sarimaTD_example

Graham Casey Gibson, Evan L. Ray, Nutcha Wattanachit

3/13/2020

An example of cdcForecast Utils for the sarimaTD model at the regional level

```
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
```

First we get the ILI data for the regions

```
flu_data <- download_and_preprocess_flu_data()
```

Next, we define the date parameters we need for the current challenge.

```
season_start <- "2020-EW10"
season_end <- "2020-EW35"
current_date_in_cdc_format <- get_current_date_from_flu_data(flu_data)
```

We also define a list of targets relevant to the spatial scale we are forecasting.

```
targets <-  c("wk ahead", "Peak height", "Peak week","First week below baseline","Below baseline for 3 
```

Next, we need a funtion that computes a "trajectory-matrix", which is an (nsim,time_left_in_season) matrix where each row is a forecast "trajectory". We first define this relative to a single geography.

```
get_trajectories_one_region <- function(location, flu_data) {
  # subset to current region
  region_data <- flu_data[flu_data$region == location,]

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

  # times (could really be done once outside this function)

  forecast_horizon <- get_required_forecast_horizon(
    targets = targets,
    h_max = 6,
    season_end_ew = season_end,
    cdc_report_ew = current_date_in_cdc_format
  )

  # predictions
  preds <- simulate(
          object = sarimaFit,
          nsim = 1000,
          seed = 1,
          newdata = region_data$weighted_ili,
          h = forecast_horizon
        )

  # compute time from start of season
  time_from_start_of_season <- get_time_from_start_of_season(season_start = season_start,current_time =
  #append observed data to trajectories
  # NOTE: this is where you would add a backfill model on the
  # observed data
  trajectory_matrix <- cbind(matrix(rep(tail(region_data$weighted_ili,time_from_start_of_season),1000),


  ## remove trajectories that do not respect ILI bounds
  trajectory_matrix[trajectory_matrix < 0.0] <- 0.0
  trajectory_matrix[trajectory_matrix > 100] <- 100

  return(trajectory_matrix)
}
```

```
trajectories_by_region <- tibble(
  location = c("Region 1","National")
) %>%
  mutate(
    trajectories = purrr::map(
      c("Region 1","National"),
      get_trajectories_one_region,
      flu_data = flu_data)
  )
```

We now have a list of trajectories that we can map to a submission data frame.

```
submission_df <- multi_trajectories_to_binned_distributions(
  multi_trajectories = trajectories_by_region,
  targets = targets,
  h_max = 6,
  bins = c(seq(0, 25, by = .1), 100),
  season_start_ew = season_start,
  season_end_ew = season_end,
  cdc_report_ew = current_date_in_cdc_format)
```

Finally, we add the point forecasts and verify

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

```
##    location                    target
## 1  Region 1                 1 wk ahead
## 2  Region 1                 2 wk ahead
## 3  Region 1                 3 wk ahead
## 4  Region 1                 4 wk ahead
## 5  Region 1                 5 wk ahead
## 6  Region 1                 6 wk ahead
## 7  Region 1                  Peak week
## 8  Region 1                Peak height
## 9  Region 1 Below baseline for 3 weeks
## 10 Region 1  First week below baseline
## 11 National                 1 wk ahead
## 12 National                 2 wk ahead
## 13 National                 3 wk ahead
## 14 National                 4 wk ahead
## 15 National                 5 wk ahead
## 16 National                 6 wk ahead
## 17 National                  Peak week
## 18 National                Peak height
## 19 National Below baseline for 3 weeks
## 20 National  First week below baseline
```

```
head(submission_df)
```

```
##   bin value     target type location
## 1   0     0 1 wk ahead  bin Region 1
## 2 0.1     0 1 wk ahead  bin Region 1
## 3 0.2     0 1 wk ahead  bin Region 1
## 4 0.3     0 1 wk ahead  bin Region 1
## 5 0.4     0 1 wk ahead  bin Region 1
## 6 0.5     0 1 wk ahead  bin Region 1
```

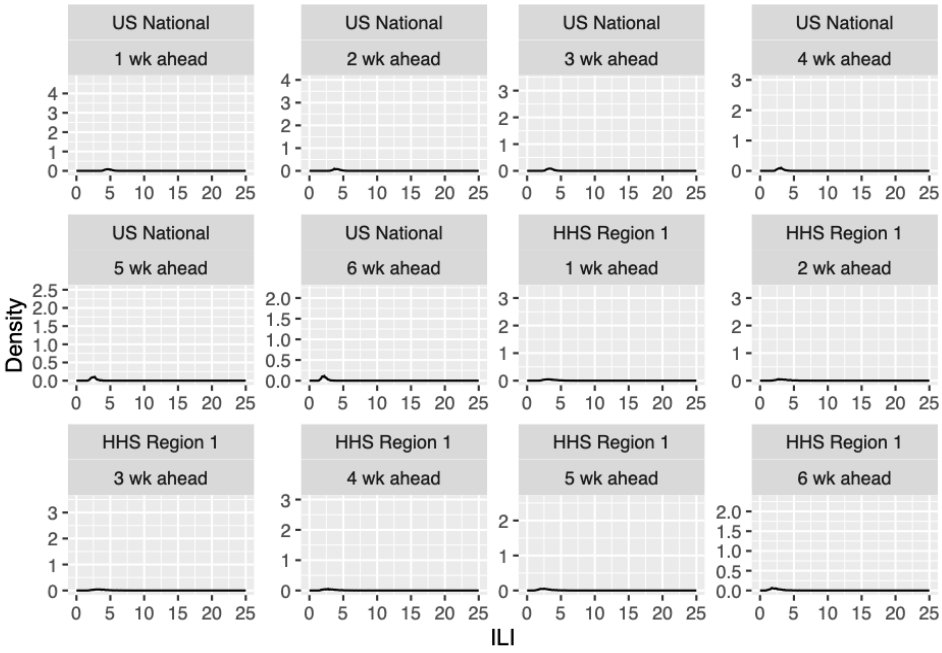
```
point_forecasts <- generate_point_forecasts(submission_df,method="Median")

submission_df_w_point_fcasts <- rbind(submission_df,point_forecasts)

submission_df_w_point_fcasts$location <- as.factor(submission_df_w_point_fcasts$location)
submission_df_w_point_fcasts$location <-
  dplyr::recode(submission_df_w_point_fcasts$location,"Region 1" = "HHS Region 1","National"="US Nationa
verify_entry(submission_df_w_point_fcasts,challenge='ilinet')
```

```
## Warning in cdcForecastUtils::verify_colnames(entry, check_week): Missing
## forecast_week - verification will proceed but forecast cannot be scored
```

```
## These locations have no forecast: HHS Region 2HHS Region 3HHS Region 4HHS Region 5HHS Region 6HHS Re
```

```
density_plots <- get_viz_from_submission_df(submission_df_w_point_fcasts)
density_plots[[1]]
```

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

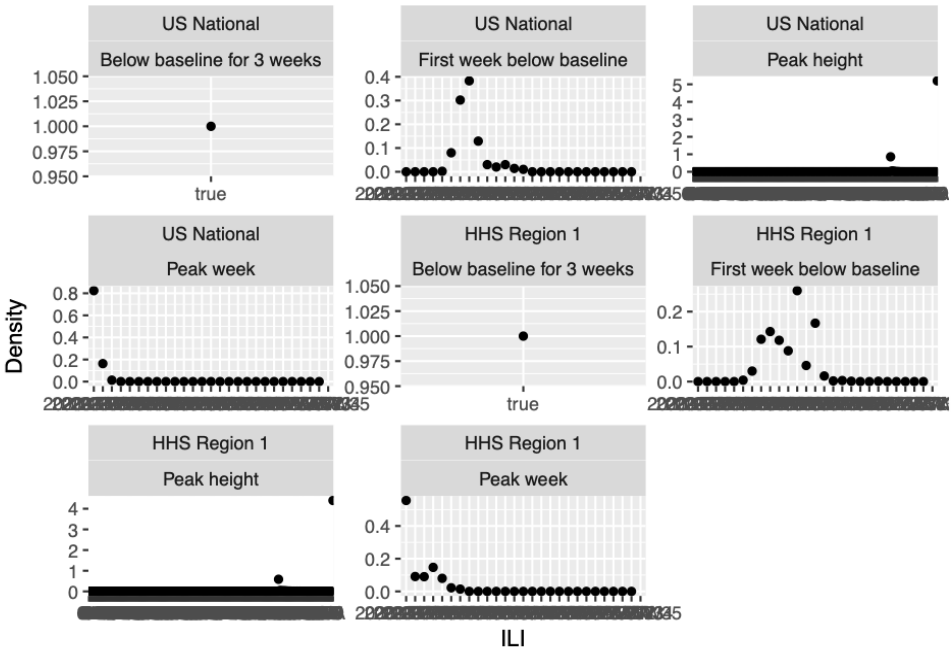



```
density_plots[[2]]
```

```
## Warning in FUN(X[[i]], ...): NAs introduced by coercion
```

```
## Warning in FUN(X[[i]], ...): NAs introduced by coercion
```

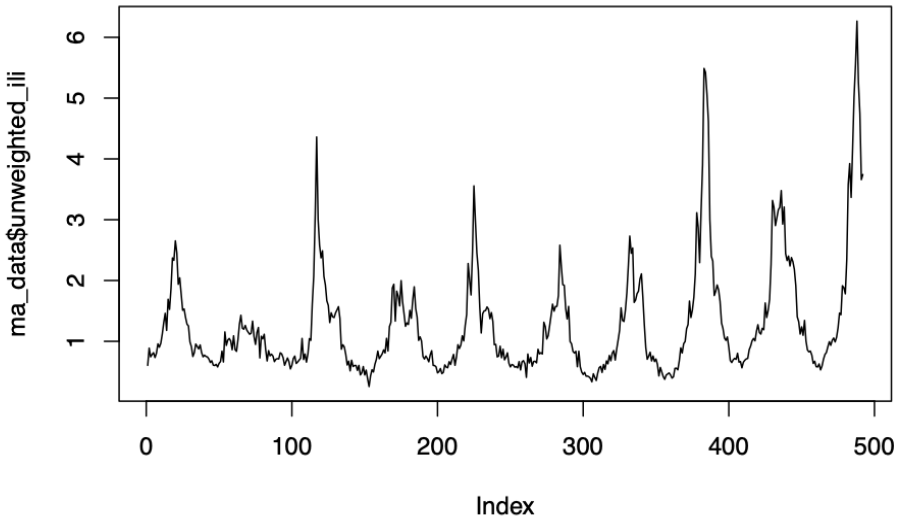
```
## Warning: Removed 4 rows containing missing values (geom_point).
```



An example of cdcForecast Utils for the sarimaTD model at the state level

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

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



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

  # fit sarima model
  sarimaFit <- sarimaTD::fit_sarima(tail(state_data$unweighted_ili,100),
    ts_frequency = 52)

  # times (could really be done once outside this function)

  forecast_horizon <- get_required_forecast_horizon(
    targets = c("wk ahead", "Peak height", "Peak week"),
    h_max = 6,
    season_end_ew = season_end,
    cdc_report_ew = current_date_in_cdc_format
  )

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

```
          h = forecast_horizon
        )

  # prepend observed data
  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
  trajectory_matrix[trajectory_matrix > 100] <- 100

  return(trajectory_matrix)
}
```

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

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

```
trajectories_by_state
```

```
## # A tibble: 5 x 2
##   location   trajectories
##   <chr>      <list>
## 1 Alabama    <dbl[,26] [1,000 x 26]>
## 2 Alaska     <dbl[,26] [1,000 x 26]>
## 3 Arizona    <dbl[,26] [1,000 x 26]>
## 4 Arkansas   <dbl[,26] [1,000 x 26]>
## 5 California <dbl[,26] [1,000 x 26]>
```

```
submission_df <- multi_trajectories_to_binned_distributions(
  multi_trajectories = trajectories_by_state,
  targets = c("wk ahead", "Peak height", "Peak week"),
  h_max = 6,
  bins = c(seq(0, 25, by = .1), 100),
  season_start_ew = season_start,
  season_end_ew = season_end,
  cdc_report_ew = current_date_in_cdc_format)
```

```
head(submission_df)
```

```
## # A tibble: 6 x 5
##   location bin   value target     type
##   <chr>    <chr> <dbl> <chr>      <chr>
## 1 Alabama  0         0 1 wk ahead bin
## 2 Alabama  0.1       0 1 wk ahead bin
## 3 Alabama  0.2       0 1 wk ahead bin
## 4 Alabama  0.3       0 1 wk ahead bin
## 5 Alabama  0.4       0 1 wk ahead bin
```

```
## 6 Alabama  0.5       0 1 wk ahead bin
```

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

```
##      location      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 height
## 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 height
## 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 height
## 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 height
## 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 height
```

```
head(submission_df)
```

```
## # A tibble: 6 x 5
##   location bin   value target     type
##   <chr>    <chr> <dbl> <chr>      <chr>
## 1 Alabama  0         0 1 wk ahead bin
## 2 Alabama  0.1       0 1 wk ahead bin
## 3 Alabama  0.2       0 1 wk ahead bin
```

```
## 4 Alabama  0.3       0 1 wk ahead bin
## 5 Alabama  0.4       0 1 wk ahead bin
## 6 Alabama  0.5       0 1 wk ahead bin
```

```
point_forecasts <- generate_point_forecasts(submission_df,method="Median")

submission_df_w_point_fcasts <- rbind(submission_df,point_forecasts)



verify_entry(submission_df_w_point_fcasts,challenge='state_ili')
```

```
## Warning in cdcForecastUtils::verify_colnames(entry, check_week): Missing
## forecast_week - verification will proceed but forecast cannot be scored
```

```
## These locations have no forecast: ColoradoConnecticutDelawareDistrict of ColumbiaGeorgiaHawaiiIdahoI
```

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

```
## Warning: Unknown or uninitialised column: 'forecast_week'.
```

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

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