sarimaTD_example

Graham Casey Gibson, Evan L. Ray, Nutcha Wattanachit

3/13/2020

An example of cdcForecast Utils for the sarimaTD model

First we load up sarimaTD and grab the cdc ILI data for Massachusetts.

```
library(sarimaTD)
library(cdcForecastUtils)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following object is masked from 'package:base':
##
##     date
```

```
library(dplyr)
```

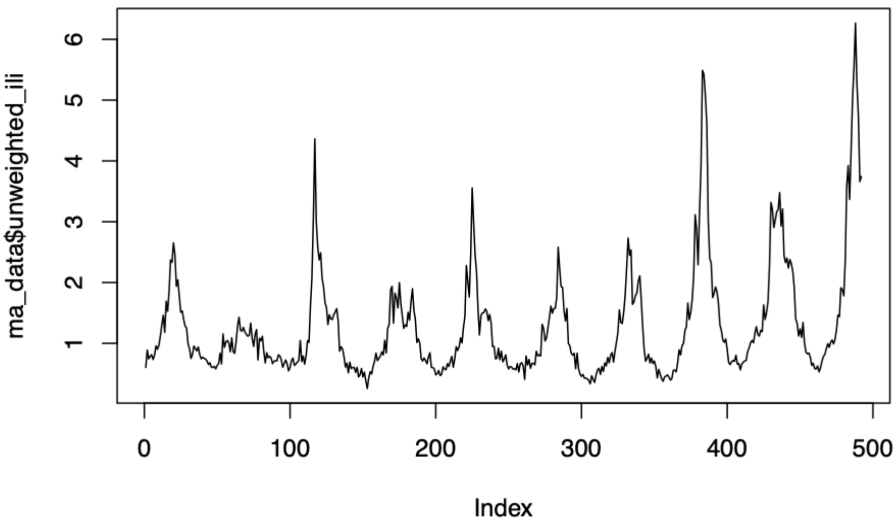
```
##
## Attaching package: 'dplyr'

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

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

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

```
flu_data <- download_and_preprocess_state_flu_data()
ma_data <- flu_data[flu_data$region == "Massachusetts",]
plot(ma_data$unweighted_ili,type='l')
```



We then fit sarimaTD up to the current date

```
sarimaFit <- sarimaTD::fit_sarima(tail(ma_data$unweighted_ili,200),
                        ts_frequency = 52)
```

Next, we generate a matrix of nsim by time till end of season. In order to do that we first get the current date and figure out how many steps there are left in the current season. We know that the season ends on YYYY-EW20.

```
current_date_in_cdc_format <-paste0(tail(ma_data$year,1),"-EW", ifelse(nchar(tail(ma_data$week,1))==2,ta
season_end <- "2020-EW20"
time_left_in_season <- get_time_left_in_season(current_date_in_cdc_format,season_end)


preds <- simulate(
        object = sarimaFit,
        nsim = 1000,
        seed = 1,
        newdata = ma_data$unweighted_ili,
        h = time_left_in_season
      )
```

We next append the observed data from the current season which we know starts on YYYY-EW40 to the predictions to create a trajectory matrix.

```
season_start <- "2019-EW40"
time_from_start_of_season <- get_time_from_start_of_season(season_start,current_date_in_cdc_format)

trajectory_matrix <- cbind(matrix(rep(tail(ma_data$unweighted_ili,time_from_start_of_season),1000),nrow
```

Finally, we convert the predicted trajectory matrix to a submission data frame.

```
submission_df <- trajectories_to_binned_distributions(trajectories = trajectory_matrix,
                                                      bins = c(seq(0,13,by=.1),100),
                                                      season_start_ew = season_start,
                                                             season_end_ew = season_end,
                                                             cdc_report_ew = current_date_in_cdc_format
                                                             h_max = 6)

head(submission_df)
```

```
##   bin value     target type forecast_week
## 1   0     0 1 wk ahead  Bin     2020-EW10
## 2 0.1     0 1 wk ahead  Bin     2020-EW10
## 3 0.2     0 1 wk ahead  Bin     2020-EW10
## 4 0.3     0 1 wk ahead  Bin     2020-EW10
## 5 0.4     0 1 wk ahead  Bin     2020-EW10
## 6 0.5     0 1 wk ahead  Bin     2020-EW10
```

```
generate_csv_from_submission_df(submission_df,"./")
```

Multiple States

```
states <- unique(flu_data$region)[1:5]
states
```

```
## [1] "Alabama"    "Alaska"     "Arizona"    "Arkansas"   "California"
```

Function to do all the steps to get the trajectories matrix for a single state:

```
get_trajectories_one_state <- function(state, flu_data) {
  # subset to state data
  state_data <- flu_data[flu_data$region == state,]

  # fit sarima model
  sarimaFit <- sarimaTD::fit_sarima(tail(state_data$unweighted_ili,200),
    ts_frequency = 52)

  # times (could really be done once outside this function)
  current_date_in_cdc_format <-paste0(tail(state_data$year,1),"-EW", ifelse(nchar(tail(state_data$week,
  season_end <- "2020-EW20"
  time_left_in_season <- get_time_left_in_season(current_date_in_cdc_format,season_end)

  # predictions
  preds <- simulate(
          object = sarimaFit,
          nsim = 1000,
          seed = 1,
          newdata = state_data$unweighted_ili,
          h = time_left_in_season
        )

  # prepend observed data
  season_start <- "2019-EW40"
  time_from_start_of_season <- get_time_from_start_of_season(season_start,current_date_in_cdc_format)

  trajectory_matrix <- cbind(matrix(rep(tail(ma_data$unweighted_ili,time_from_start_of_season),1000),nr

  trajectory_matrix[trajectory_matrix < 0.0] <- 0.0
  return(trajectory_matrix)
}
```

Call the function once for each state; assemble matrices in a tibble

```
trajectories_by_state <- tibble(
  state = states
) %>%
  mutate(
    trajectories = purrr::map(
      states,
      get_trajectories_one_state,
      flu_data = flu_data)
  )
```

```
trajectories_by_state
```

```
## # A tibble: 5 x 2
##   state      trajectories
##   <chr>      <list>
## 1 Alabama    <dbl[,34] [1,000 x 34]>
## 2 Alaska     <dbl[,34] [1,000 x 34]>
## 3 Arizona    <dbl[,34] [1,000 x 34]>
## 4 Arkansas   <dbl[,34] [1,000 x 34]>
## 5 California <dbl[,34] [1,000 x 34]>
```

```
submission_df <- multi_trajectories_to_binned_distributions(
  multi_trajectories = trajectories_by_state,
  bins = c(seq(0, 13, by = .1), 100),
  season_start_ew = season_start,
  season_end_ew = season_end,
  cdc_report_ew = current_date_in_cdc_format,
  h_max = 6)
```

```
head(submission_df)
```

```
## # A tibble: 6 x 6
##   state   bin   value target     type  forecast_week
##   <chr>   <chr> <dbl> <chr>      <chr> <chr>
## 1 Alabama 0         0 1 wk ahead Bin   2020-EW10
## 2 Alabama 0.1       0 1 wk ahead Bin   2020-EW10
## 3 Alabama 0.2       0 1 wk ahead Bin   2020-EW10
## 4 Alabama 0.3       0 1 wk ahead Bin   2020-EW10
## 5 Alabama 0.4       0 1 wk ahead Bin   2020-EW10
## 6 Alabama 0.5       0 1 wk ahead Bin   2020-EW10
```

```
submission_df %>%
  distinct(state, target) %>%
  as.data.frame()
```

```
##         state          target
## 1     Alabama      1 wk ahead
## 2     Alabama      2 wk ahead
## 3     Alabama      3 wk ahead
## 4     Alabama      4 wk ahead
## 5     Alabama      5 wk ahead
## 6     Alabama      6 wk ahead
## 7     Alabama       Peak Week
## 8     Alabama Peak Percentage
## 9      Alaska      1 wk ahead
## 10     Alaska      2 wk ahead
## 11     Alaska      3 wk ahead
## 12     Alaska      4 wk ahead
## 13     Alaska      5 wk ahead
## 14     Alaska      6 wk ahead
## 15     Alaska       Peak Week
## 16     Alaska Peak Percentage
## 17    Arizona      1 wk ahead
## 18    Arizona      2 wk ahead
## 19    Arizona      3 wk ahead
## 20    Arizona      4 wk ahead
## 21    Arizona      5 wk ahead
## 22    Arizona      6 wk ahead
## 23    Arizona       Peak Week
## 24    Arizona Peak Percentage
## 25   Arkansas      1 wk ahead
## 26   Arkansas      2 wk ahead
## 27   Arkansas      3 wk ahead
## 28   Arkansas      4 wk ahead
## 29   Arkansas      5 wk ahead
## 30   Arkansas      6 wk ahead
## 31   Arkansas       Peak Week
## 32   Arkansas Peak Percentage
## 33 California      1 wk ahead
## 34 California      2 wk ahead
## 35 California      3 wk ahead
## 36 California      4 wk ahead
## 37 California      5 wk ahead
## 38 California      6 wk ahead
## 39 California       Peak Week
## 40 California Peak Percentage
```

```
submission_df$location <-submission_df$state
head(submission_df)
```

```
## # A tibble: 6 x 7
##   state   bin   value target     type  forecast_week location
##   <chr>   <chr> <dbl> <chr>      <chr> <chr>         <chr>
## 1 Alabama 0         0 1 wk ahead Bin   2020-EW10     Alabama
## 2 Alabama 0.1       0 1 wk ahead Bin   2020-EW10     Alabama
## 3 Alabama 0.2       0 1 wk ahead Bin   2020-EW10     Alabama
## 4 Alabama 0.3       0 1 wk ahead Bin   2020-EW10     Alabama
## 5 Alabama 0.4       0 1 wk ahead Bin   2020-EW10     Alabama
## 6 Alabama 0.5       0 1 wk ahead Bin   2020-EW10     Alabama
```

```
generate_csv_from_submission_df(submission_df,"./")
sub_file<-read_entry("./2020-EW10.csv")
```

```
## Warning in format(round(as.numeric(bin[!is.na(bin) & bin != "none"]), 1), : NAs
## introduced by coercion
```

```
## Warning in cdcForecastUtils::verify_colnames(entry, check_week = F): These extra
## columns are ignored: xstate
```