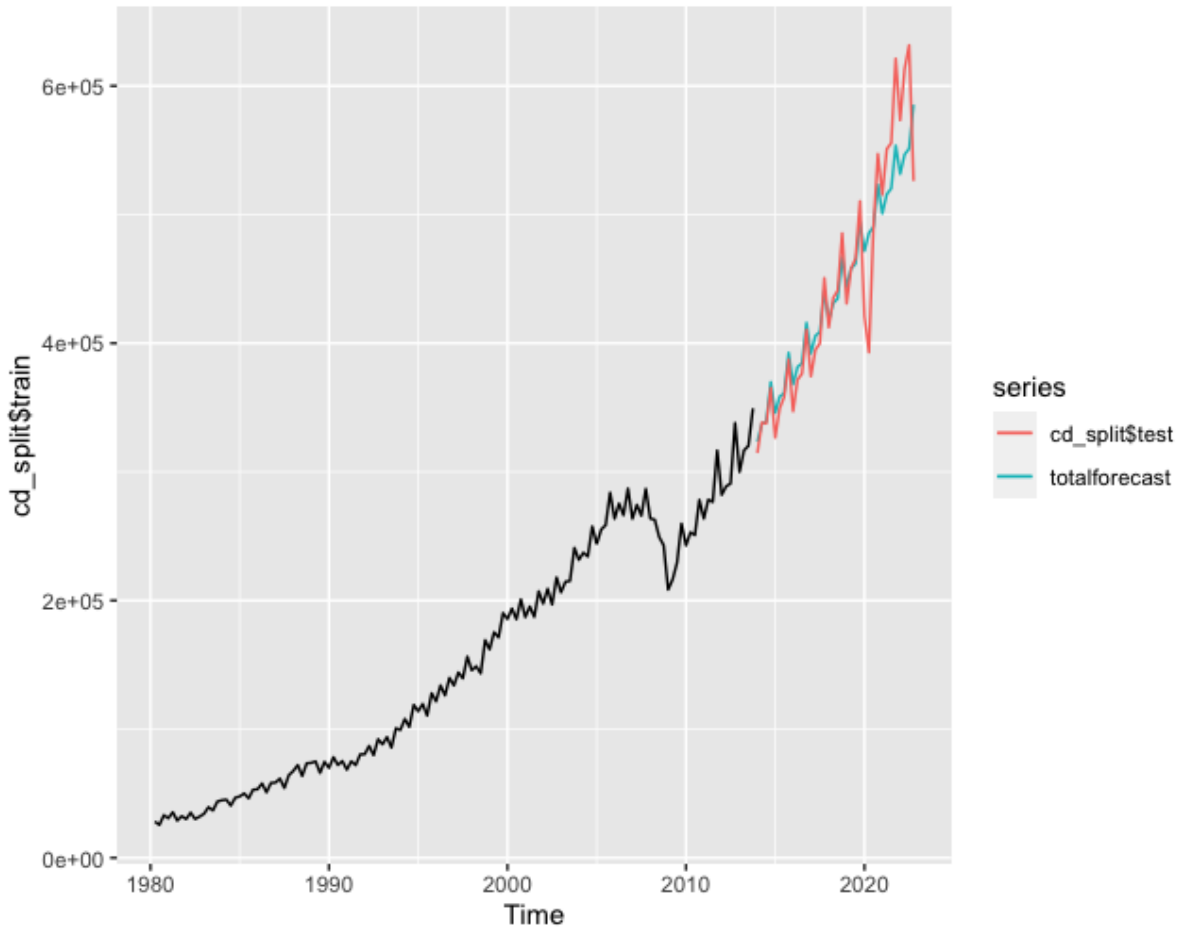

```
arima1$coef
checkresiduals(arima1)
accuracy(totalforecast, cd_split$test)
```

```
> arima1$coef
          ar1           ar2           ma1           ma2         drift          xreg
-1.087229e+00 -6.916208e-01  1.367989e+00  9.999892e-01  1.203708e-01 -1.917787e-06
> checkresiduals(arima1)

        Ljung-Box test

data:  Residuals from Regression with ARIMA(2,1,2) errors
Q* = 4.2275, df = 3, p-value = 0.2379

Model df: 6.   Total lags used: 9

> accuracy(totalforecast, cd_split$test)
                ME     RMSE      MAE          MPE     MAPE      ACF1 Theil's U
Test set 4749.539 36786.94 25213.63 -0.009151501 5.231766 0.5662614 0.7734346
```

A3.3.6 - Financial + ARIMA (2, 1, 2) model

```
arima1 <- Arima(seaadj, order=c(2,1,2),
                xreg=cbind(f_lag_split$train), lambda="auto",
```

```
include.drift = TRUE)
totalforecast = (forecast(arima1, xreg=cbind(f_lag_split$test),h = m,
lambda=arima1$lambda))$mean + (snaive(season, h = m))$mean
autoplot(cd_split$train) + autolayer(totalforecast) +
autolayer(cd_split$test)
```

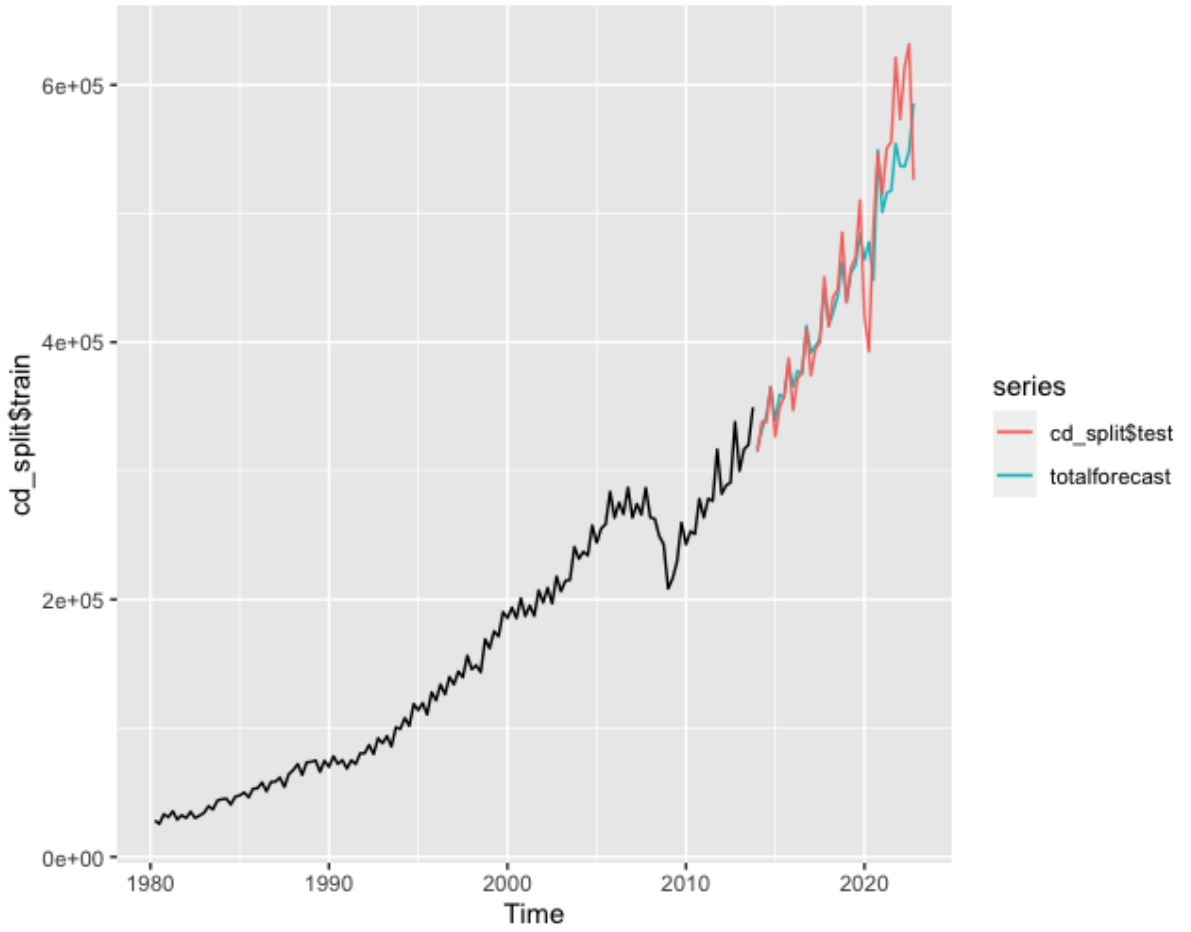



```
arima1$coef
checkresiduals(arima1)
accuracy(totalforecast, cd_split$test)
```

```
> arima1$coef
          ar1           ar2           ma1           ma2         drift          xreg
-1.103427e+00 -7.107183e-01  1.369492e+00  9.999812e-01  1.160001e-01  6.949116e-08
> checkresiduals(arima1)

        Ljung-Box test

data:  Residuals from Regression with ARIMA(2,1,2) errors
Q* = 4.5752, df = 3, p-value = 0.2057

Model df: 6.   Total lags used: 9

> accuracy(totalforecast, cd_split$test)
               ME    RMSE      MAE        MPE     MAPE     ACF1 Theil's U
Test set 1948.116 32438.5 21670.53 -0.3902125 4.521393 0.407072 0.6914903
```

A3.3.7 - % GDP Change + ARIMA (2, 1, 2) model

```
arima1 <- Arima(seaadj, order=c(2,1,2),
                xreg=cbind(us_lag_split$train), lambda="auto",
include.drift = TRUE)
totalforecast = (forecast(arima1, xreg=cbind(us_lag_split$test),h = m,
lambda=arima1$lambda))$mean + (snaive(season, h = m))$mean
autoplot(cd_split$train) + autolayer(totalforecast) +
autolayer(cd_split$test)
```




```
arima1$coef
checkresiduals(arima1)
```

```
> arima1$coef
        ar1         ar2         ma1         ma2       drift        xreg
-0.64689321 -0.97928026  0.74061736  0.90555444  0.11472593  0.01546987
> checkresiduals(arima1)

        Ljung-Box test

data:  Residuals from Regression with ARIMA(2,1,2) errors
Q* = 10.533, df = 3, p-value = 0.01454

Model df: 6.   Total lags used: 9
```

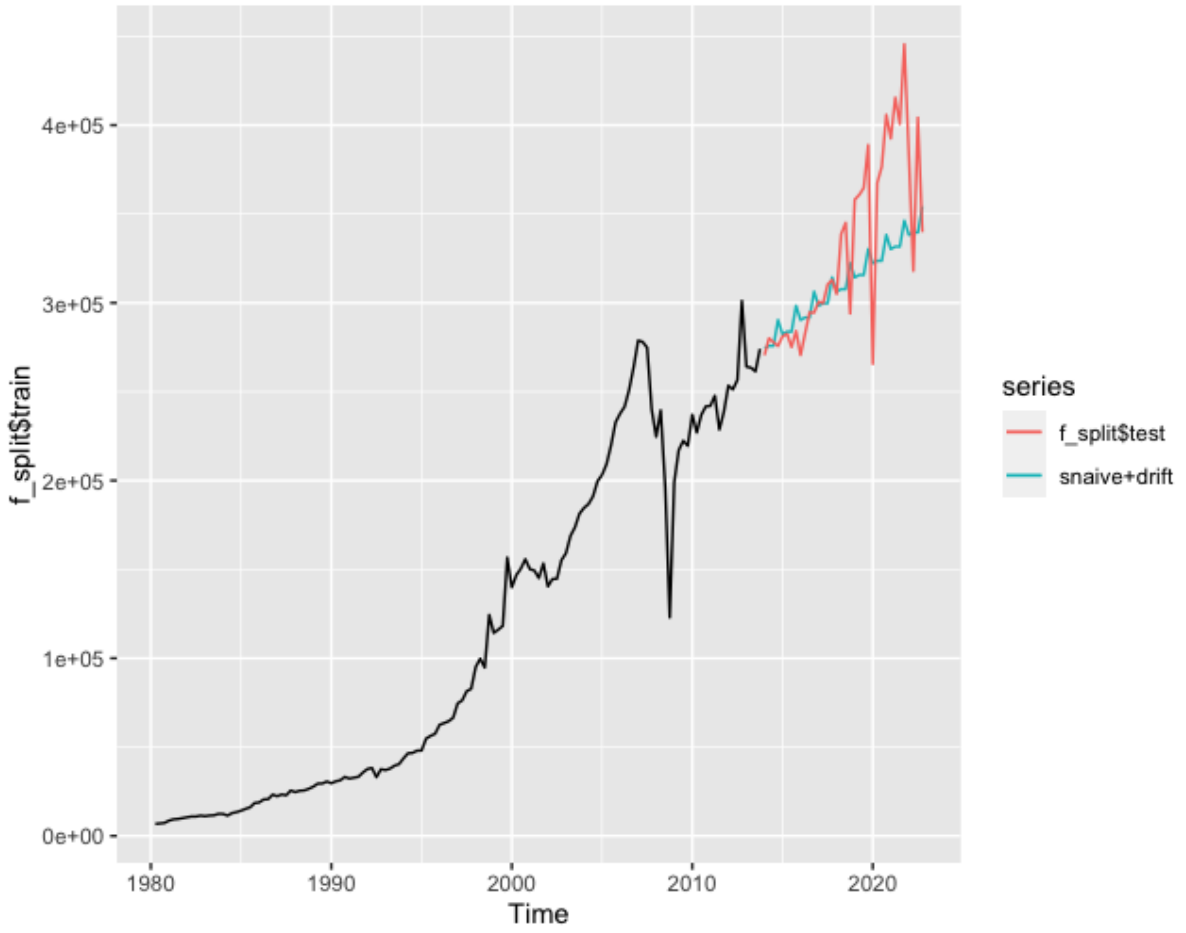
A3.4 Financials

A3.4.1 - snaive + drift benchmark

```
snaive_series <- snaive(f_split$train, h=m)$mean
totalforecast <- rwf(f_split$train, drift=TRUE, h=m)$mean +
snaive_series - mean(snaive_series)
accuracy(totalforecast, f_split$test)
```

```
> accuracy(totalforecast, f_split$test)
                ME     RMSE      MAE      MPE     MAPE      ACF1 Theil's U
Test set 18517.38 40507.86 30090.4 4.258064 8.32447 0.3555433 0.9241653
```

```
autoplot(f_split$train) + autolayer(totalforecast,
series="snaive+drift") + autolayer(f_split$test)
```



A3.4.2 - ARIMA (0, 1, 0) model

```
arima1 <- Arima(seaadj, order=c(0,1,0), lambda="auto", include.drift =
TRUE)
checkresiduals(arima1)
arima1$coef
arima1$aicc
```

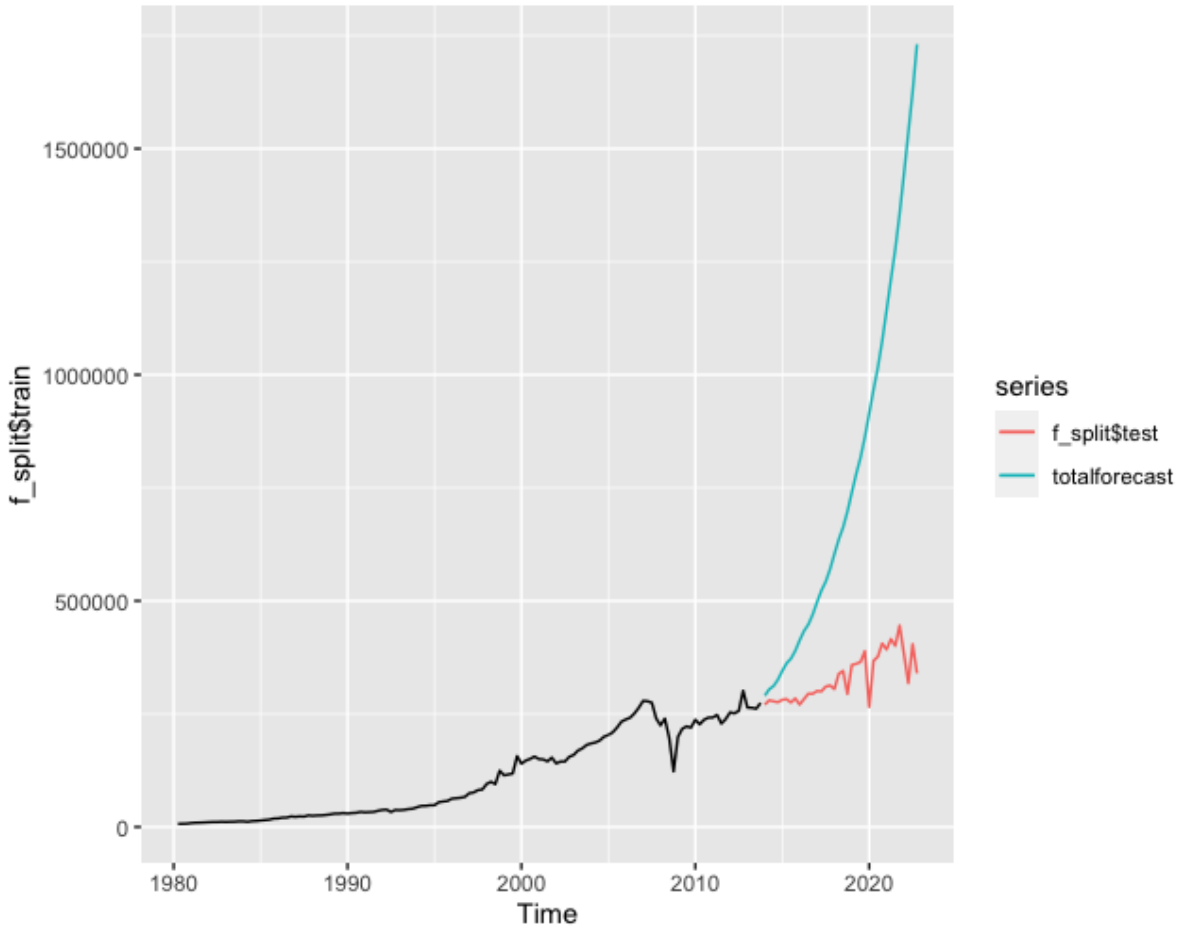
```
> checkresiduals(arima1)

        Ljung-Box test

data:  Residuals from ARIMA(0,1,0) with drift
Q* = 10.155, df = 7, p-value = 0.1799

Model df: 1.   Total lags used: 8

> arima1$coef
      drift
0.002833585
> arima1$aicc
[1] -959.2007
```

A3.4.3 - ARIMA (1, 1, 0) model

```
arima1 <- Arima(seaadj, order=c(1,1,0), lambda="auto", include.drift = TRUE)
checkresiduals(arima1)
arima1$coef
arima1$aic
```

```
> checkresiduals(arima1)

        Ljung-Box test

data:  Residuals from ARIMA(1,1,0) with drift
Q* = 8.7021, df = 6, p-value = 0.191

Model df: 2.   Total lags used: 8

> arima1$coef
         ar1        drift
-0.146318483  0.002827439
> arima1$aic
[1] -960.1954
```

```
totalforecast = (forecast(arima1,h = m, lambda=arima1$lambda))$mean + (snaive(season, h = m))$mean
autoplot(f_split$train) + autolayer(totalforecast) + autolayer(f_split$test)
```




```
accuracy(totalforecast, f_split$test)
```

```
> accuracy(totalforecast, f_split$test)
                ME     RMSE    MAE       MPE     MAPE      ACF1 Theil's U
Test set -439774 577451.5 439774 -124.6559 124.6559 0.8782897   13.0318
```

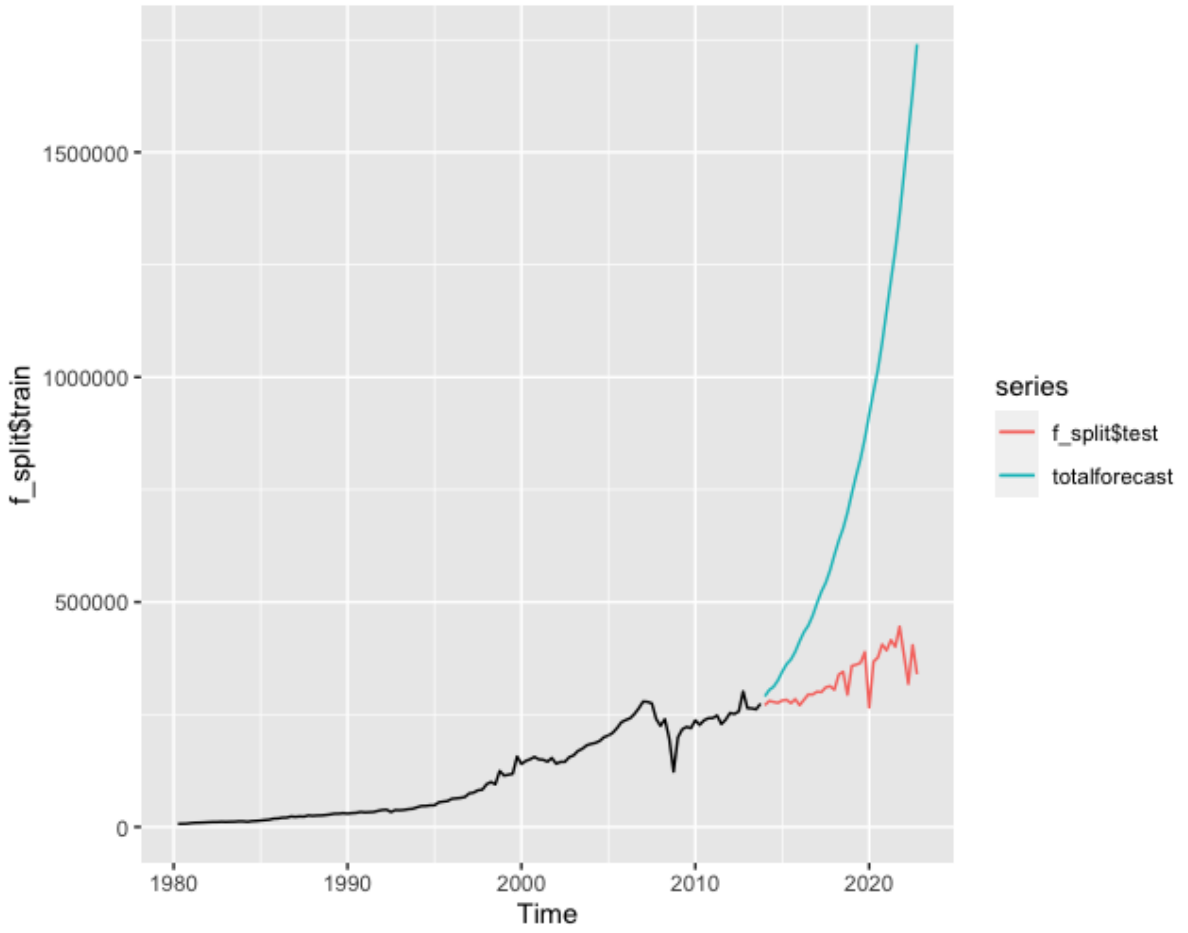
A3.4.4 - ARIMA (2, 1, 0) model

```
arima1 <- Arima(seaadj, order=c(2,1,0), lambda="auto", include.drift =
TRUE)
checkresiduals(arima1)
arima1$coef
arima1$aic
```

```
> checkresiduals(arima1)

        Ljung-Box test

data:  Residuals from ARIMA(2,1,0) with drift
Q* = 8.4986, df = 5, p-value = 0.1308

Model df: 3.   Total lags used: 8

> arima1$coef
         ar1          ar2        drift
-0.141060953  0.037430624  0.002826682
> arima1$aic
[1] -958.3833
```

A3.4.5 - Consumer Discretionary + ARIMA (1, 1, 0) model

```
arima1 <- Arima(seaadj, order=c(1,1,0),
                xreg=cbind(cd_lag_split$train), lambda="auto",
include.drift = TRUE)
totalforecast = (forecast(arima1, xreg=cbind(cd_lag_split$test),h = m,
lambda=arima1$lambda))$mean + (snaive(season, h = m))$mean
autoplot(f_split$train) + autolayer(totalforecast) +
autolayer(f_split$test)
```

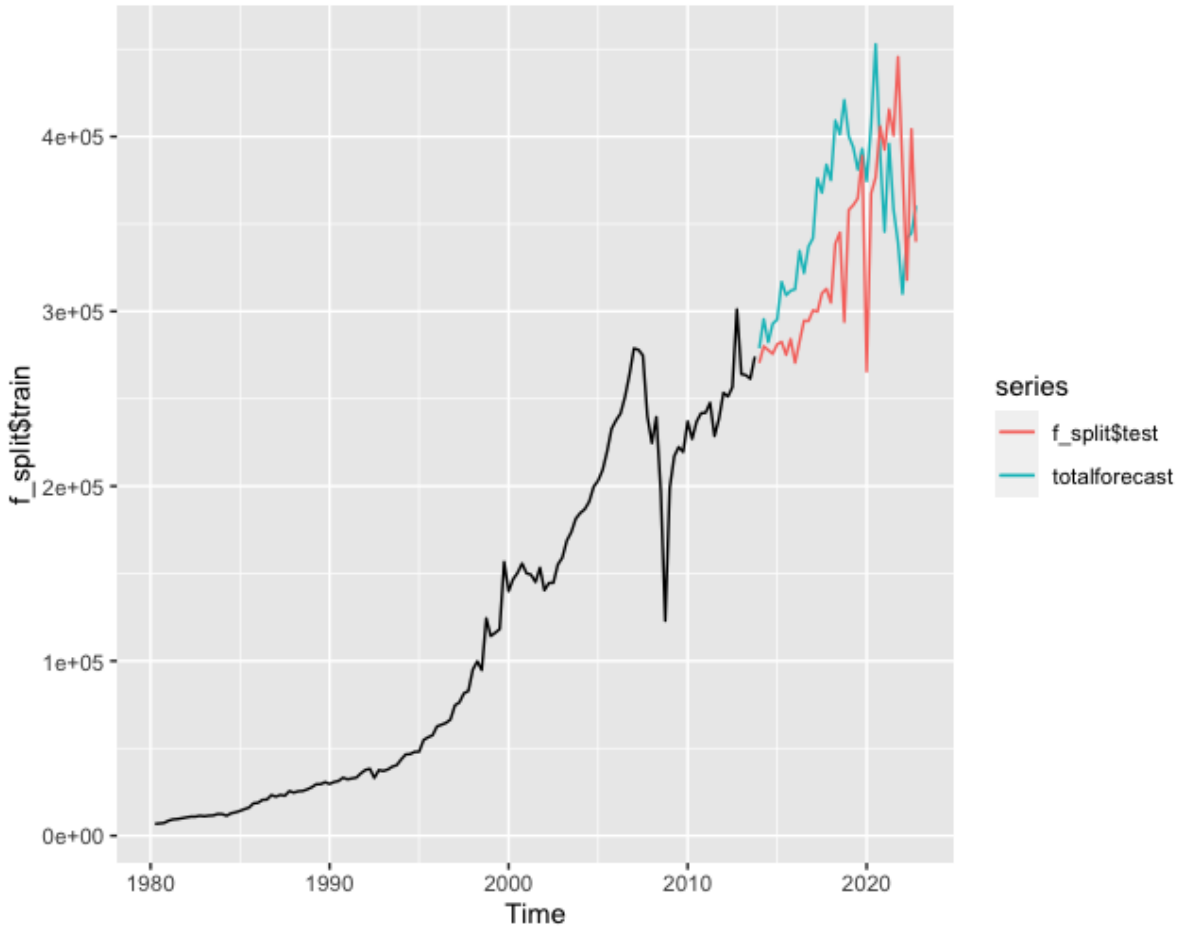


```
checkresiduals(arima1)
accuracy(totalforecast, f_split$test)
```

```
> checkresiduals(arima1)

        Ljung-Box test

data:  Residuals from Regression with ARIMA(1,1,0) errors
Q* = 8.6923, df = 5, p-value = 0.122

Model df: 3.   Total lags used: 8

> accuracy(totalforecast, f_split$test)
                ME     RMSE      MAE       MPE     MAPE      ACF1 Theil's U
Test set -441586.9 580321.8 441586.9 -125.1557 125.1557 0.8780662  13.09374
```

A3.4.6 - Healthcare + ARIMA (0, 1, 0) model

```
arima1 <- Arima(seaadj, order=c(1,1,0),
                xreg=cbind(hc_lag_split$train), lambda="auto",
include.drift = TRUE)
totalforecast = (forecast(arima1, xreg=cbind(hc_lag_split$test),h = m,
lambda=arima1$lambda))$mean + (snaive(season, h = m))$mean
autoplot(f_split$train) + autolayer(totalforecast) +
autolayer(f_split$test)
```




```
checkresiduals(arima1)
accuracy(totalforecast, f_split$test)
```

```
> checkresiduals(arima1)

        Ljung-Box test

data:  Residuals from Regression with ARIMA(1,1,0) errors
Q* = 7.7006, df = 5, p-value = 0.1735

Model df: 3.   Total lags used: 8

> accuracy(totalforecast, f_split$test)
                 ME     RMSE      MAE       MPE     MAPE      ACF1 Theil's U
Test set -24757.35 53763.04 44932.01 -8.901814 13.82993 0.5576287    1.3318
```

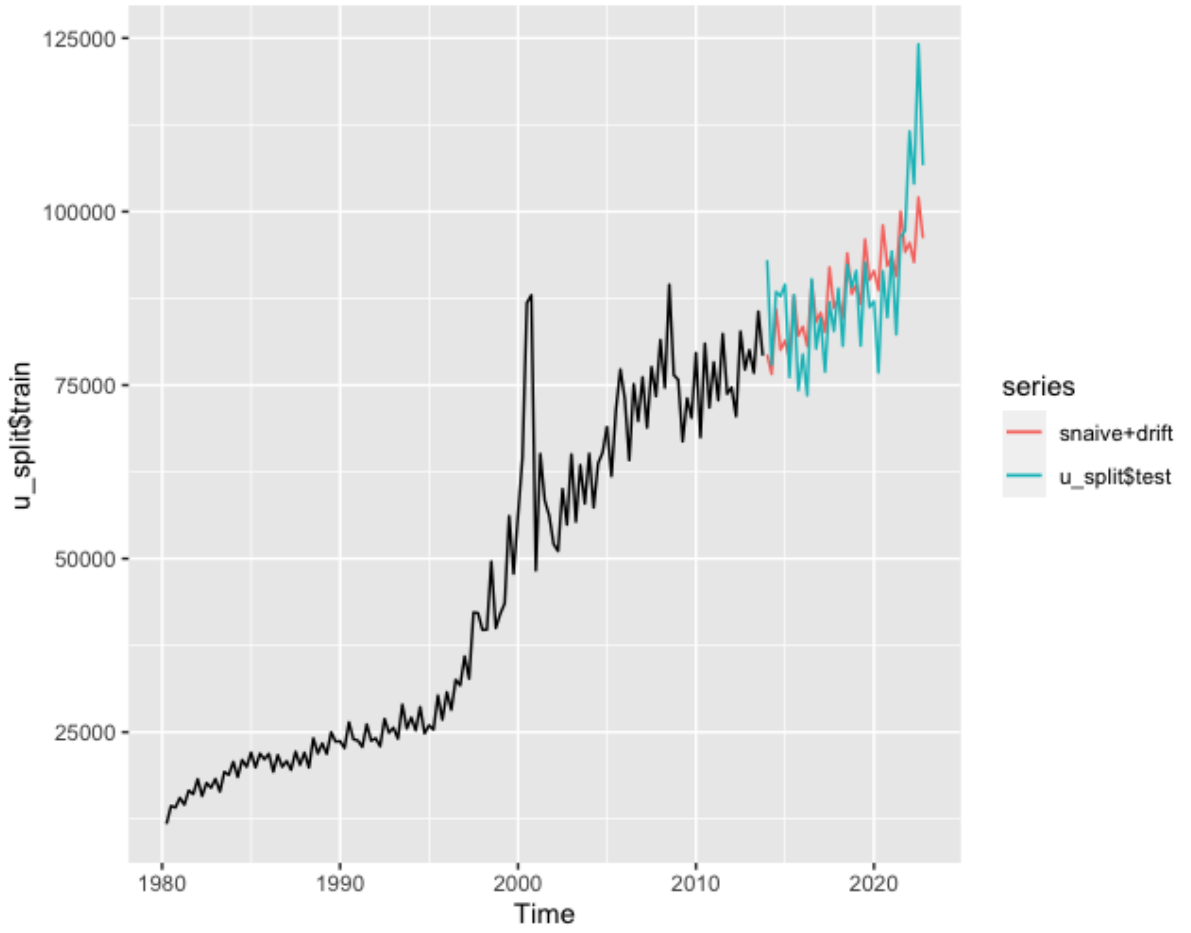
A3.5 Utilities

A3.5.1 - snaive + drift benchmark

```
snaive_series <- snaive(u_split$train, h=m)$mean
totalforecast <- rwf(u_split$train, drift=TRUE, h=m)$mean +
snaive_series - mean(snaive_series)
accuracy(totalforecast, u_split$test)
```

```
> accuracy(totalforecast, u_split$test)
                ME     RMSE      MAE        MPE     MAPE      ACF1 Theil's U
Test set 9.114021 7380.499 5646.056 -0.7006745 6.209274 0.5742129 0.6294915
```

```
autoplot(u_split$train) + autolayer(totalforecast,
series="snaive+drift") + autolayer(u_split$test)
```

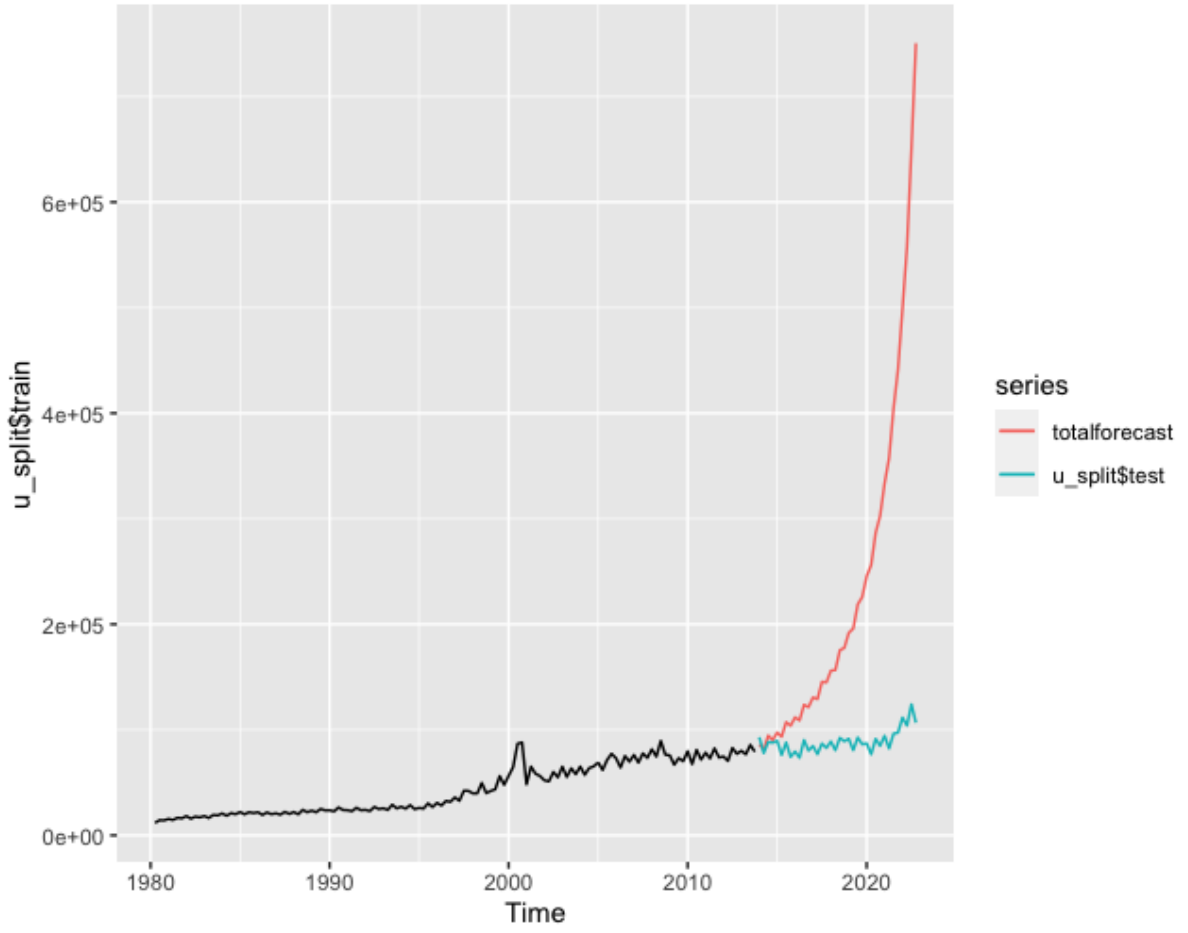


A3.5.2 - ARIMA (1, 1, 1) model

```
arima1 <- Arima(seaadj, order=c(1,1,1), lambda="auto", include.drift =
TRUE)
checkresiduals(arima1)
arima1$coef
arima1$aicc
```

```
> checkresiduals(arima1)

        Ljung-Box test

data:  Residuals from ARIMA(1,1,1) with drift
Q* = 0.49244, df = 5, p-value = 0.9924

Model df: 3.   Total lags used: 8

> arima1$coef
          ar1           ma1         drift
-3.950400e-01  1.103364e-01  5.065342e-06
> arima1$aicc
[1] -2532.444
```

```
totalforecast = (forecast(arima1,h = m, lambda=arima1$lambda))$mean +
(snaive(season, h = m))$mean
autoplot(u_split$train) + autolayer(totalforecast) +
autolayer(u_split$test)
```



A3.5.3 - ARIMA (1, 1, 0) model

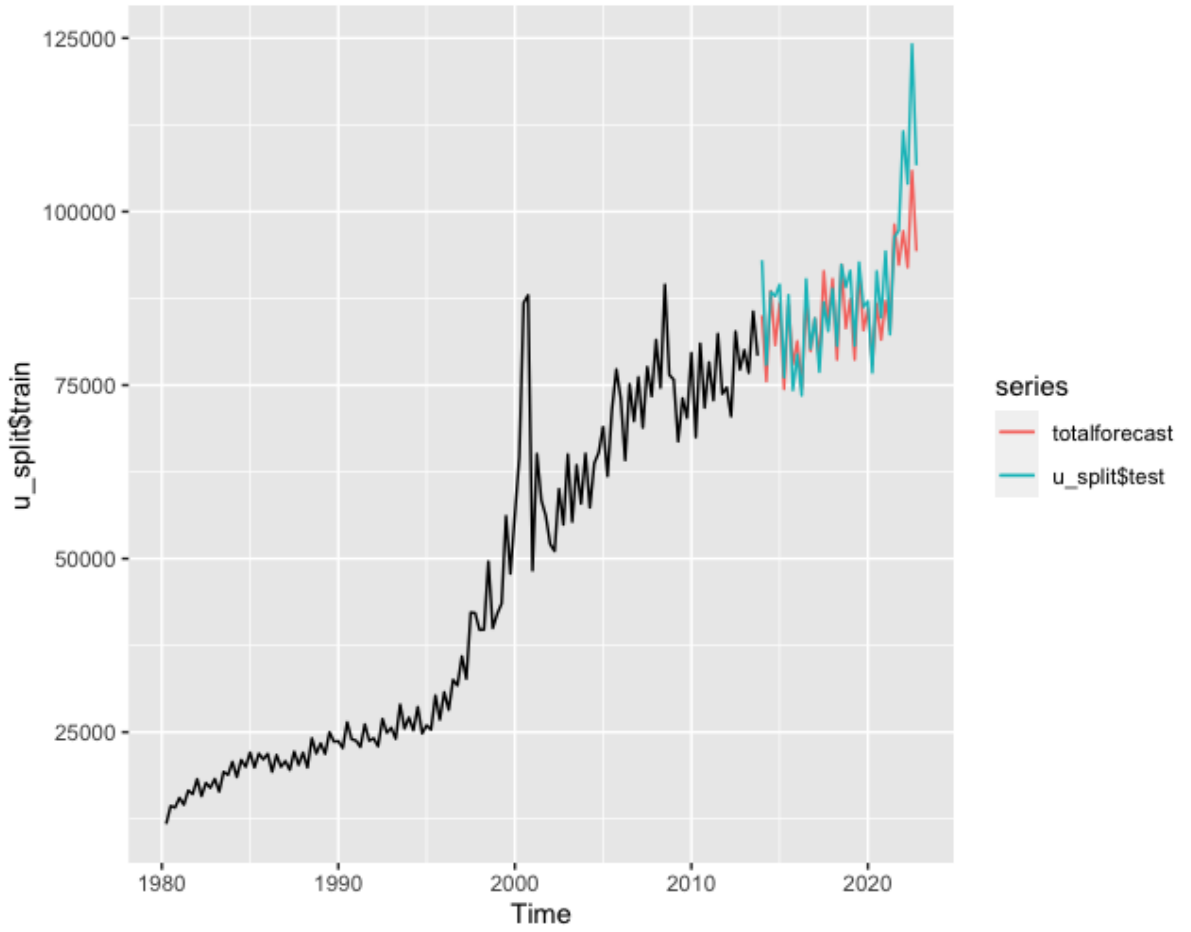
```
arima1 <- Arima(seaadj, order=c(1,1,0), lambda="auto", include.drift =
TRUE)
```

```
checkresiduals(arima1)
arima1$coef
arima1$aicc
```

```
> checkresiduals(arima1)

        Ljung-Box test

data:  Residuals from ARIMA(1,1,0) with drift
Q* = 0.5128, df = 6, p-value = 0.9977

Model df: 2.   Total lags used: 8

> arima1$coef
          ar1         drift
-2.934095e-01  5.057882e-06
> arima1$aicc
[1] -2534.286
```

A3.5.4 - Materials + ARIMA (1, 1, 0) model

```
arima1 <- Arima(seaadj, order=c(1,1,0),
                xreg=cbind(m_lag_split$train), lambda="auto")
totalforecast = (forecast(arima1, xreg=cbind(m_lag_split$test),h = m,
lambda=arima1$lambda))$mean + (snaive(season, h = m))$mean
autoplot(u_split$train) + autolayer(totalforecast) +
autolayer(u_split$test)
```

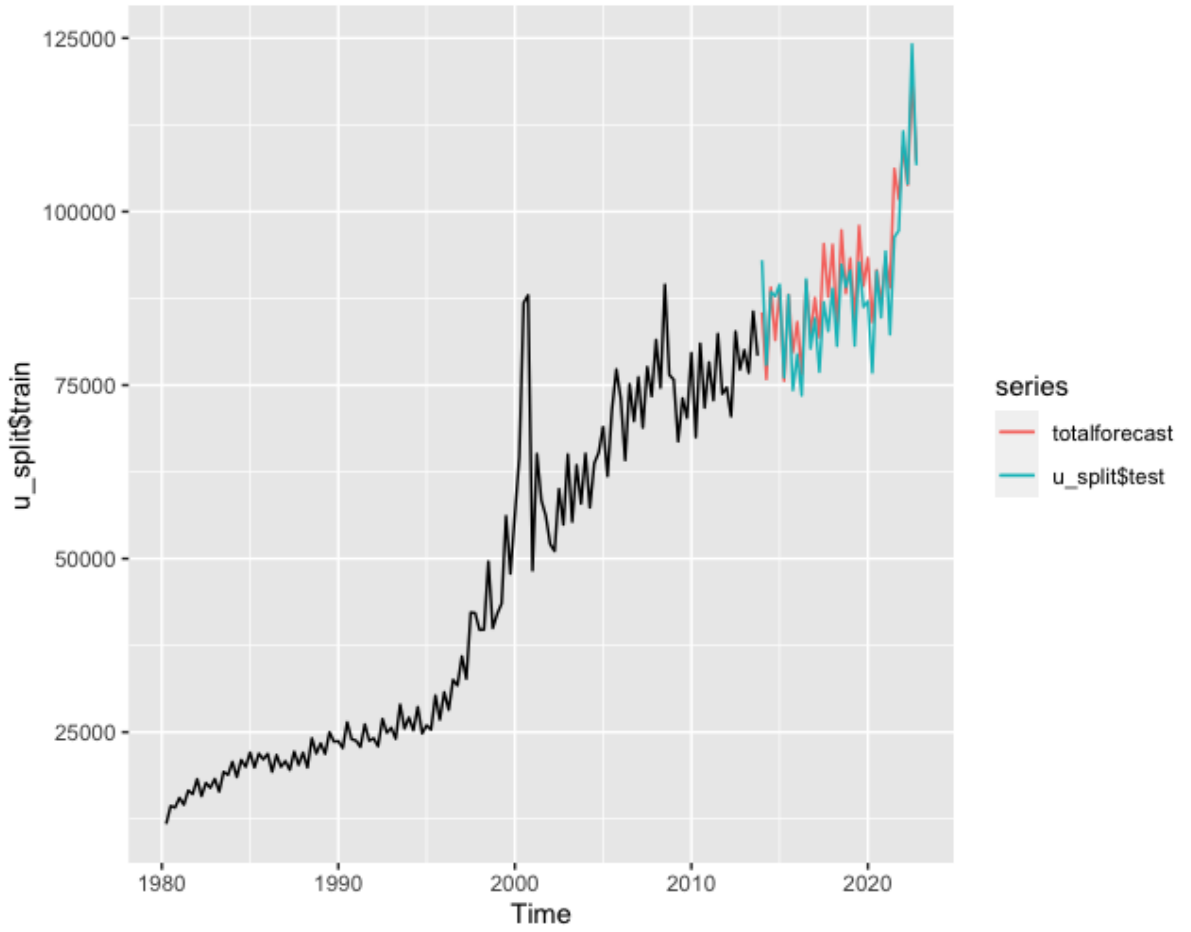



```
arima1$coef
checkresiduals(arima1)
accuracy(totalforecast, u_split$test)
```

```
> accuracy(totalforecast, u_split$test)
                ME     RMSE      MAE       MPE    MAPE      ACF1 Theil's U
Test set 2894.049 5790.622 3936.564 2.827399 4.10584 0.6137251 0.4827079
```

A3.5.5 - Materials + Real Estate + ARIMA (1, 1, 0) model

```
arima1 <- Arima(seaadj, order=c(1,1,0),
                xreg=cbind(m_lag_split$train, re_lag_split$train),
lambda="auto")
totalforecast = (forecast(arima1, xreg=cbind(m_lag_split$test,
re_lag_split$test),h = m, lambda=arima1$lambda))$mean + (snaive(season,
h = m))$mean
autoplot(u_split$train) + autolayer(totalforecast) +
autolayer(u_split$test)
```




```
arima1$coef
checkresiduals(arima1)
accuracy(totalforecast, u_split$test)
```

```
> arima1$coef
              ar1  m_lag_split$train re_lag_split$train
    -2.269070e-01       3.909263e-10       6.040535e-10
> checkresiduals(arima1)

        Ljung-Box test

data:  Residuals from Regression with ARIMA(1,1,0) errors
Q* = 0.5018, df = 5, p-value = 0.9921

Model df: 3.   Total lags used: 8

> accuracy(totalforecast, u_split$test)
                ME     RMSE      MAE       MPE     MAPE      ACF1 Theil's U
Test set -2099.031 4428.247 3558.994 -2.567941 4.101357 0.4219272 0.4268195
```

A4 VECM

A4.2 Data Preprocessing

```
> nsdiffs(in_train)
[1] 1
> ndiffs(diff(in_train,4))
[1] 1
> in_train_diff = diff(diff(in_train,4))
> nsdiffs(hc_train)
[1] 1
> ndiffs(diff(hc_train,4))
[1] 1
> hc_train_diff = diff(diff(hc_train,4))
> nsdiffs(cd_train)
[1] 1
> ndiffs(diff(cd_train,4))
[1] 0
> cd_train_diff = diff(cd_train,4)
> nsdiffs(u_train)
[1] 0
> ndiffs(u_train)
[1] 1
> u_train_diff = diff(u_train)
> nsdiffs(it_train)
[1] 1
> ndiffs(diff(it_train,4))
[1] 1
> it_train_diff = diff(diff(it_train,4))
>
```

A4.3 Cointegration Test

```
> summary(ca.jo(data.frame(it_train,hc_train,in_train)))

######################
# Johansen-Procedure #
######################

Test type: maximal eigenvalue statistic (lambda max) , with linear trend

Eigenvalues (lambda):
[1] 0.2622317 0.0918050 0.0213052

Values of teststatistic and critical values of test:

          test 10pct  5pct  1pct
r <= 2 |  2.91  6.50  8.18 11.65
r <= 1 | 13.00 12.91 14.90 19.19
r = 0  | 41.06 18.90 21.07 25.75

Eigenvectors, normalised to first column:
(These are the cointegration relations)

            it_train.l2 hc_train.l2 in_train.l2
it_train.l2   1.0000000  1.00000000    1.000000
hc_train.l2   0.2845497 -0.77597566    0.324517
in_train.l2  -1.6680556 -0.01626497   -1.107980

Weights W:
(This is the loading matrix)

            it_train.l2  hc_train.l2 in_train.l2
it_train.d -0.04715977 -0.228687606 0.009319877
hc_train.d -0.04413493 -0.004134812 0.003292089
in_train.d -0.03259281 -0.069068366 0.070003331

> |
```

A3.4 Model Parameters

```
> vecmodel
                          ECT  Intercept it_train -1 hc_train -1 in_train -1 cd_train -1
Equation it_train -0.0001155712   393.6007 -0.78788683  0.35381786  0.22843970  0.42778219
Equation hc_train  0.0100447927 -2094.5016  0.13376810 -0.46824744  0.02479535  0.04790779
Equation in_train -0.0111766014  2266.4383  0.04097489  0.46445481 -0.22649851  0.30559575
Equation cd_train -0.0026105932  1934.3421 -0.35715318  0.44081975 -0.02535171  0.28546122
Equation u_train   0.0007989047   263.3187  0.01264632 -0.05298945  0.18141002 -0.06217514
                   u_train -1  it_train -2 hc_train -2 in_train -2 cd_train -2   u_train -2
Equation it_train  0.20729544 -0.107181552   0.2134962  0.01098233 -0.03611479  0.027353294
Equation hc_train -0.03581552  0.253469379  -0.2169006 -0.10156682 -0.01750367 -0.037635088
Equation in_train  0.02024274 -0.031812455   0.7415374 -0.35843324  0.43259026  0.037956468
Equation cd_train -0.04409268  0.046897015   0.3256483 -0.28142881  0.09158105  0.001692262
Equation u_train  -0.46464264 -0.000258426  -0.1852523  0.04405766  0.04436798 -0.283770624
                  it_train -3 hc_train -3 in_train -3 cd_train -3  u_train -3 it_train -4 hc_train -4
Equation it_train  0.07420670  0.20524353  -0.5692940  0.15137126 -0.17152551  0.50506201  0.13506422
Equation hc_train  0.02982926 -0.17582538  -0.2134249 -0.05066648 -0.11105843  0.13523395  0.04490451
Equation in_train  0.01664342  0.49384314  -0.4125969  0.18609727  0.00161805  0.06849762  0.37840752
Equation cd_train  0.01979010  0.02831916  -0.5965411 -0.01750278 -0.19335576  0.10462488  0.30246198
Equation u_train   0.15989418  0.13749699  -0.1451463  0.11992135 -0.20605854  0.16686285 -0.03917012
                   in_train -4 cd_train -4   u_train -4 it_train -5 hc_train -5 in_train -5
Equation it_train -0.122316760  0.21072186 -0.224325504  0.11721387  0.20603616 -0.18677121
Equation hc_train  0.002490368  0.01535082  0.069063513  0.02601033  0.10526777  0.03084774
Equation in_train  0.362070715  0.23108558 -0.006179245 -0.28132797  0.29119107  0.24945129
Equation cd_train -0.209637403  0.80909919 -0.102592037  0.09348041 -0.02509135  0.09884947
Equation u_train  -0.074786033 -0.07409235 -0.038734200  0.18657990 -0.05729075 -0.21339784
                  cd_train -5   u_train -5
Equation it_train -0.19488106 -0.146192412
Equation hc_train -0.10044010  0.061750152
Equation in_train -0.13012784  0.075977548
Equation cd_train -0.24092886 -0.068111203
Equation u_train   0.09131627 -0.008316382
>
```

A4.5 Forecast and OOS Tests

```
Equation u_train   0.09131627 -0.008316382
> accuracy(x = it_lag_split$test,fcastts[,1])
                ME      RMSE       MAE        MPE      MAPE       ACF1 Theil's U
Test set 9702.944 51731.23 38351.72 -0.0788492 9.905257 0.7484004  1.388264
> accuracy(x = hc_lag_split$test, fcastts[,2])
                ME      RMSE       MAE       MPE      MAPE       ACF1 Theil's U
Test set 105634.5 131423.2 105634.5 17.82817 17.82817 0.9113264  5.289972
> accuracy(x = in_lag_split$test, fcastts[,3])
                ME      RMSE       MAE        MPE      MAPE       ACF1 Theil's U
Test set -34688.51 44802.16 36076.52 -9.381449 9.704467 0.6261421  1.929548
> accuracy(x = cd_lag_split$test, fcastts[,4])
                ME      RMSE       MAE       MPE      MAPE       ACF1 Theil's U
Test set 70301.14 93238.03 70351.43 13.90105 13.91645 0.8274438  1.995288
> accuracy(x = u_lag_split$test, fcastts[,5])
                ME      RMSE       MAE        MPE      MAPE       ACF1 Theil's U
Test set -15355.92 17419.05 15679.44 -18.22074 18.54002 0.2960693  1.760249
>
```