The Business of Eating: Forecasting Trends in Victoria's Food Industry

Kevin Tran

2024-01-26

Contents

1 Introduction

This project embarks on an analytical journey to forecast the future of food service turnovers in Victoria Australia. I will be using Seasonal Naive, Exponential Smoothing (ETS) and AutoRegressive Integrated Moving Average (ARIMA) forecasting models to provide a comprehensive outlook on the movement in food service sales.

The objective is to offer valuable insights of the food-service market in Victoria, and to demonstrate the efficiency of forecasting technique on how they contribute to economical planning and decision-making. As Australia (Victoria included) continues to navigate through the post-COVID era and the rise of cost-of-living crisis, I hope this forecasting demonstration sheds light of the food service industry.

Data source: https://explore.data.abs.gov.au/vis?tm=turnover&pg=0&df%5Bds%5D=INDUSTRY_TOPICS&df%5Bid%5D=RT&df%5Bag%5D=ABS&df%5Bvs%5D=1.0.0&pd=2022-07%2C&dq=.20%2B41...&ly%5Bcl%5D=INDUSTRY&ly%5Brw%5D=TIME_PERIOD&ly%5Brs%5D=MEASURE%2CTSEST

The dataset contains the following variables:

variable	class	description
TIME_PERIOD	character	month and year of observation ranging from 1981 - 2024
OBS_VALUE_VIC	double	observed turnover value for VIC
OBS_VALUE_NSW	double	observed turnover value for NSW

```
fs_turnover <- read_csv("vic-food-turnover.csv") %>%
  rename(Turnover = OBS_VALUE_VIC, Month = TIME_PERIOD) %>%
  dplyr::select(Month, Turnover) %>%
  mutate(Month = yearmonth(Month)) %>%
  as_tsibble(index = Month)
```

```
## Rows: 500 Columns: 3
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): TIME_PERIOD
## dbl (2): OBS_VALUE_VIC, OBS_VALUE_NSW
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

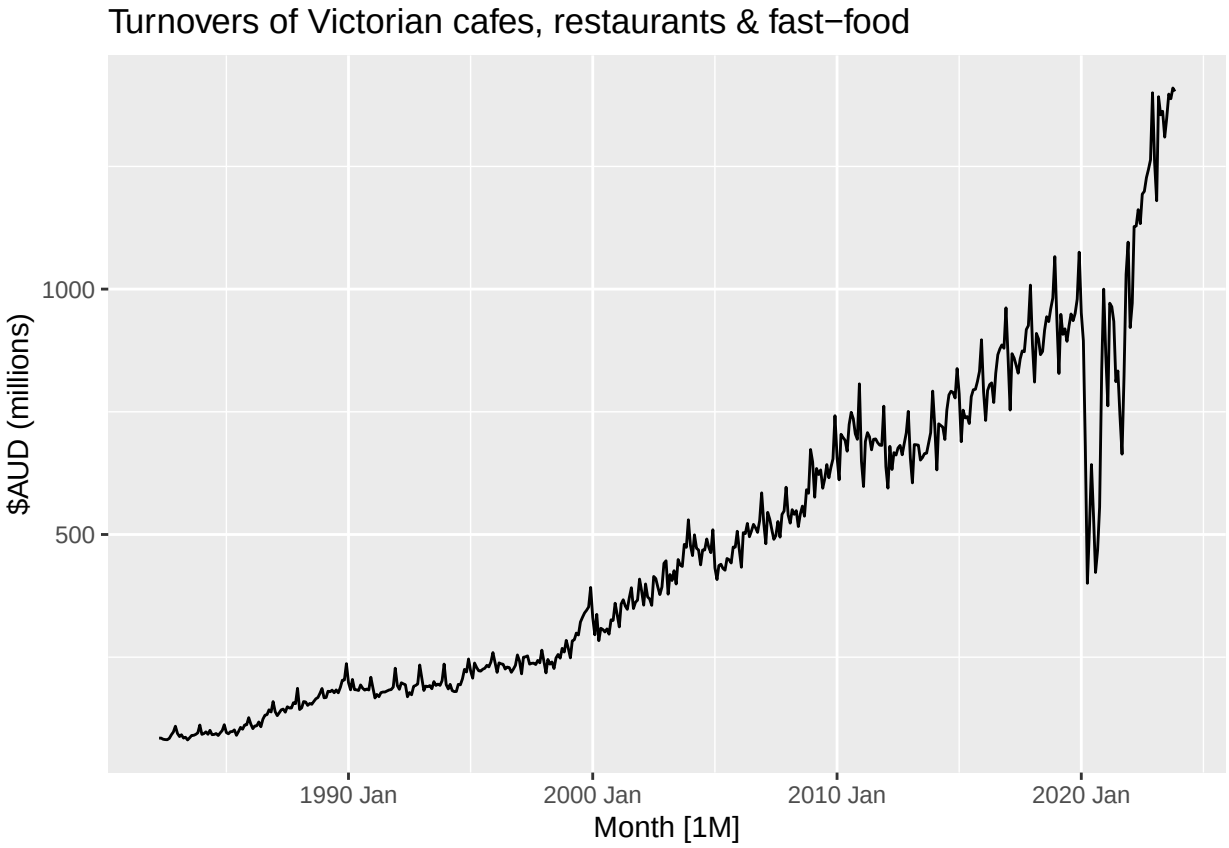
```
head(fs_turnover)
```

```
## # A tsibble: 6 x 2 [1M]
##      Month Turnover
##      <mth>    <dbl>
## 1 1982 Apr     85.1
## 2 1982 May     85.1
## 3 1982 Jun     82.8
## 4 1982 Jul     82.1
## 5 1982 Aug     81.8
## 6 1982 Sep     84.6
```

2 Data Preprocessing and Exploring the Time-Series

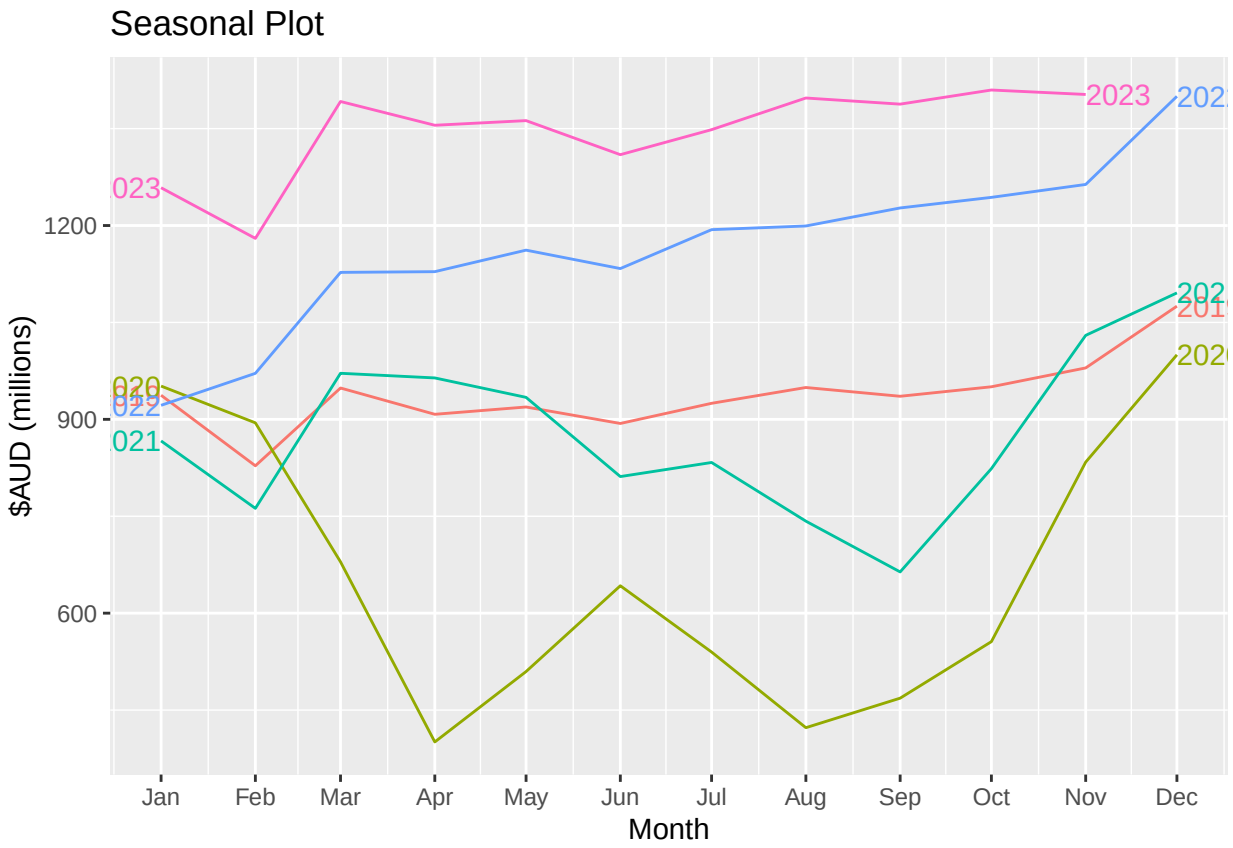
In this section I have renamed the OBS_VALUE to turnover and TIME_PERIOD to Month to ensure the variable names are more meaningful and easy to understand. I plotted the time-series graph to give an idea of what the forecasting patterns looks like.

```
fs_turnover %>% autoplot(Turnover) + ggtitle("Turnovers of Victorian cafes, restaurants & fast-food") +
  ylab("$AUD (millions)")
```

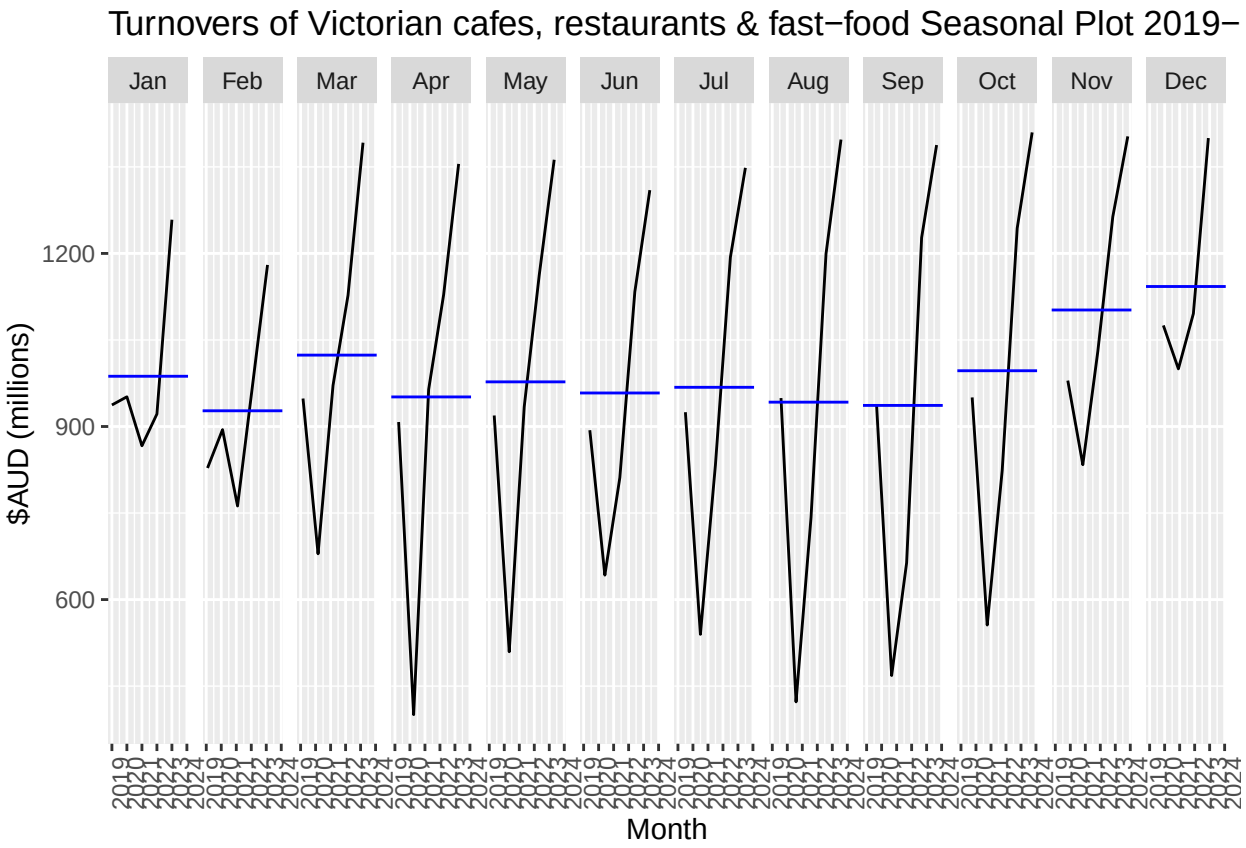


As we can observe, there is a general increase of turnovers for the food service industry. We notice however around 2020, there is a huge downward peak because of the COVID-19 pandemic. But post-pandemic we can observe a steady upward trend indicating the food-service turnover is doing well.

```
# seasonal plot
fs_turnover %>%
  filter(year(Month)>2018) %>%
  gg_season(Turnover, labels = 'both') + guides(colour = "none") +
  labs(
    title = "Seasonal Plot ",
    y = "$AUD (millions)")
```



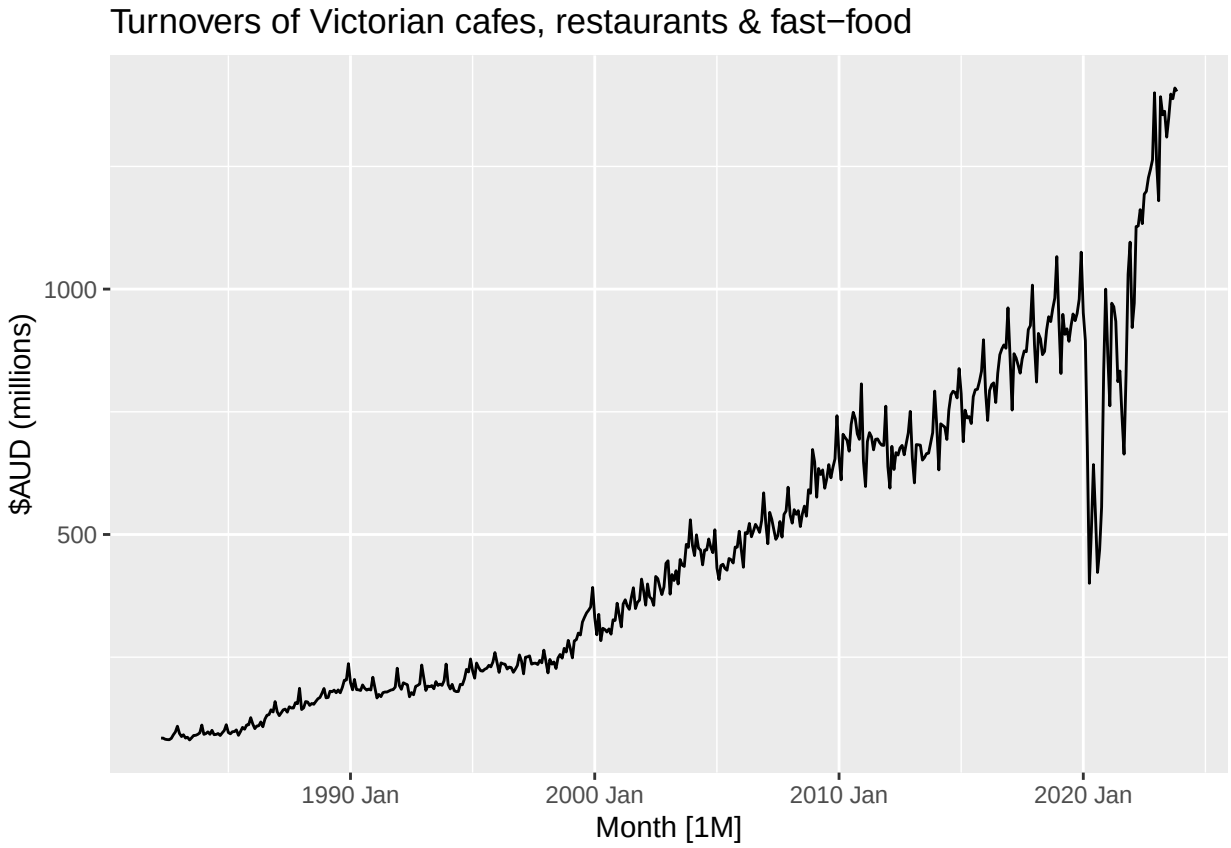

```
#subseries plot
fs_turnover %>%
  filter(year(Month)>2018) %>%
  gg_subseries(Turnover) +
  labs(
    title = "Turnovers of Victorian cafes, restaurants & fast-food Seasonal Plot 2019-2023",
    y = "$AUD (millions)"
  )
```



Plotting the seasonal plot, we can observe that between 2018 and 2023 in around September, there is a gradual increase of the turnover all the way until around January where it begins to gradually decrease. We can further see in the sub-series plot a sharp in September, peaking around December. This could reflect an increase in dining out associated with spring and summer activities, including end-of-year celebrations. After December, there is a smaller margin possible due to a combination of end of holiday season and people may choose to eat at home or go on vacation outside of Australia.

2.1 Data Transformation

```
fs_turnover %>% autoplot(Turnover) + ggtitle("Turnovers of Victorian cafes, restaurants & fast-food") +
  ylab("$AUD (millions)")
```

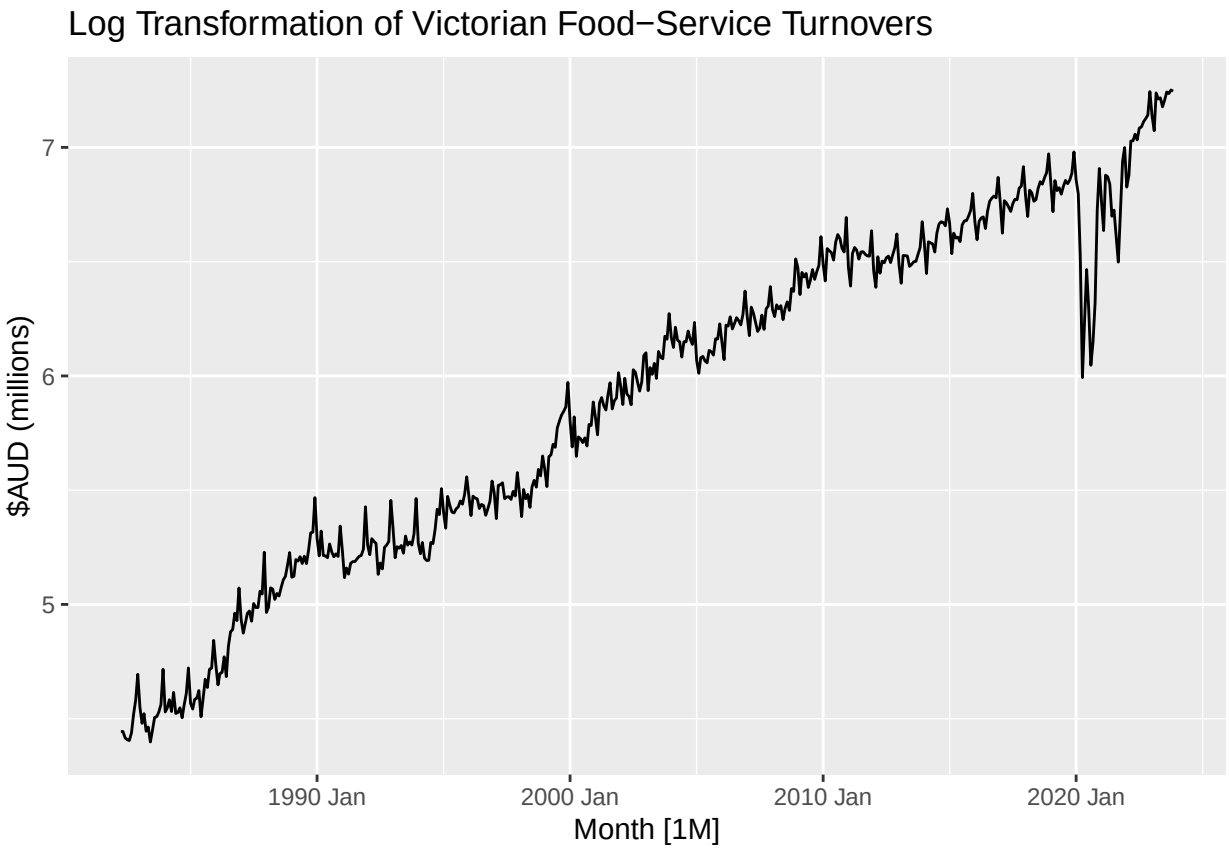



```
lambda <- fs_turnover %>% features(Turnover, features = guerrero) %>%
   pull(lambda_guerrero)
lambda
```

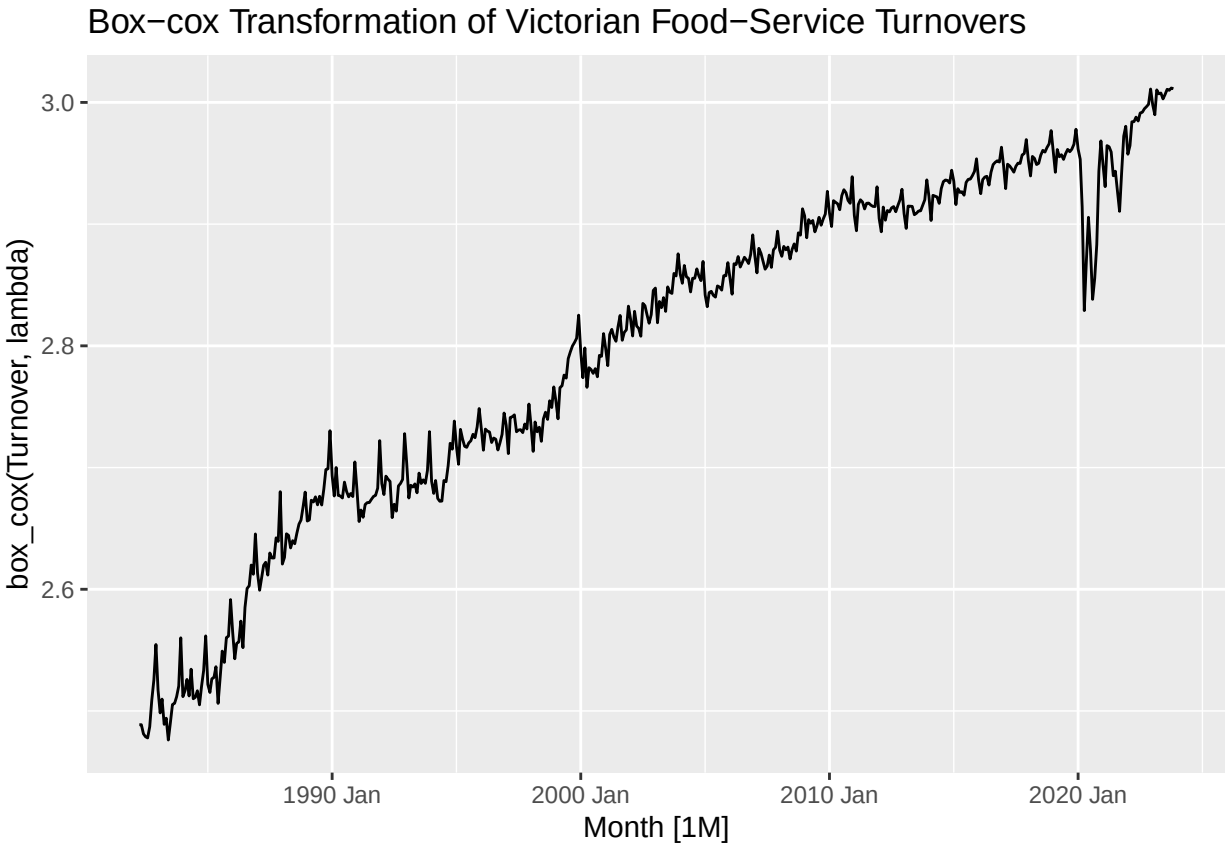
```
## [1] -0.292089
```

```
fs_logtransformed <- fs_turnover %>%
  mutate(log_turnover = log(Turnover))

fs_logtransformed %>% autoplot(log_turnover) + ggtitle("Log Transformation of Victorian Food-Service Tu
  ylab("$AUD (millions)")
```




```
fs_turnover %>%  autoplot(box_cox(Turnover, lambda)) + ggtitle("Box-cox Transformation of Victorian Food
```



We can see that performing Wilcoxon test we get a lambda of -0.29. We do not see a major difference in both the log and box-cox transformation, as both of the plots appear more linear compared to the original data. This indicates our variance has been stabilised. I will be going for a log transformation since there is not much difference throughout the rest of the analysis.

3 Exploring Forecasting Models

Now we are going to compare three forecasting models to see which performs the best at predicting the turnovers of the years. First we will split our data into training and validation sets.

```
#training 1982 - 2019
fs_train <- fs_logtransformed %>% slice(1:453)
#test 2020- present
fs_test <- fs_logtransformed %>%
  filter(year(Month) >= 2020)
```

3.1 Seasonal Naive vs drift model

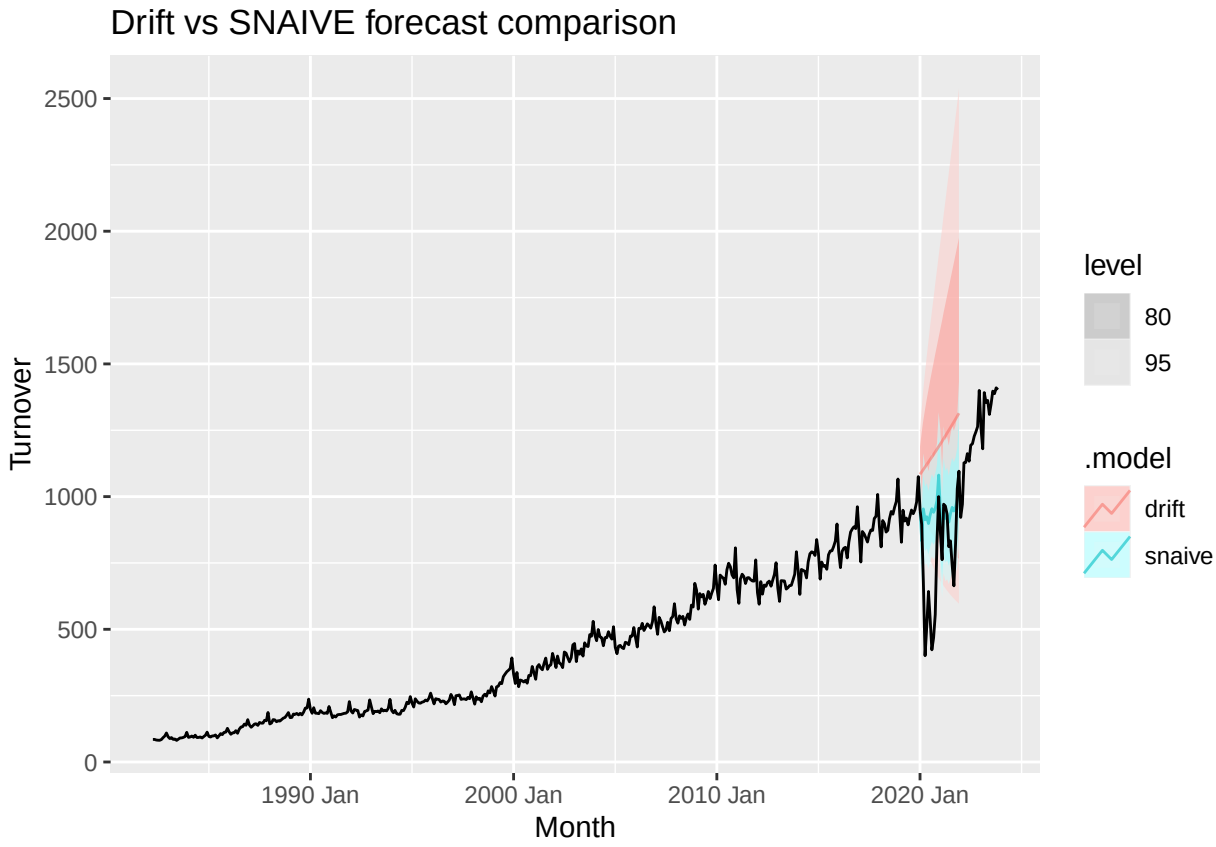
I have opted for a seasonal naive method, making an assumption that most customers will be dining out during spring-summer seasonal and hence we may notice some seasonal trends. A drift method is also considered because as we observed before, there has been a constant increase over time.

```
fit_train <- fs_train %>%
  model(
    snaive = SNAIVE(log(Turnover)),
    drift = RW(log(Turnover) ~ drift())
  )
```

```
fit_train %>%  glance()
```

```
## # A tibble: 2 x 2
##   .model  sigma2
##   <chr>    <dbl>
## 1 snaive 0.00675
## 2 drift  0.00540
```

```
fc_test <- fit_train %>%
  forecast(h = "2 years")
fc_test %>% autoplot(fs_turnover, alpha=0.6) + ggtitle("Drift vs SNAIVE forecast comparison")
```




```
fit_train %>% accuracy() %>% dplyr::select(MAE, RMSE, MAPE, MASE)
```

```
## # A tibble: 2 x 4
##     MAE  RMSE  MAPE  MASE
##   <dbl> <dbl> <dbl> <dbl>
## 1  29.6  39.4  7.70 1
## 2  22.9  34.9  5.66 0.775
```

```
fc_test %>% accuracy(fs_turnover) %>% dplyr::select(MAE, RMSE, MAPE, MASE)
```

```
## # A tibble: 2 x 4
##     MAE  RMSE  MAPE  MASE
##   <dbl> <dbl> <dbl> <dbl>
## 1  429.  464.  67.8 14.5
## 2  194.  259.  34.9  6.55
```

It seems the drift method has a lower sigma^2 and RMSE suggesting it may be fitting the historical data

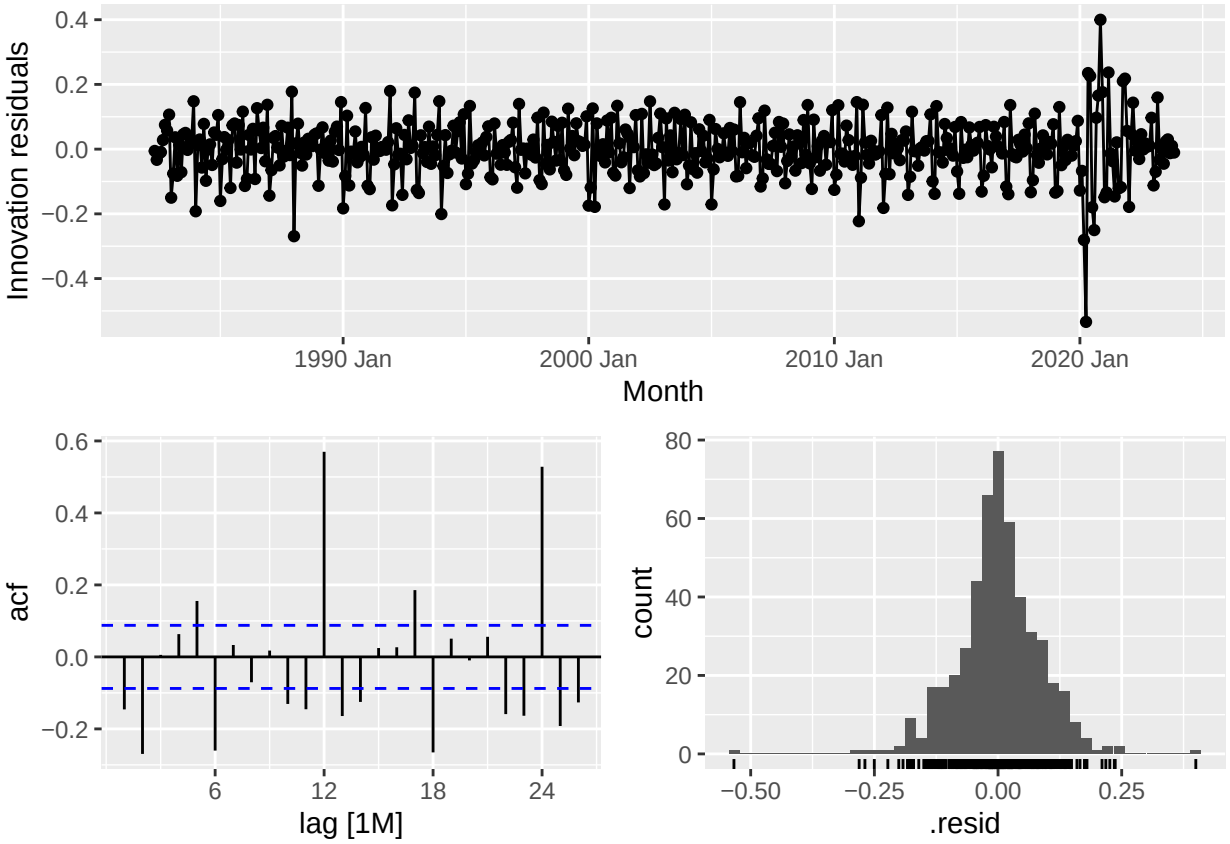
better. However, if seasonal patterns are significant and you need to forecast turnover for food services, a method that accounts for both trend and seasonality would be more suitable. Observing the plot we can see the drift capture the trend over time. Therefore a drift model will be considered. Now we will take a look at the residuals.

```
fit_snaive <- fs_turnover %>%
  model(
    drift = RW(log(Turnover) ~ drift()),
  )
fit_snaive %>%
  gg_tsresiduals()
```

```
## Warning: Removed 1 row containing missing values (`geom_line()`).
```

```
## Warning: Removed 1 rows containing missing values (`geom_point()`).
```

```
## Warning: Removed 1 rows containing non-finite values (`stat_bin()`).
```

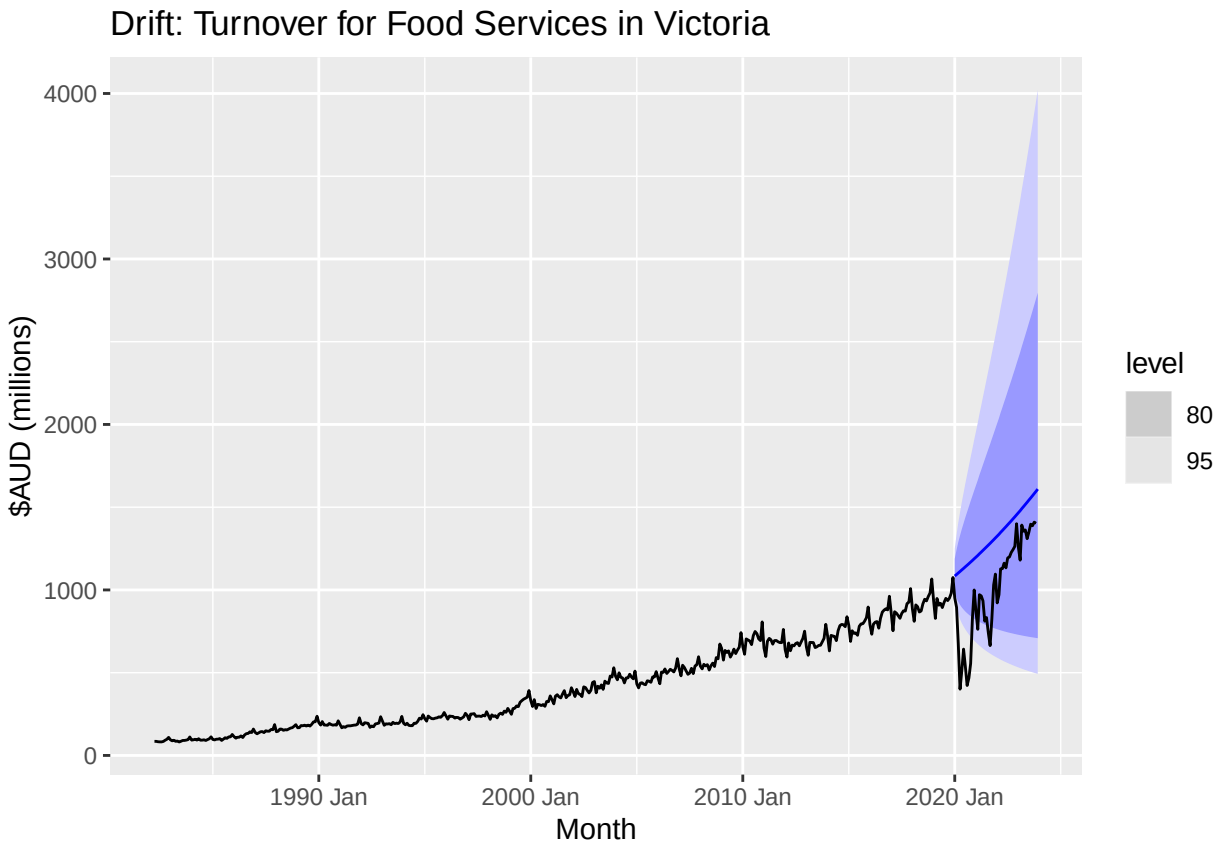


The residuals do not indicate any pattern of some sort over time, suggesting that our variance of errors are homoskedastic. There are a couple of small outliers on both sides in the normal distribution plot. Regarding the ACF plot, the lags are within the confidence bounds, suggesting there is no significant autocorrelation in the residuals at different lag times.

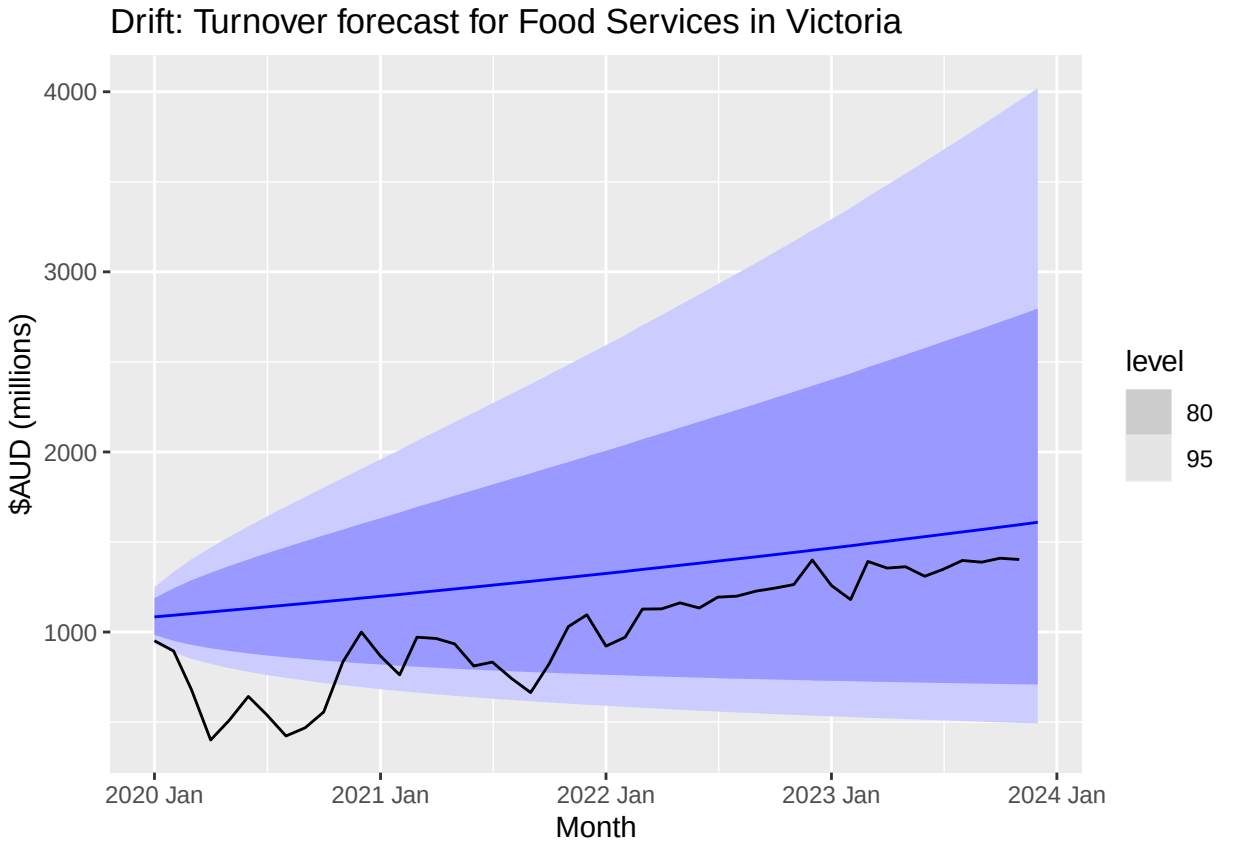
```
turnover_benchmark_fit <-  fs_train %>%
  model(drift = RW(log(Turnover) ~ drift()))

#forecast 3 years from dec 2019 since our data ends on november 2023.
turnover_benchmark_fc <- turnover_benchmark_fit %>%
  forecast(h = '4 years')
```

```
turnover_benchmark_fc %>% autoplot(fs_turnover) +
  labs(title = "Drift: Turnover for Food Services in Victoria",
       y = "$AUD (millions)") +
  guides(colour = guide_legend(title = "Forecast"))
```




```
turnover_benchmark_fc %>% autoplot(fs_test) +
  labs(title = "Drift: Turnover forecast for Food Services in Victoria",
       y = "$AUD (millions)") +
  guides(colour = guide_legend(title = "Forecast"))
```




```
benchmark_point_intervals <- fs_train%>%
  model(drift = RW(log(Turnover) ~ drift())) %>%
  forecast(h = '4 years') %>%
  hilo()
print(head(benchmark_point_intervals))
```

```
## # A tsibble: 6 x 6 [1M]
## # Key:       .model [1]
##   .model    Month        Turnover .mean                  `80%`
##   <chr>     <mth>          <dist> <dbl>                 <hilo>
## 1 drift  2020 Jan t(N(7, 0.0054)) 1084. [983.9556, 1188.166]80
## 2 drift  2020 Feb  t(N(7, 0.011)) 1093. [951.4507, 1242.626]80
## 3 drift  2020 Mar  t(N(7, 0.016)) 1102. [928.3573, 1287.911]80
## 4 drift  2020 Apr  t(N(7, 0.022)) 1112. [910.0498, 1328.648]80
## 5 drift  2020 May  t(N(7, 0.027)) 1121. [894.7515, 1366.617]80
## 6 drift  2020 Jun  t(N(7, 0.033)) 1130. [881.5583, 1402.725]80
## # i 1 more variable: `95%` <hilo>
```

We can see that our forecasted plot shows a constant upward trend over time.We can als observe forecasted turnover statistics. For example, in January 2020, the forecasted turnover is 1084 million with an 80% prediction interval between approximately 984 and 1188 million, and a 95% interval between 936 and 1249 million.

However, the plot doesn't show any seasonal adjustments or account for potential cyclical events that could impact turnover (like holidays or economic changes). Other models are to be considered.

3.2 Evaluating Exponential Smoothing (ETS)

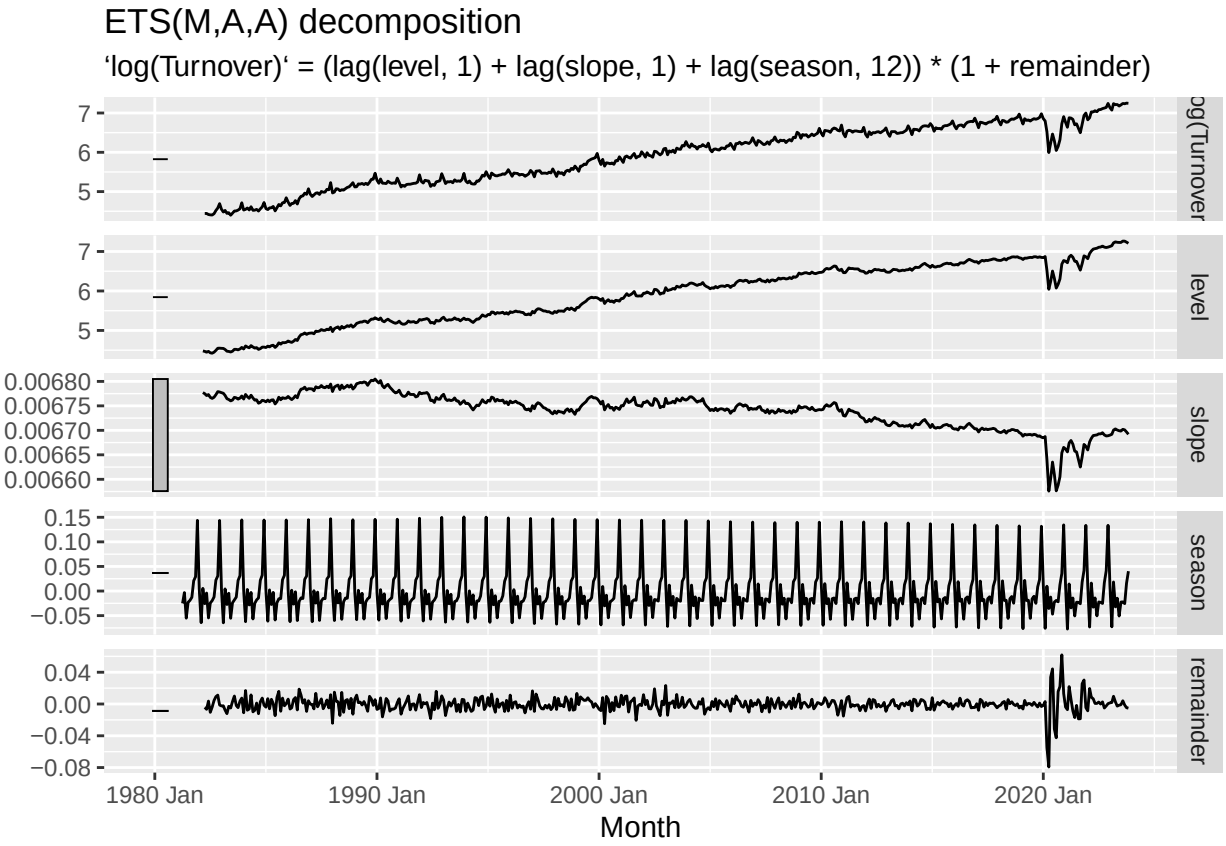
```
fit_ets <- fs_turnover %>%
  model(ETS(log(Turnover)))
report(fit_ets)
```

```
## Series: Turnover
## Model: ETS(M,A,A)
## Transformation: log(Turnover)
##   Smoothing parameters:
##     alpha = 0.9414445
##     beta  = 0.0001245297
##     gamma = 0.03128236
##
##   Initial states:
##      l[0]        b[0]        s[0]      s[-1]       s[-2]     s[-3]      s[-4]
##  4.480777 0.006777159 0.005449828 -0.0642712 0.003061288 0.1436884 0.02914804
##       s[-5]       s[-6]       s[-7]       s[-8]       s[-9]       s[-10]
##  0.02086616 -0.01270929 -0.01896974 -0.02289417 -0.05514377 -0.003125694
##      s[-11]
##  -0.0250999
##
##   sigma^2:  1e-04
##
##      AIC     AICc      BIC
## 246.1705 247.4402 317.8188
```

R has given us an ETS(M,A,A) model, assuming that our model has an additive trend and additive seasonality. lpha is the smoothing parameter for the level, which is very close to 1, suggesting a strong weighting on recent observations for estimating the level of the series.The smoothing parameter for alpha is close to 1, suggesting strong weighting on recent observations. Our beta is also very small, indicating tha the trend does not change drastically.

```
components(fit_ets) %>% autoplot()
```

```
## Warning: Removed 12 rows containing missing values (`geom_line()`).
```



We can further observe our slow slowly decreases over time with this model and our seasonality peaks are almost identical.

```
#ETS-  MAM c
ETS_fit <- fs_train %>%
  model(
    hw_multi = ETS(log(Turnover) ~ error("M") + trend("A") + season("M")),
    no_trend = ETS(log(Turnover) ~ error("M") + trend("N") + season("M")))
report(ETS_fit)
```

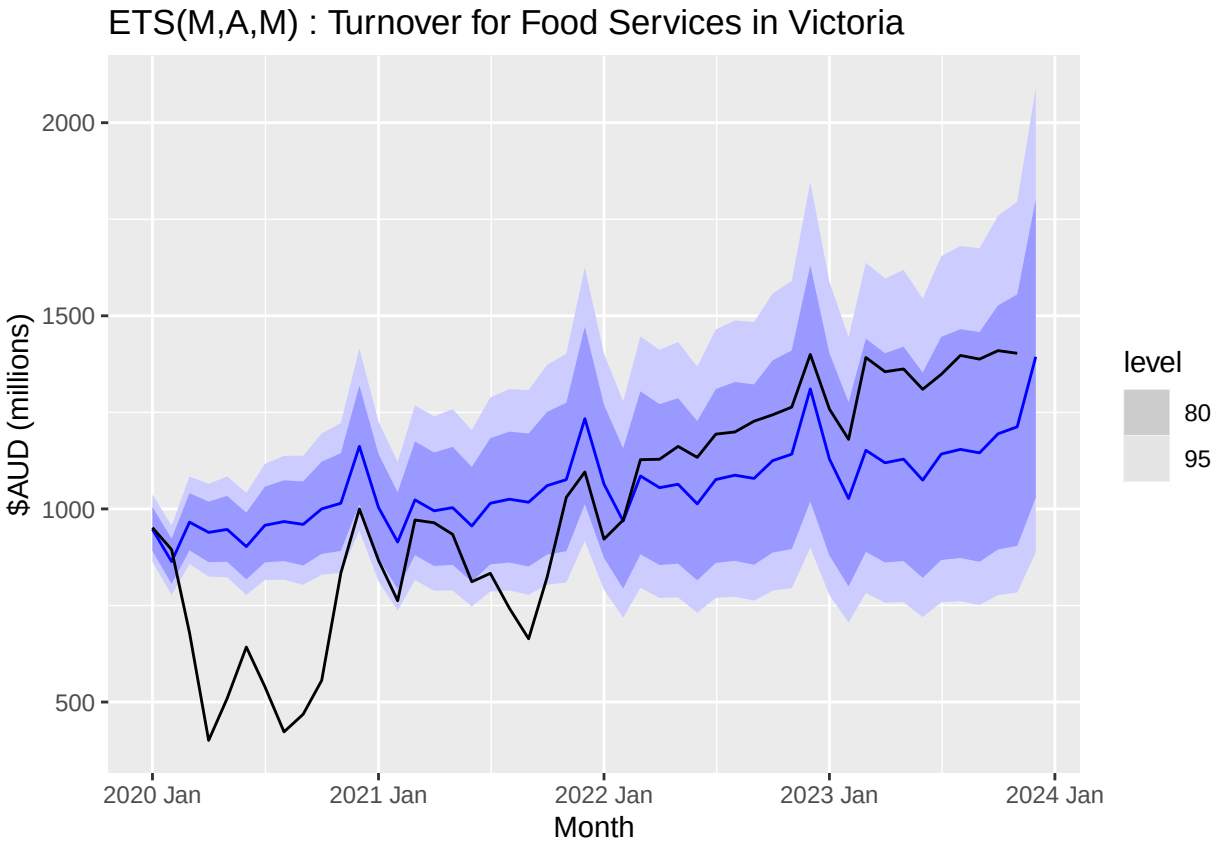
```
## Warning in report.mdl_df(ETS_fit): Model reporting is only supported for
## individual models, so a glance will be shown. To see the report for a specific
## model, use `select()` and `filter()` to identify a single model.
```

```
## # A tibble: 2 x 9
##   .model      sigma2 log_lik   AIC  AICc   BIC     MSE    AMSE     MAE
##   <chr>        <dbl>   <dbl> <dbl> <dbl> <dbl>   <dbl>   <dbl>   <dbl>
## 1 hw_multi 0.0000468    88.5 -143. -142. -73.0 0.00137 0.00205 0.00516
## 2 no_trend 0.0000516    66.4 -103. -102. -41.1 0.00150 0.00232 0.00545
```

Rather than using R's suggested model, I have opted for holt-winters multiplicative model and no trend model to see if there is any difference. We can see that the AICc value is much lower for the hw multiplicative model. Thus I will use this model for further forecasting analysis.

```
ETS_fit <- fs_train %>%
  model(
    ETS(log(Turnover) ~ error("M") + trend("A") + season("M")))
ETS_fc <- ETS_fit %>%
  forecast(h = '4 years')
```

```
ETS_fc  %>% autoplot(filter(fs_turnover, year(Month)>2019))  +
  labs(title = "ETS(M,A,M) : Turnover for Food Services in Victoria",
       y = "$AUD (millions)") +
  guides(colour = guide_legend(title = "Forecast"))
```




```
ETS_point_intervals <- ETS_fit %>%
  forecast(h= '4 years') %>%
  hilo()
print(head(ETS_point_intervals))
```

```
## # A tsibble: 6 x 6 [1M]
## # Key:       .model [1]
##   .model                    Month          Turnover .mean                  `80%`
##   <chr>                     <mth>            <dist> <dbl>                 <hilo>
## 1 "ETS(log(Turnover) ~~ 2020 Jan t(N(6.9, 0.0022))  947. [891.0235, 1004.7698]80
## 2 "ETS(log(Turnover) ~~ 2020 Feb t(N(6.8, 0.0028))  864. [805.9425,  922.9147]80
## 3 "ETS(log(Turnover) ~~ 2020 Mar t(N(6.9, 0.0036))  966. [892.8791, 1040.7777]80
## 4 "ETS(log(Turnover) ~~ 2020 Apr t(N(6.8, 0.0042))  939. [862.1911, 1018.7190]80
## 5 "ETS(log(Turnover) ~~ 2020 May t(N(6.9, 0.0049))  947. [863.1546, 1033.6327]80
## 6 "ETS(log(Turnover) ~~ 2020 Jun t(N(6.8, 0.0056))  902. [817.7877,  990.2291]80
## # i 1 more variable: `95%` <hilo>
```

Compared to the drift model previsouly, our forecasted pattern is much better. This time it accounts for for the seasonality peaks around the month of January with a steady but stable trend over time as indicated by the widths of the peaks.

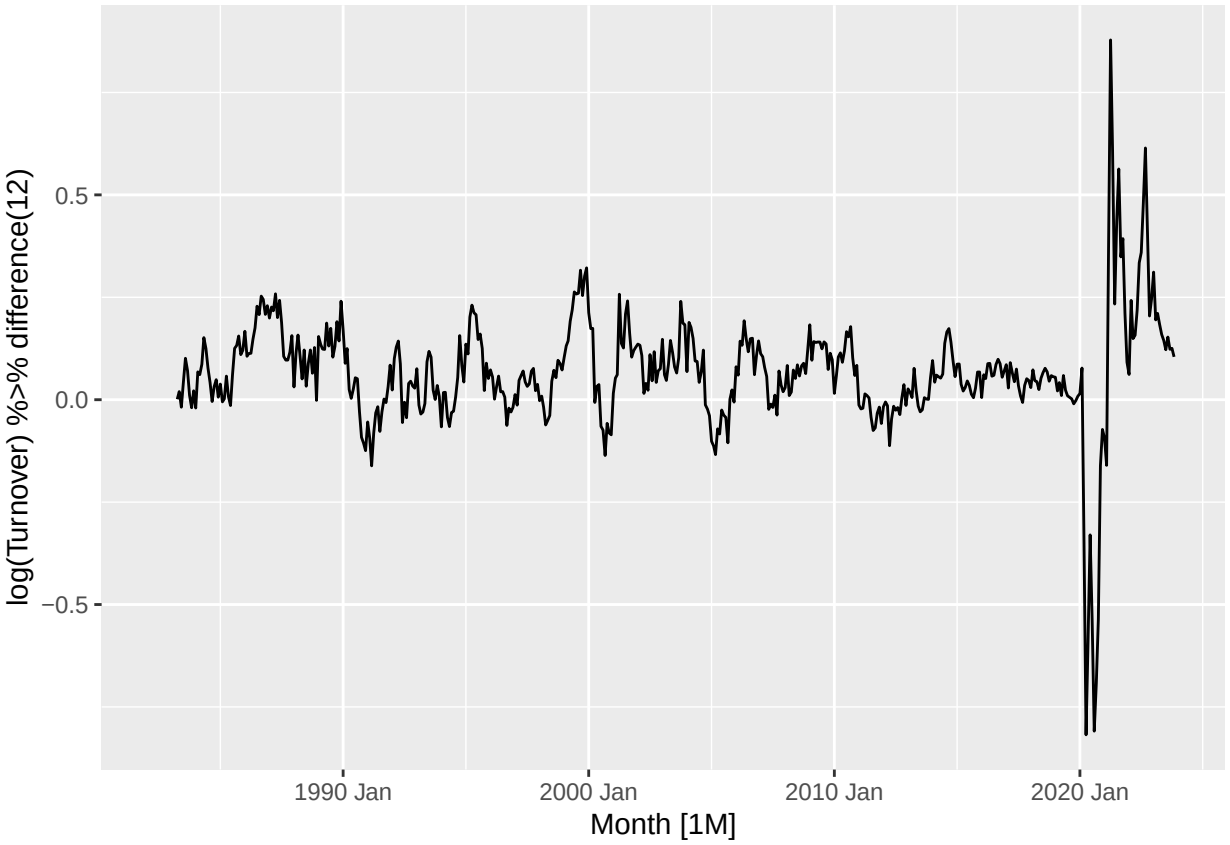
3.3 ARIMA

Now we will move onto ARIMA modelling. AR: captures the relationship between an observation and lagged observation. I: difference of raw observations MA: combination of pass error terms.

Before modelling, we will have to transform our data to account for seasonality difference. Further transformation

```
fs_turnover %>% autoplot(log(Turnover)  %>%
  difference(12))
```

```
## Warning: Removed 12 rows containing missing values (`geom_line()`).
```

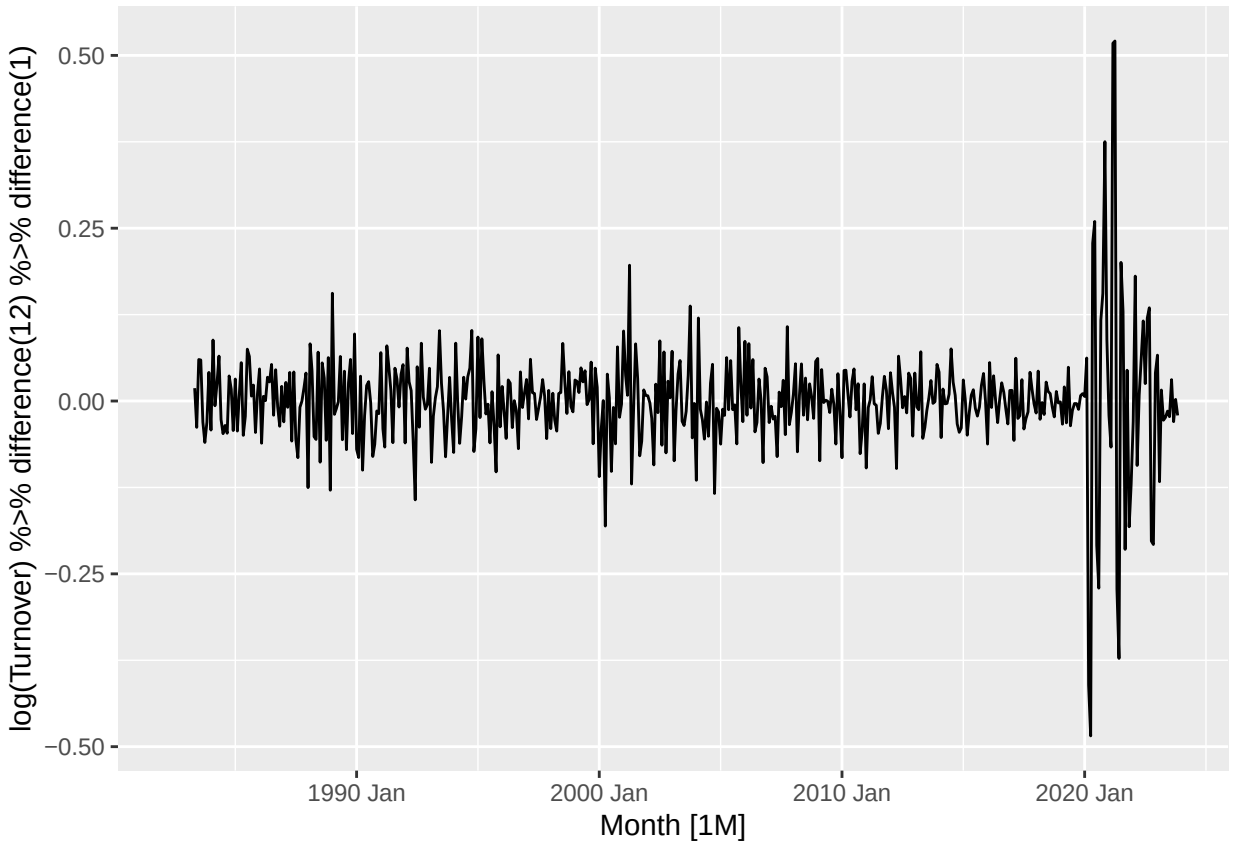



```
#Automatic difference selection
fs_turnover %>% mutate(log_turnover = log(Turnover)) %>%
  features(log_turnover, unitroot_nsdiffs)
```

```
## # A tibble: 1 x 1
##   nsdiffs
##     <int>
## 1       0
```

```
fs_turnover %>% autoplot(log(Turnover)  %>%
  difference(12) %>% difference (1))
```

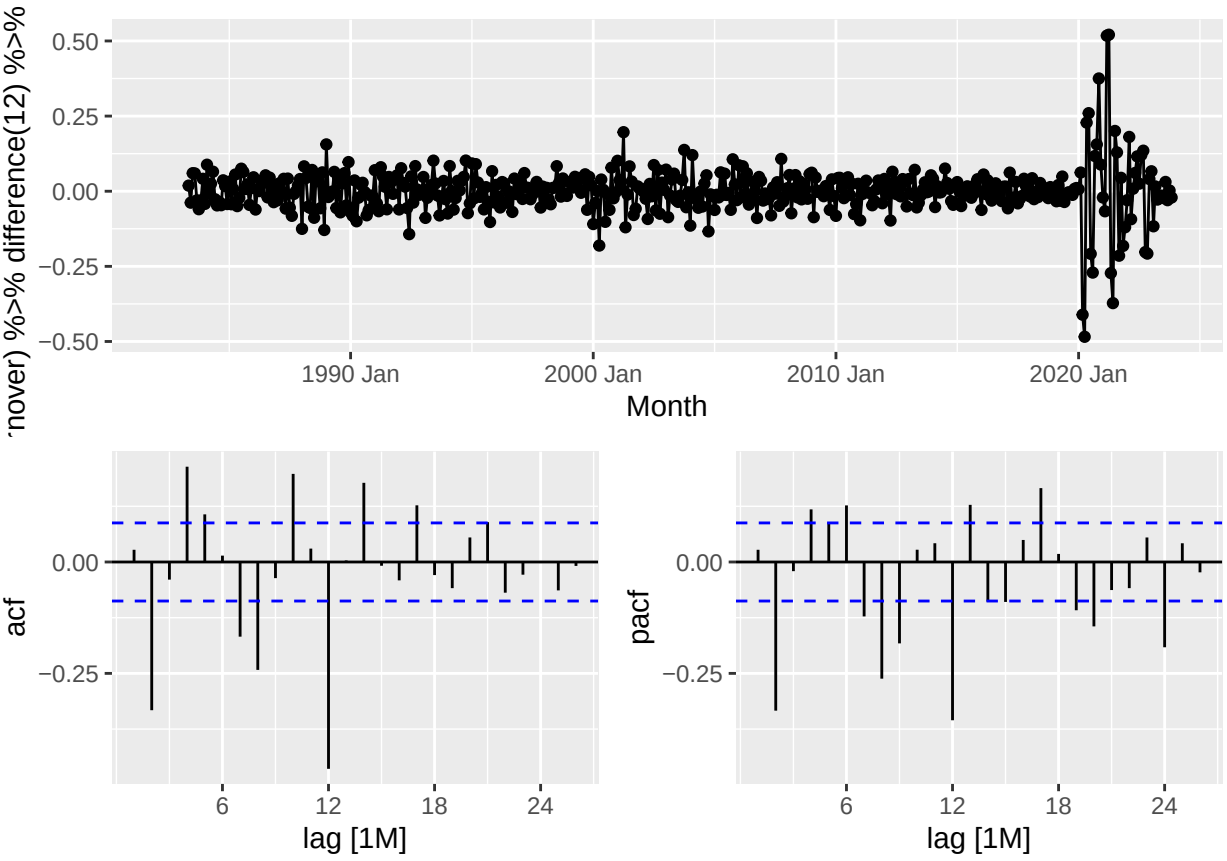
```
## Warning: Removed 13 rows containing missing values (`geom_line()`).
```




```
fs_turnover %>% gg_tsdisplay(log(Turnover) %>% difference(12) %>% difference(),
                             plot_type = 'partial')
```

```
## Warning: Removed 13 rows containing missing values (`geom_line()`).
```

```
## Warning: Removed 13 rows containing missing values (`geom_point()`).
```



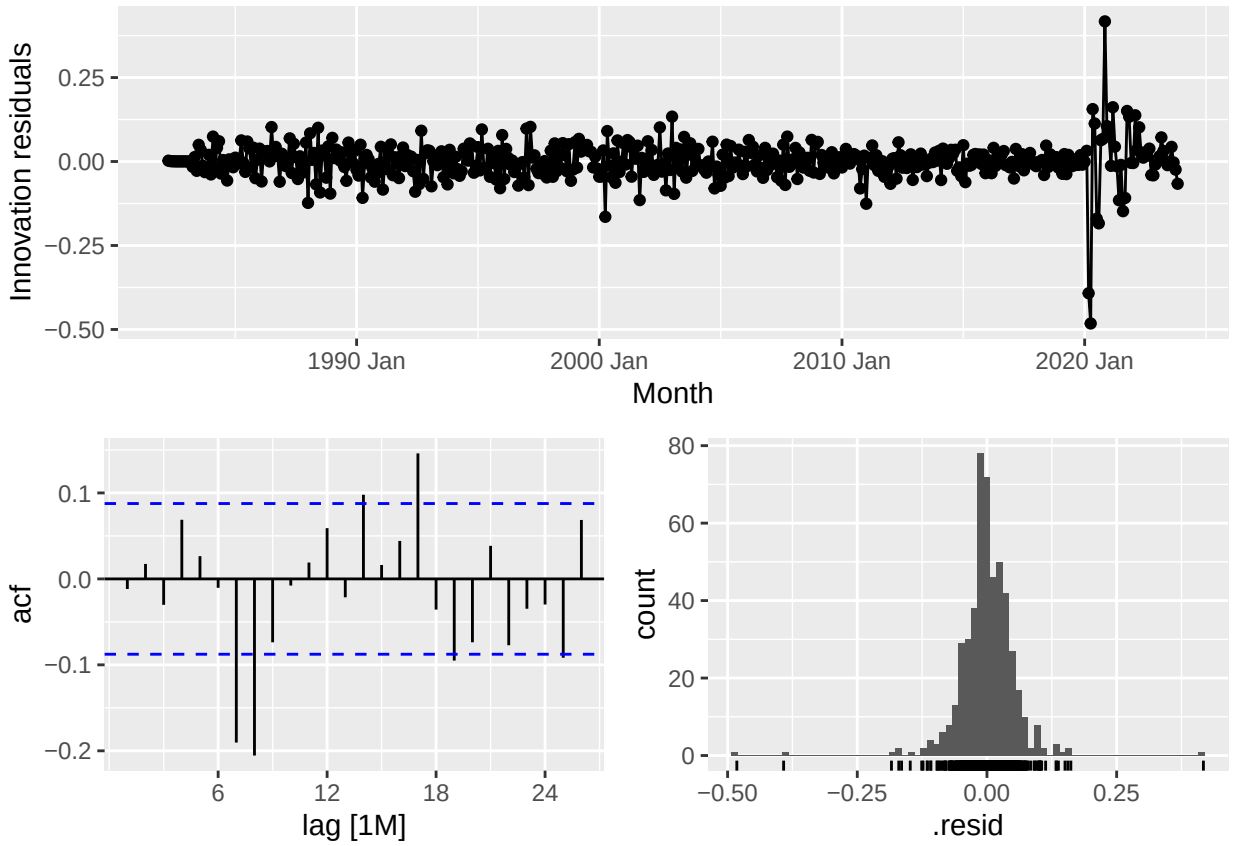
We can observe the following:

- non-seasonal spikes in the PACF: AR(2)
- seasonal MA(1) due to the large spike at 12 in the ACF Hence the model that I chosen is: ARIMA(2,1,0)(0,1,1)

```
arima_210_011 <- fs_turnover %>% model(
  ARIMA(log(Turnover)~pdq(2,1,0)+PDQ(0,1,1))
  )
arima_210_011 %>% report()
```

```
## Series: Turnover
## Model: ARIMA(2,1,0)(0,1,1)[12]
## Transformation: log(Turnover)
##
## Coefficients:
##          ar1      ar2     sma1
##       0.0461  -0.2736  -0.9012
## s.e.  0.0440   0.0438   0.0312
##
## sigma^2 estimated as 0.003224:  log likelihood=697.42
## AIC=-1386.84   AICc=-1386.76   BIC=-1370.09
```

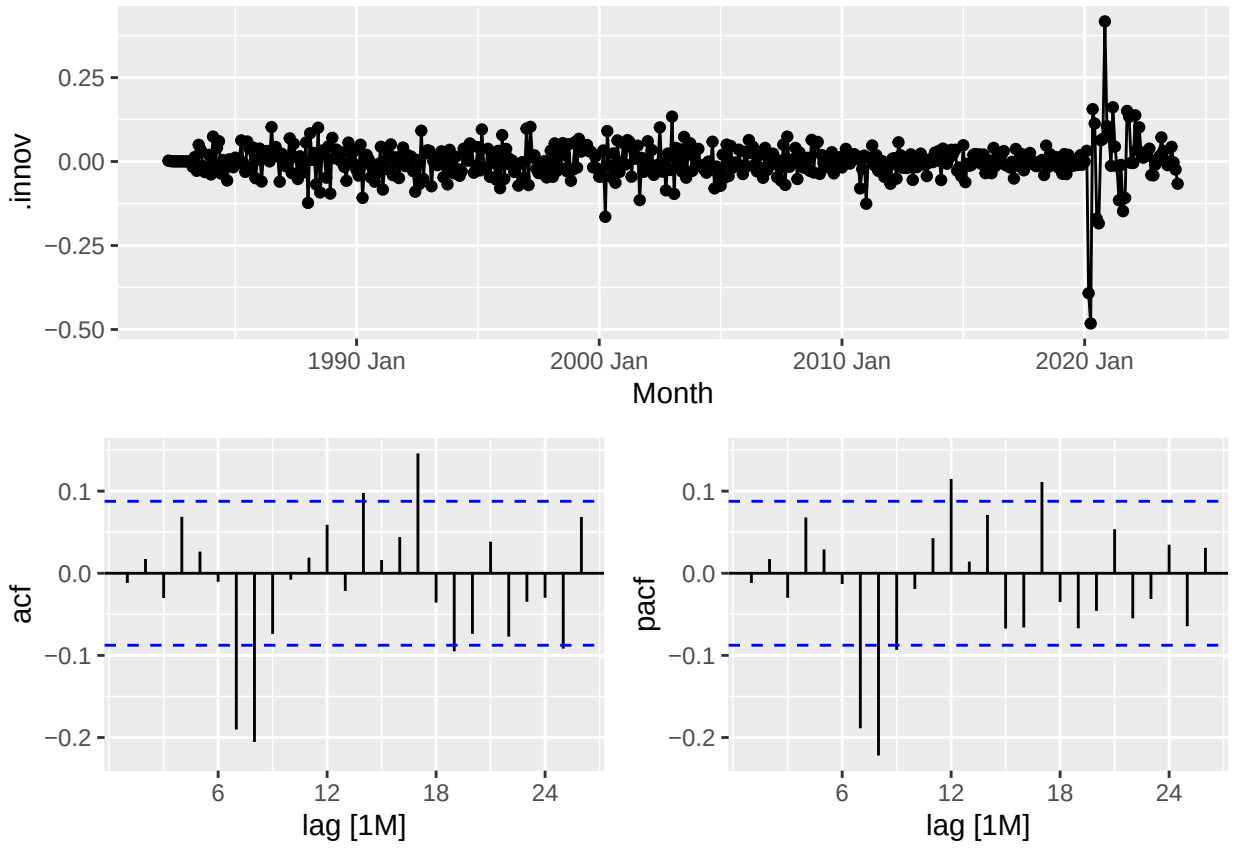
```
arima_210_011 %>% gg_tsresiduals()
```




```
augment(arima_210_011) %>% features(.innov, ljung_box, lag=24, dof=3)
```

```
## # A tibble: 1 x 3
##   .model                                               lb_stat    lb_pvalue
##   <chr>                                                  <dbl>        <dbl>
## 1 ARIMA(log(Turnover) ~ pdq(2, 1, 0) + PDQ(0, 1, 1))    78.9 0.0000000125
```

```
augment(arima_210_011) %>% gg_tsdisplay(.innov, plot_type = 'partial')
```



3.3.1 Other models worth considering:

- MA(2): 2 spikes in ACF plot: ARIMA(0,1,2)(0,1,1)
- AR(4): large spike in ACF: ARIMA(4,1,0)(0,1,1)
- AR(2)MA(2): ARIMA(2,1,2)(0,1,1)

```
arima.models <- fs_turnover %>% model(
  arima_210_011 = ARIMA(log(Turnover)~pdq(2,1,0)+PDQ(0,1,1)),
  arima_012_011 = ARIMA(log(Turnover)~pdq(0,1,2)+PDQ(0,1,1)),
  arima_410_011 = ARIMA(log(Turnover)~pdq(4,1,0)+PDQ(0,1,1)),
  arima_212_011 = ARIMA(log(Turnover)~pdq(2,1,2)+PDQ(0,1,1))
  )
arima.models %>%  glance() %>% select(.model, AIC, AICc, BIC)
```

```
## # A tibble: 4 x 4
##   .model           AIC   AICc    BIC
##   <chr>          <dbl>  <dbl>  <dbl>
## 1 arima_210_011 -1387. -1387. -1370.
## 2 arima_012_011 -1380. -1380. -1364.
## 3 arima_410_011 -1386. -1386. -1361.
## 4 arima_212_011 -1396. -1396. -1371.
```

Comparing the models that I have manually chosen, ARIMA(2,1,2)(0,1,1) has the lowest AICc value. Before using this model to forecast, it is worth considering the model R has chosen for us.

The auto.arima() function is useful, but anything automated can be a little dangerous!

```
arima_models_auto <- fs_turnover %>%
  model(arima_auto1 = ARIMA(log(Turnover)),
```

```
        arima_auto2 = ARIMA(log(Turnover), stepwise = FALSE, approximation = FALSE)
  )
arima_models_auto %>% select(arima_auto1) %>%  report()
```

```
## Series: Turnover
## Model: ARIMA(0,1,4)(2,0,0)[12]
## Transformation: log(Turnover)
##
## Coefficients:
##          ma1      ma2     ma3     ma4    sar1    sar2
##       -0.0111  -0.2386  0.0891  0.2553  0.4458  0.2995
## s.e.   0.0449   0.0425  0.0568  0.0581  0.0427  0.0437
##
## sigma^2 estimated as 0.003926:  log likelihood=672.79
## AIC=-1331.59   AICc=-1331.36   BIC=-1302.1
```

```
arima_models_auto %>% select(arima_auto2) %>%  report()
```

```
## Series: Turnover
## Model: ARIMA(2,1,2)(2,0,0)[12]
## Transformation: log(Turnover)
##
## Coefficients:
##          ar1      ar2      ma1     ma2    sar1    sar2
##       0.7536  -0.8991  -0.8745  0.8300  0.4258  0.3251
## s.e.  0.0386   0.0378   0.0447  0.0512  0.0434  0.0447
##
## sigma^2 estimated as 0.003775:  log likelihood=682.71
## AIC=-1351.42   AICc=-1351.2   BIC=-1321.94
```

The auto funcitonchose an ARIMA(0,1,4)(2,0,0)[12] for us. Putting all our models together we get:

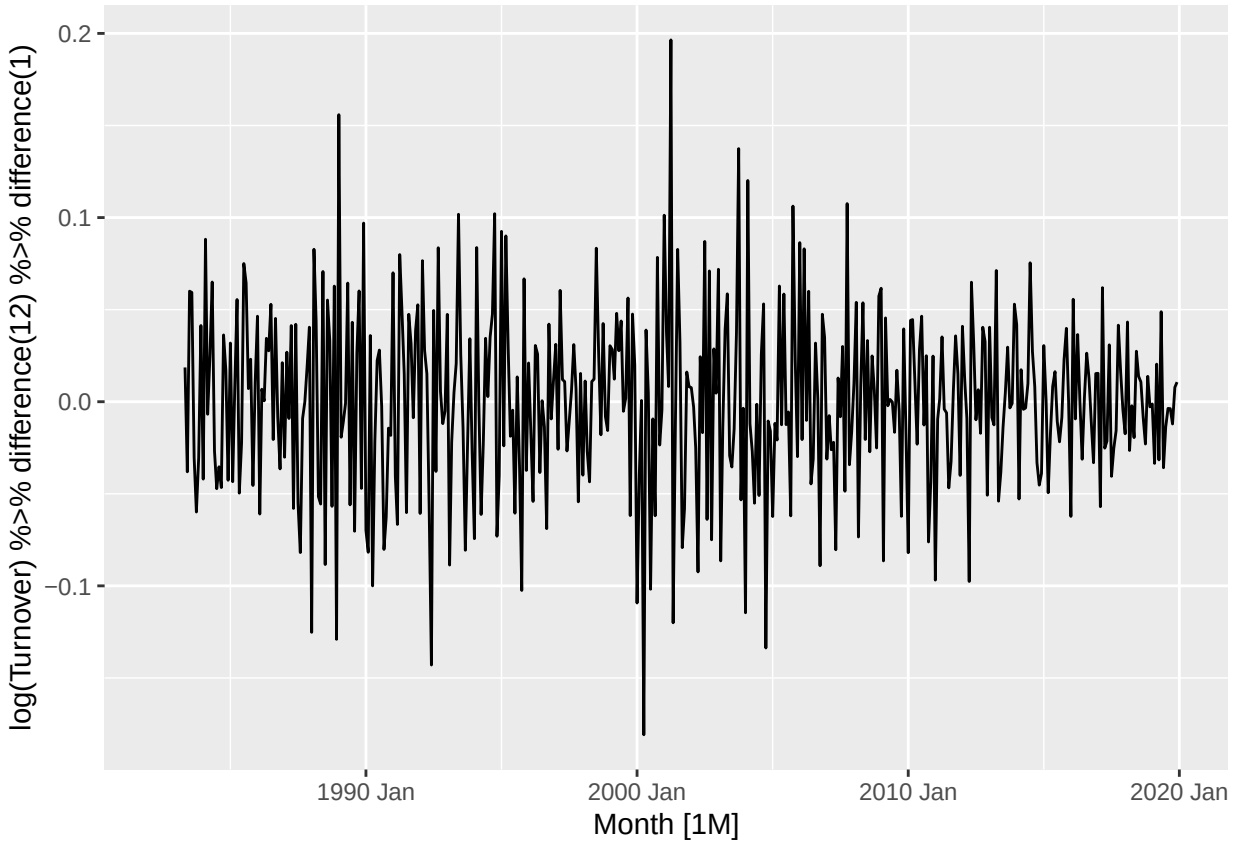
```
fit_all <- fs_turnover %>%
  model(
    arima_210_011 = ARIMA(log(Turnover)~pdq(2,1,0)+PDQ(0,1,1)),
  arima_012_011 = ARIMA(log(Turnover)~pdq(0,1,2)+PDQ(0,1,1)),
  arima_410_011 = ARIMA(log(Turnover)~pdq(4,1,0)+PDQ(0,1,1)),
  arima_212_011 = ARIMA(log(Turnover)~pdq(2,1,2)+PDQ(0,1,1)),
  arima_212_200 = ARIMA(log(Turnover), stepwise = FALSE, approx = FALSE)
  )
glance(fit_all)
```

```
## # A tibble: 5 x 8
##   .model         sigma2 log_lik    AIC   AICc    BIC ar_roots   ma_roots
##   <chr>           <dbl>   <dbl>  <dbl>  <dbl>  <dbl> <list>     <list>
## 1 arima_210_011 0.00322    697. -1387. -1387. -1370. <cpl [2]>  <cpl [12]>
## 2 arima_012_011 0.00326    694. -1380. -1380. -1364. <cpl [0]>  <cpl [14]>
## 3 arima_410_011 0.00322    699. -1386. -1386. -1361. <cpl [4]>  <cpl [12]>
## 4 arima_212_011 0.00314    704. -1396. -1396. -1371. <cpl [2]>  <cpl [14]>
## 5 arima_212_200 0.00378    683. -1351. -1351. -1322. <cpl [26]> <cpl [2]>
```

Out of all the models, ARIMA(2,1,2)(2,0,0) still has the lowest AICc value of -1396. Hence moving foward we will use this model to forecast.

```
#ARIMA - best arima model ARIMA(2,1,2)(0,1,1)
fs_train %>% autoplot(log(Turnover)  %>%
  difference(12) %>% difference (1))
```

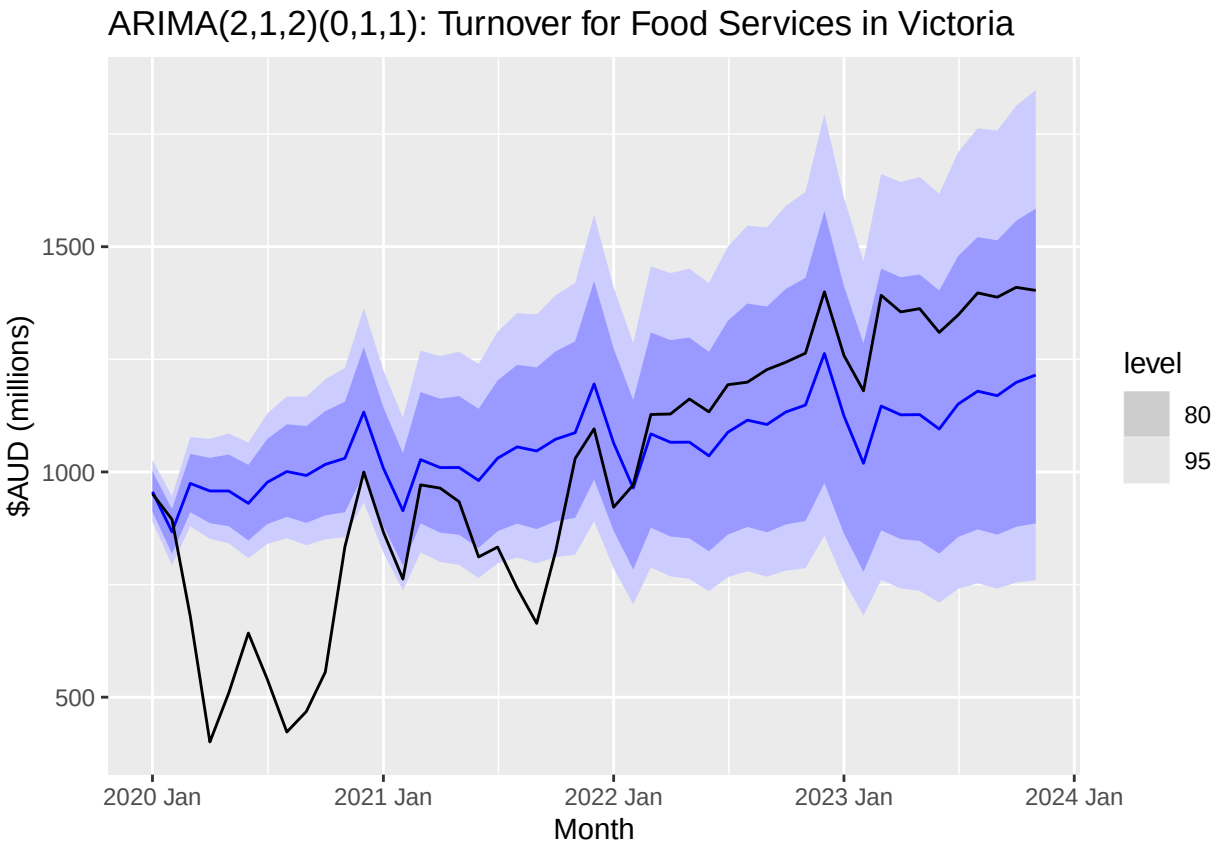
```
## Warning: Removed 13 rows containing missing values (`geom_line()`).
```




```
arima_fit <- fs_train %>%
  model(
    arima012111 = ARIMA(log(Turnover) ~ pdq(2,1,2) + PDQ(0,1,1))
  )

arima_fc <- arima_fit %>%
  dplyr::select(arima012111) %>%
  forecast(h =47)

arima_fc %>% autoplot(filter(fs_turnover, year(Month)>2019)) +
   labs(title = "ARIMA(2,1,2)(0,1,1): Turnover for Food Services in Victoria",
        y = "$AUD (millions)") +
  guides(colour = guide_legend(title = "Forecast"))
```




```
arima_point_intervals <- arima_fit %>%
  forecast(h= '3 years') %>%
  hilo()
print(head(arima_point_intervals))
```

```
## # A tsibble: 6 x 6 [1M]
## # Key:       .model [1]
##   .model         Month          Turnover .mean                   `80%`
##   <chr>          <mth>            <dist> <dbl>                  <hilo>
## 1 arima012111 2020 Jan t(N(6.9, 0.0013))  956. [911.5082, 1001.2476]80
## 2 arima012111 2020 Feb  t(N(6.8, 0.002))  867. [818.0932,  917.6423]80
## 3 arima012111 2020 Mar t(N(6.9, 0.0027))  974. [910.5940, 1039.9905]80
## 4 arima012111 2020 Apr t(N(6.9, 0.0035))  958. [886.3467, 1031.2607]80
## 5 arima012111 2020 May t(N(6.9, 0.0042))  958. [879.5888, 1038.5672]80
## 6 arima012111 2020 Jun t(N(6.8, 0.0049))  930. [847.9715, 1015.5281]80
## # i 1 more variable: `95%` <hilo>
```

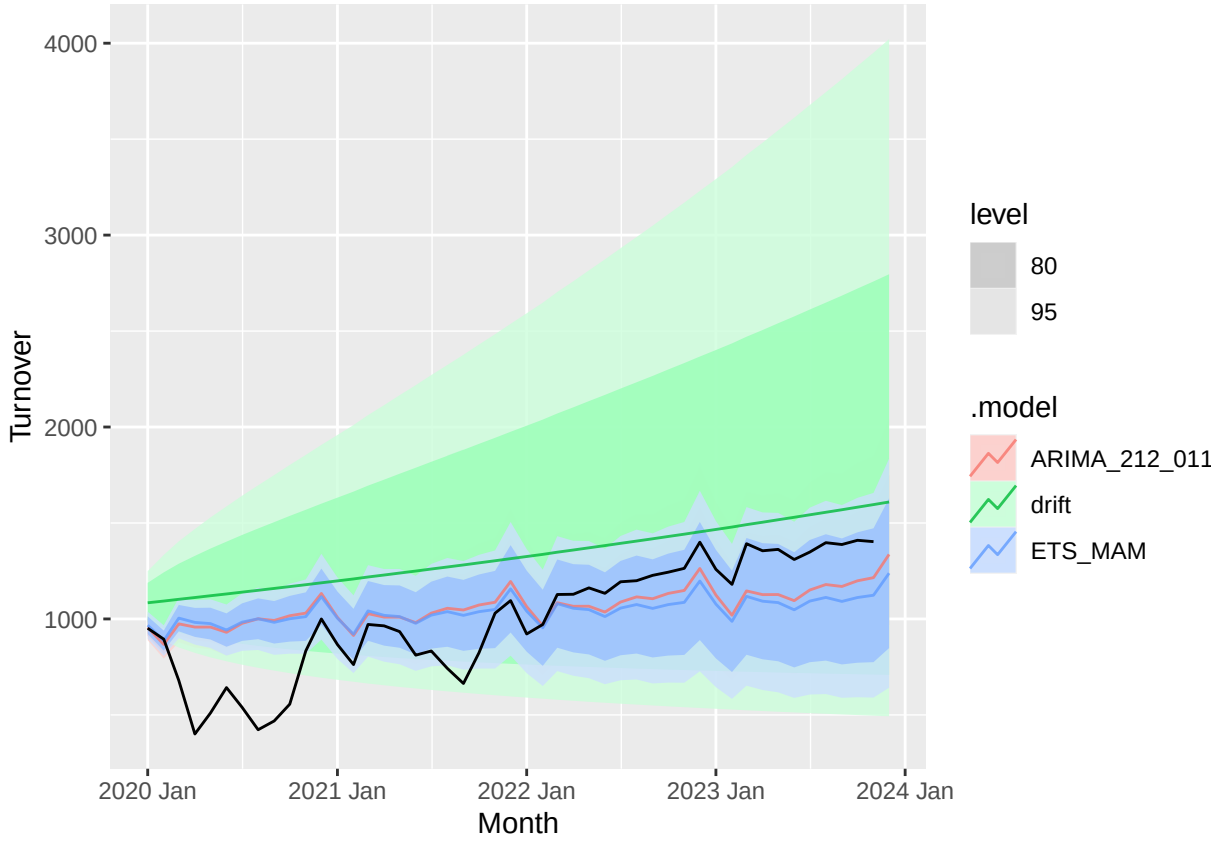
Looking at the forecasted plot, it seems quite similar to our ETS model that we plotted previously.

4 Choosing Overall Model

Now that we have conducted analysis on three different forecasting models, we are going to evaluate which of these three peforms the best using the validation set.

```
models_final <- fs_train %>% model(
  drift = RW(log(Turnover) ~ drift()),
  ETS_MAM=ETS(Turnover~error("M")+trend("A")+season("M")),
  ARIMA_212_011=ARIMA(log(Turnover)~pdq(2,1,2)+PDQ(0,1,1))
```

```
)
models_final %>%
  forecast(h=48) %>%
  autoplot(fs_turnover %>% filter(year(Month)>=2020), alpha=0.8)
```




```
accuracy_benchmark <- accuracy(turnover_benchmark_fc, fs_test) %>%
  mutate(Model = "Drift") %>%
  select(Model, .type, ME, RMSE, MAPE)
```

```
## Warning: The future dataset is incomplete, incomplete out-of-sample data will be treated as missing.
## 1 observation is missing at 2023 Dec
```

```
accuracy_ETS <- accuracy(ETS_fc, fs_test) %>%
  mutate(Model = "ETS") %>%
  select(Model, .type, ME, RMSE, MAPE)
```

```
## Warning: The future dataset is incomplete, incomplete out-of-sample data will be treated as missing.
## 1 observation is missing at 2023 Dec
```

```
accuracy_arima <- accuracy(arima_fc, fs_test) %>%
  mutate(Model = "ARIMA") %>%
  select(Model, .type, ME, RMSE, MAPE)

combined_accuracy <- bind_rows(accuracy_benchmark, accuracy_ETS, accuracy_arima)

kable(combined_accuracy, format = "markdown", caption = "Forecasting Models Accuracy Metrics")
```

Table 2: Forecasting Models Accuracy Metrics

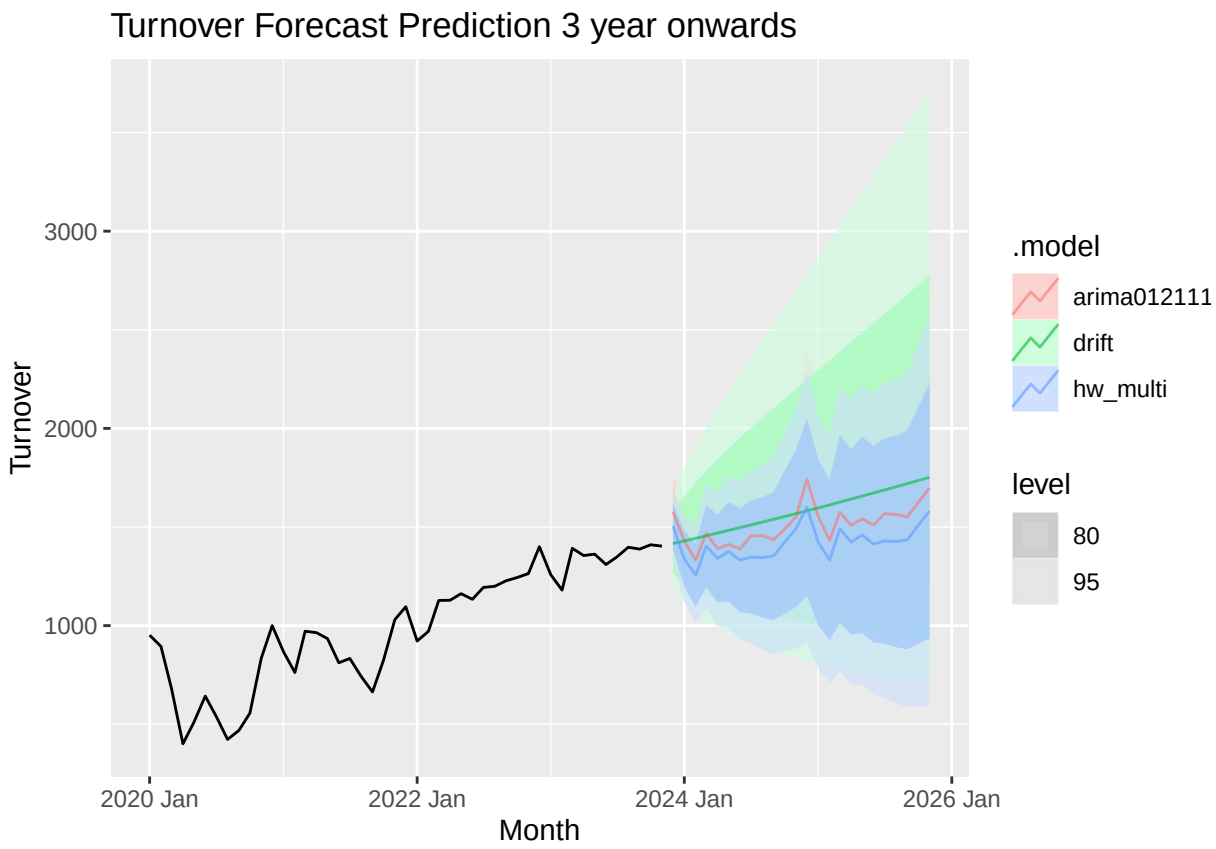
Model	.type	ME	RMSE	MAPE
Drift	Test	-319.47849	364.6433	43.07454
ETS	Test	-49.45844	236.5708	26.34269
ARIMA	Test	-58.38439	243.4702	27.12925

Forecasting on our current time period, we can see that the ARIMA and ETS model have similar patterns, but our ETS model has a lower RMSE value.

4.1 Forecasting next 3 years

```
all_model_fit <- fs_turnover %>%
  model(
    drift = RW(log(Turnover) ~ drift()),
    arima012111 = ARIMA(log(Turnover) ~ pdq(2,1,2) + PDQ(1,1,1)),
    hw_multi = ETS(Turnover ~ error("M") + trend("A") + season("M"))
        )
all_model_fc <- all_model_fit %>%
  forecast(h=24)

all_model_fc  %>%
  autoplot(filter(fs_turnover, year(Month)>=2020), alpha=0.6) +
  ggtitle("Turnover Forecast Prediction 3 year onwards")
```



```r
accuracy_threeyears <- models_final %>%
  forecast(h = "3 years") %>%
  accuracy(fs_test) %>%
  select(.model, .type, ME, RMSE, MAPE)

kable(accuracy_threeyears, format = "markdown", caption = "Forecasting Accuracy Metrics for the next 3 Y
```

Table 3: Forecasting Accuracy Metrics for the next 3 Years

.model	.type	ME	RMSE	MAPE
ARIMA_212_011	Test	-138.7808	253.5340	30.78510
ETS_MAM	Test	-125.4994	257.5440	31.29655
drift	Test	-360.7494	403.1124	51.98983

Drift: Widest prediction intervals as indicated by the large shaded green area. This indicates a higher level of uncertainty in its forecasts.

ARIMA: Narrower prediction interval as its red shade is smaller, indicated lower level of uncertainty.

ETS(M,A,M) Holt Winters Multiplicative: Much narrower prediction that expands over time towards the end. This indicates its increasing in uncertainty as we project further into the future.

If we were to forecast over the next three years however, the RMSE value is much lower for the ARIMA model. What we have noticed is that the ETS and ARIMA RMSE values are quite close to each other, so either of these two models can be considered to forecast.