Covid-19 Impact and Prediction of Travel in California - Date Updated Through October 2020

Corinne Medeiros and Amy Nestingen

11/8/2020

knitr::opts\_chunk$set(echo = TRUE)  
library(knitr)

# Overview

We are tasked to predict travel in California based on the impact of Covid-19. We started our analysis in Python where we cleaned and analyzed the data. Now, we we apply a model to the clean data and predict amount of trips while understanding California based on confirmed Convid-19 cases and deaths.

## Import libraries

#imports  
suppressWarnings(library(stats))  
suppressWarnings(library(ggplot2))  
suppressWarnings(library(plyr))  
suppressWarnings(library(dplyr))

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:plyr':  
##   
## arrange, count, desc, failwith, id, mutate, rename, summarise,  
## summarize

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

suppressWarnings(library(readr))  
suppressWarnings(library(forecast))

## Registered S3 method overwritten by 'xts':  
## method from  
## as.zoo.xts zoo

## Registered S3 method overwritten by 'quantmod':  
## method from  
## as.zoo.data.frame zoo

suppressWarnings(library(fpp2))

## -- Attaching packages ---------------------------------------------- fpp2 2.4 --

## v fma 2.4 v expsmooth 2.3

##

suppressWarnings(library(TTR))  
suppressWarnings(library(tidyr))

## Read in Data sets and Combine

In this section, we read in the data, did a couple of cleaning tweaks, grouped the data to the appropriate level and combined the data sets. The data is grouped by YearMonth and County name. We are predicting the number of trips in California so this field is summed. The new casese of Covid and the new deaths are also summed.

#Set Working Directory and Check  
setwd('C:/Users/nesti/OneDrive/Bellevue/DSC 630/DSC 630 Project')  
getwd()

## [1] "C:/Users/nesti/OneDrive/Bellevue/DSC 630/DSC 630 Project"

#Read in Trips Data   
dfTrip <- read.csv("New\_Trips\_by\_Distance\_CA\_clean.csv")  
dfCovid <- read.csv("New\_covid\_data\_CA\_clean.csv")  
  
  
  
#Trip: Print Table  
#head(dfTrip)  
  
#Trip:Remove County from County name  
  
dfTrip$County.Name <- as.character(dfTrip$County.Name)  
dfTrip$County.Name = substr(dfTrip$County.Name,1,nchar(dfTrip$County.Name)-7)  
#head(dfTrip)  
  
  
#Trip;Format Date  
dfTrip$Date <- as.Date(dfTrip$Date,format = "%m/%d/%Y")  
mode(dfTrip$Date)

## [1] "numeric"

dfTrip$YearMonth<-format(dfTrip$Date,"%Y-%m")  
#head(dfTrip)  
  
  
#Trip: Group by Month  
  
grp\_dfTrip <- group\_by(dfTrip, YearMonth, County.Name) %>%   
 summarize(sum\_trips = sum(Number.of.Trips)  
 )  
head(grp\_dfTrip)

## # A tibble: 6 x 3  
## # Groups: YearMonth [1]  
## YearMonth County.Name sum\_trips  
## <chr> <chr> <int>  
## 1 2019-01 Alameda 176090786  
## 2 2019-01 Alpine 109910  
## 3 2019-01 Amador 3630102  
## 4 2019-01 Butte 25274610  
## 5 2019-01 Calaveras 4538327  
## 6 2019-01 Colusa 2552683

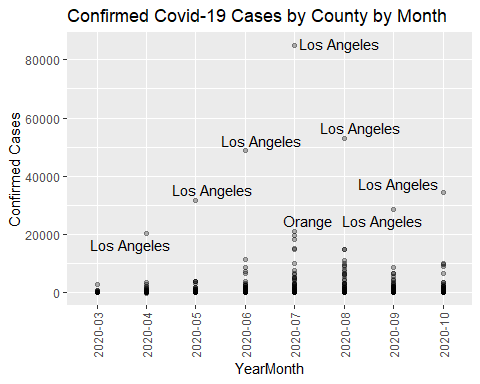
#Covid: Print table  
head(dfCovid)

## X county totalcountconfirmed totalcountdeaths newcountconfirmed  
## 1 0 Santa Clara 151 6 151  
## 2 1 Santa Clara 183 8 32  
## 3 2 Santa Clara 246 8 63  
## 4 3 Santa Clara 269 10 23  
## 5 4 Santa Clara 284 13 15  
## 6 5 Santa Clara 336 13 52  
## newcountdeaths date  
## 1 6 3/18/2020  
## 2 2 3/19/2020  
## 3 0 3/20/2020  
## 4 2 3/21/2020  
## 5 3 3/22/2020  
## 6 0 3/23/2020

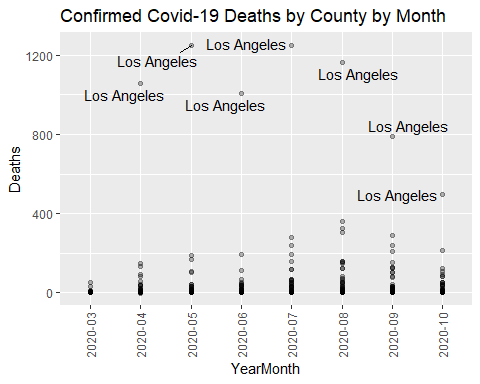
#Covid: Format Date  
dfCovid$date <- as.Date(dfCovid$date,format = "%m/%d/%Y")  
mode(dfCovid$date)

## [1] "numeric"

dfCovid$YearMonth<-format(dfCovid$date,"%Y-%m")  
#head(dfCovid)  
  
#rename so merge correctly  
names(dfCovid) [2] <- "County.Name"  
names(dfCovid) [1] <- "ID"  
#head(dfCovid)  
  
  
#Covid: Group by Month  
grp\_dfCovid <- group\_by(dfCovid, YearMonth, County.Name) %>%   
 summarise(sum\_new.count.confirmed = sum(newcountconfirmed),  
 sum\_new.count.deaths = sum(newcountdeaths)  
 )  
  
library(ggrepel)  
#Plot Covid Data  
ggplot(data = grp\_dfCovid, aes(x = YearMonth, y = sum\_new.count.confirmed)) +   
 geom\_point(alpha = .3) +  
 theme(axis.text.x = element\_text(angle = 90, vjust = 0.5, hjust=1))+  
 ggtitle("Confirmed Covid-19 Cases by County by Month") + xlab("YearMonth") + ylab("Confirmed Cases") +  
 geom\_text\_repel(data = subset(grp\_dfCovid, sum\_new.count.confirmed > 20000), aes(label = County.Name))



ggplot(data = grp\_dfCovid, aes(x = YearMonth, y = sum\_new.count.deaths)) +   
 geom\_point(alpha = .3) +  
 theme(axis.text.x = element\_text(angle = 90, vjust = 0.5, hjust=1))+  
 ggtitle("Confirmed Covid-19 Deaths by County by Month") + xlab("YearMonth") + ylab("Deaths") +  
 geom\_text\_repel(data = subset(grp\_dfCovid, sum\_new.count.deaths > 400), aes(label = County.Name))



##Merge Data sets  
merge\_tc <- left\_join(grp\_dfTrip, grp\_dfCovid, by = NULL, copy = FALSE)

## Joining, by = c("YearMonth", "County.Name")

## Warning: Column `County.Name` joining character vector and factor, coercing into  
## character vector

head(merge\_tc)

## # A tibble: 6 x 5  
## # Groups: YearMonth [1]  
## YearMonth County.Name sum\_trips sum\_new.count.confirmed sum\_new.count.deaths  
## <chr> <chr> <int> <int> <int>  
## 1 2019-01 Alameda 176090786 NA NA  
## 2 2019-01 Alpine 109910 NA NA  
## 3 2019-01 Amador 3630102 NA NA  
## 4 2019-01 Butte 25274610 NA NA  
## 5 2019-01 Calaveras 4538327 NA NA  
## 6 2019-01 Colusa 2552683 NA NA

summary(merge\_tc)

## YearMonth County.Name sum\_trips   
## Length:1273 Length:1273 Min. :1.507e+04   
## Class :character Class :character 1st Qu.:4.597e+06   
## Mode :character Mode :character Median :2.044e+07   
## Mean :7.230e+07   
## 3rd Qu.:6.782e+07   
## Max. :1.487e+09   
##   
## sum\_new.count.confirmed sum\_new.count.deaths  
## Min. : 0.0 Min. : 0.00   
## 1st Qu.: 21.5 1st Qu.: 0.00   
## Median : 223.0 Median : 3.00   
## Mean : 2001.1 Mean : 38.16   
## 3rd Qu.: 1577.5 3rd Qu.: 23.50   
## Max. :85048.0 Max. :1252.00   
## NA's :810 NA's :810

#Change NA to 0  
  
merge\_tc <- merge\_tc %>% replace\_na(list(sum\_new.count.confirmed = 0, sum\_new.count.deaths = 0))  
  
merge\_tc

## # A tibble: 1,273 x 5  
## # Groups: YearMonth [22]  
## YearMonth County.Name sum\_trips sum\_new.count.confirmed sum\_new.count.deaths  
## <chr> <chr> <int> <dbl> <dbl>  
## 1 2019-01 Alameda 176090786 0 0  
## 2 2019-01 Alpine 109910 0 0  
## 3 2019-01 Amador 3630102 0 0  
## 4 2019-01 Butte 25274610 0 0  
## 5 2019-01 Calaveras 4538327 0 0  
## 6 2019-01 Colusa 2552683 0 0  
## 7 2019-01 Contra Costa 124728006 0 0  
## 8 2019-01 Del Norte 2954173 0 0  
## 9 2019-01 El Dorado 19948834 0 0  
## 10 2019-01 Fresno 113160851 0 0  
## # ... with 1,263 more rows

#Test and Train Data In this section we split the data into test and train groups. This is so we know how the models preform with fresh data. We also group the data one last time by YearMonth. This is because we are predicting travel in California, not by County.

#Spilt data into test and train  
dt = sort(sample(nrow(merge\_tc), nrow(merge\_tc)\*.7))  
dat\_train\_ug<-merge\_tc[dt,]  
dat\_test\_ug<-merge\_tc[-dt,]  
  
nrow(dat\_train\_ug); nrow(dat\_test\_ug)

## [1] 891

## [1] 382

dat\_train <- group\_by(dat\_train\_ug, YearMonth) %>%   
 summarise(  
 sum\_trips\_MM = sum(sum\_trips)/1000000  
 )  
  
dat\_test <- group\_by(dat\_test\_ug, YearMonth) %>%   
 summarise(  
 sum\_trips\_MM = sum(sum\_trips)/1000000  
 )  
  
nrow(dat\_train); nrow(dat\_test)

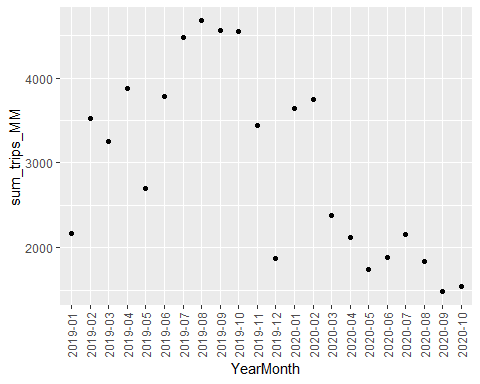
## [1] 22

## [1] 22

head(dat\_test)

## # A tibble: 6 x 2  
## YearMonth sum\_trips\_MM  
## <chr> <dbl>  
## 1 2019-01 2113.  
## 2 2019-02 259.  
## 3 2019-03 1423.  
## 4 2019-04 1190.  
## 5 2019-05 2421.  
## 6 2019-06 1361.

#Plot train data  
ggplot(data = dat\_train, aes(x = YearMonth, y = sum\_trips\_MM)) +   
 geom\_point() +  
 theme(axis.text.x = element\_text(angle = 90, vjust = 0.5, hjust=1))

 #Model

dat\_ts <- ts(dat\_train[, 2], start = c(2019, 1), end = c(2020,9), frequency = 12)  
  
head(dat\_ts)

## Jan Feb Mar Apr May Jun  
## 2019 2162.295 3518.996 3255.510 3880.595 2703.179 3785.853

#To Calculate Mape  
mape <- function(actual,pred){  
 mape <- mean(abs((actual - pred)/actual))\*100  
 return (mape)  
}

#Naive FOrecasting Model

naive\_mod <- naive(dat\_ts, h = 21)  
summary(naive\_mod)

##   
## Forecast method: Naive method  
##   
## Model Information:  
## Call: naive(y = dat\_ts, h = 21)   
##   
## Residual sd: 854.839   
##   
## Error measures:  
## ME RMSE MAE MPE MAPE MASE ACF1  
## Training set -34.16212 854.839 657.7825 -6.428313 23.84805 0.3833025 -0.1533892  
##   
## Forecasts:  
## Point Forecast Lo 80 Hi 80 Lo 95 Hi 95  
## Oct 2020 1479.052 383.53200 2574.573 -196.4014 3154.506  
## Nov 2020 1479.052 -70.24738 3028.352 -890.3971 3848.502  
## Dec 2020 1479.052 -418.44454 3376.549 -1422.9187 4381.023  
## Jan 2021 1479.052 -711.98832 3670.093 -1871.8552 4829.960  
## Feb 2021 1479.052 -970.60559 3928.710 -2267.3761 5225.481  
## Mar 2021 1479.052 -1204.41347 4162.518 -2624.9544 5583.059  
## Apr 2021 1479.052 -1419.42200 4377.527 -2953.7816 5911.886  
## May 2021 1479.052 -1619.54707 4577.652 -3259.8465 6217.951  
## Jun 2021 1479.052 -1807.50864 4765.613 -3547.3089 6505.414  
## Jul 2021 1479.052 -1985.28711 4943.392 -3819.1976 6777.302  
## Aug 2021 1479.052 -2154.37753 5112.482 -4077.7991 7035.904  
## Sep 2021 1479.052 -2315.94139 5274.046 -4324.8897 7282.994  
## Oct 2021 1479.052 -2470.90236 5429.007 -4561.8821 7519.987  
## Nov 2021 1479.052 -2620.00938 5578.114 -4789.9216 7748.026  
## Dec 2021 1479.052 -2763.87963 5721.984 -5009.9521 7968.057  
## Jan 2022 1479.052 -2903.02896 5861.134 -5222.7627 8180.867  
## Feb 2022 1479.052 -3037.89367 5995.998 -5429.0204 8387.125  
## Mar 2022 1479.052 -3168.84676 6126.951 -5629.2959 8587.401  
## Apr 2022 1479.052 -3296.21004 6254.315 -5824.0812 8782.186  
## May 2022 1479.052 -3420.26349 6378.368 -6013.8046 8971.909  
## Jun 2022 1479.052 -3541.25247 6499.357 -6198.8413 9156.946

dat\_test$naive = 36120.3  
  
mape(dat\_test$sum\_trips\_MM, dat\_test$naive)

## [1] 4134.328

#Simple Exponetial Smoothing

se\_model <- ses(dat\_ts, h = 22)  
summary(se\_model)

##   
## Forecast method: Simple exponential smoothing  
##   
## Model Information:  
## Simple exponential smoothing   
##   
## Call:  
## ses(y = dat\_ts, h = 22)   
##   
## Smoothing parameters:  
## alpha = 0.6499   
##   
## Initial states:  
## l = 2620.94   
##   
## sigma: 852.5427  
##   
## AIC AICc BIC   
## 351.2586 352.6704 354.3922   
##   
## Error measures:  
## ME RMSE MAE MPE MAPE MASE  
## Training set -72.44238 810.9298 660.5544 -9.103946 24.80748 0.3849178  
## ACF1  
## Training set 0.06448245  
##   
## Forecasts:  
## Point Forecast Lo 80 Hi 80 Lo 95 Hi 95  
## Oct 2020 1632.321 539.74369 2724.899 -38.63185 3303.274  
## Nov 2020 1632.321 329.30519 2935.337 -360.46977 3625.112  
## Dec 2020 1632.321 148.41559 3116.227 -637.11653 3901.759  
## Jan 2021 1632.321 -12.70189 3277.344 -883.52445 4148.167  
## Feb 2021 1632.321 -159.38917 3424.031 -1107.86330 4372.506  
## Mar 2021 1632.321 -294.94400 3559.586 -1315.17652 4579.819  
## Apr 2021 1632.321 -421.57176 3686.214 -1508.83696 4773.479  
## May 2021 1632.321 -540.83351 3805.476 -1691.23206 4955.874  
## Jun 2021 1632.321 -653.88231 3918.525 -1864.12528 5128.767  
## Jul 2021 1632.321 -761.59851 4026.241 -2028.86300 5293.505  
## Aug 2021 1632.321 -864.67233 4129.315 -2186.50080 5451.143  
## Sep 2021 1632.321 -963.65680 4228.299 -2337.88449 5602.527  
## Oct 2021 1632.321 -1059.00317 4323.645 -2483.70418 5748.346  
## Nov 2021 1632.321 -1151.08534 4415.728 -2624.53170 5889.174  
## Dec 2021 1632.321 -1240.21723 4504.859 -2760.84717 6025.489  
## Jan 2022 1632.321 -1326.66548 4591.308 -2893.05835 6157.701  
## Feb 2022 1632.321 -1410.65879 4675.301 -3021.51505 6286.157  
## Mar 2022 1632.321 -1492.39516 4757.037 -3146.52003 6411.162  
## Apr 2022 1632.321 -1572.04729 4836.689 -3268.33745 6532.980  
## May 2022 1632.321 -1649.76693 4914.409 -3387.19939 6651.842  
## Jun 2022 1632.321 -1725.68827 4990.330 -3503.31106 6767.953  
## Jul 2022 1632.321 -1799.93064 5064.573 -3616.85496 6881.497

df\_fc = as.data.frame(se\_model)  
dat\_test$simplexp = df\_fc$`Point Forecast`  
mape(dat\_test$sum\_trips\_MM, dat\_test$simplexp)

## [1] 100.9841

#ARIMA

arima\_model <- auto.arima(dat\_ts)

## Warning: The chosen seasonal unit root test encountered an error when testing for the first difference.  
## From stl(): series is not periodic or has less than two periods  
## 0 seasonal differences will be used. Consider using a different unit root test.

summary(arima\_model)

## Series: dat\_ts   
## ARIMA(0,1,0)   
##   
## sigma^2 estimated as 730750: log likelihood=-163.4  
## AIC=328.79 AICc=329.02 BIC=329.79  
##   
## Training set error measures:  
## ME RMSE MAE MPE MAPE MASE ACF1  
## Training set -32.43238 834.2376 626.5625 -6.117441 22.7172 0.36511 -0.1499519

fore\_arima = forecast::forecast(arima\_model, h=22)  
df\_arima = as.data.frame(fore\_arima)  
dat\_test$arima = df\_arima$`Point Forecast`  
mape(dat\_test$sum\_trips\_MM, dat\_test$arima) ## 2.1%

## [1] 85.52682

#Holt’s Trend

holt\_model <- holt(dat\_ts, h = 22)  
summary(holt\_model)

##   
## Forecast method: Holt's method  
##   
## Model Information:  
## Holt's method   
##   
## Call:  
## holt(y = dat\_ts, h = 22)   
##   
## Smoothing parameters:  
## alpha = 0.5788   
## beta = 1e-04   
##   
## Initial states:  
## l = 2999.1626   
## b = -59.4009   
##   
## sigma: 898.633  
##   
## AIC AICc BIC   
## 355.1342 359.1342 360.3568   
##   
## Error measures:  
## ME RMSE MAE MPE MAPE MASE  
## Training set -10.46485 808.5319 684.6232 -7.192363 25.02739 0.3989431  
## ACF1  
## Training set 0.08121489  
##   
## Forecasts:  
## Point Forecast Lo 80 Hi 80 Lo 95 Hi 95  
## Oct 2020 1567.6991 416.05456 2719.344 -193.5892 3328.987  
## Nov 2020 1508.2761 177.58730 2838.965 -526.8368 3543.389  
## Dec 2020 1448.8530 -39.54692 2937.253 -827.4583 3725.164  
## Jan 2021 1389.4300 -241.54842 3020.408 -1104.9363 3883.796  
## Feb 2021 1330.0070 -432.09439 3092.108 -1364.8945 4024.909  
## Mar 2021 1270.5840 -613.57801 3154.746 -1610.9931 4152.161  
## Apr 2021 1211.1610 -787.66019 3209.982 -1845.7722 4268.094  
## May 2021 1151.7379 -955.54943 3259.025 -2071.0800 4374.556  
## Jun 2021 1092.3149 -1118.15757 3302.787 -2288.3110 4472.941  
## Jul 2021 1032.8919 -1276.19272 3341.977 -2498.5483 4564.332  
## Aug 2021 973.4689 -1430.21777 3377.156 -2702.6526 4649.590  
## Sep 2021 914.0459 -1580.68897 3408.781 -2901.3218 4729.414  
## Oct 2021 854.6228 -1727.98221 3437.228 -3095.1307 4804.376  
## Nov 2021 795.1998 -1872.41155 3462.811 -3284.5597 4874.959  
## Dec 2021 735.7768 -2014.24261 3485.796 -3470.0149 4941.569  
## Jan 2022 676.3538 -2153.70236 3506.410 -3651.8436 5004.551  
## Feb 2022 616.9308 -2290.98663 3524.848 -3830.3451 5064.207  
## Mar 2022 557.5077 -2426.26572 3541.281 -4005.7800 5120.795  
## Apr 2022 498.0847 -2559.68888 3555.858 -4178.3764 5174.546  
## May 2022 438.6617 -2691.38775 3568.711 -4348.3358 5225.659  
## Jun 2022 379.2387 -2821.47914 3579.956 -4515.8368 5274.314  
## Jul 2022 319.8157 -2950.06727 3589.699 -4681.0387 5320.670

df\_holt = as.data.frame(holt\_model)  
dat\_test$holt = df\_holt$`Point Forecast`  
mape(dat\_test$sum\_trips\_MM, dat\_test$holt)

## [1] 52.48198

#TBATS

model\_tbats <- tbats(dat\_ts)  
summary(model\_tbats)

## Length Class Mode   
## lambda 0 -none- NULL   
## alpha 1 -none- numeric   
## beta 0 -none- NULL   
## damping.parameter 0 -none- NULL   
## gamma.values 0 -none- NULL   
## ar.coefficients 0 -none- NULL   
## ma.coefficients 0 -none- NULL   
## likelihood 1 -none- numeric   
## optim.return.code 1 -none- numeric   
## variance 1 -none- numeric   
## AIC 1 -none- numeric   
## parameters 2 -none- list   
## seed.states 1 -none- numeric   
## fitted.values 21 ts numeric   
## errors 21 ts numeric   
## x 21 -none- numeric   
## seasonal.periods 0 -none- NULL   
## y 21 ts numeric   
## call 2 -none- call   
## series 1 -none- character  
## method 1 -none- character

for\_tbats <- forecast::forecast(model\_tbats, h = 22)  
df\_tbats = as.data.frame(for\_tbats)  
dat\_test$tbats = df\_tbats$`Point Forecast`  
mape(dat\_test$sum\_trips\_MM, dat\_test$tbats)

## [1] 104.4489

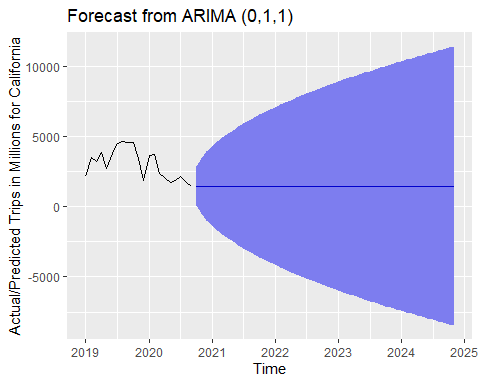
#Summary

We decided to investigate the ARIMA model further since that gave us the lowest MAPE. Below show the estimated trips to be traveled with a 90% confidence interval. As we can see, this is very hard to predict.

d.arima <- auto.arima(dat\_ts)

## Warning: The chosen seasonal unit root test encountered an error when testing for the first difference.  
## From stl(): series is not periodic or has less than two periods  
## 0 seasonal differences will be used. Consider using a different unit root test.

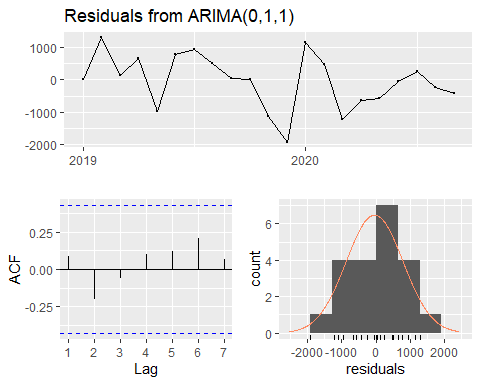
d.forecast <- forecast(d.arima, level = c(90), h = 50)  
autoplot(d.forecast, ylab = "Actual/Predicted Trips in Millions for California") + ggtitle('Forecast from ARIMA (0,1,1)')



fit <- Arima(dat\_ts, order=c(0,1,1))  
fit

## Series: dat\_ts   
## ARIMA(0,1,1)   
##   
## Coefficients:  
## ma1  
## -0.3209  
## s.e. 0.2789  
##   
## sigma^2 estimated as 727218: log likelihood=-162.89  
## AIC=329.78 AICc=330.49 BIC=331.77

checkresiduals(fit)



##   
## Ljung-Box test  
##   
## data: Residuals from ARIMA(0,1,1)  
## Q\* = 1.5757, df = 3, p-value = 0.6649  
##   
## Model df: 1. Total lags used: 4

autoplot(forecast(fit))

