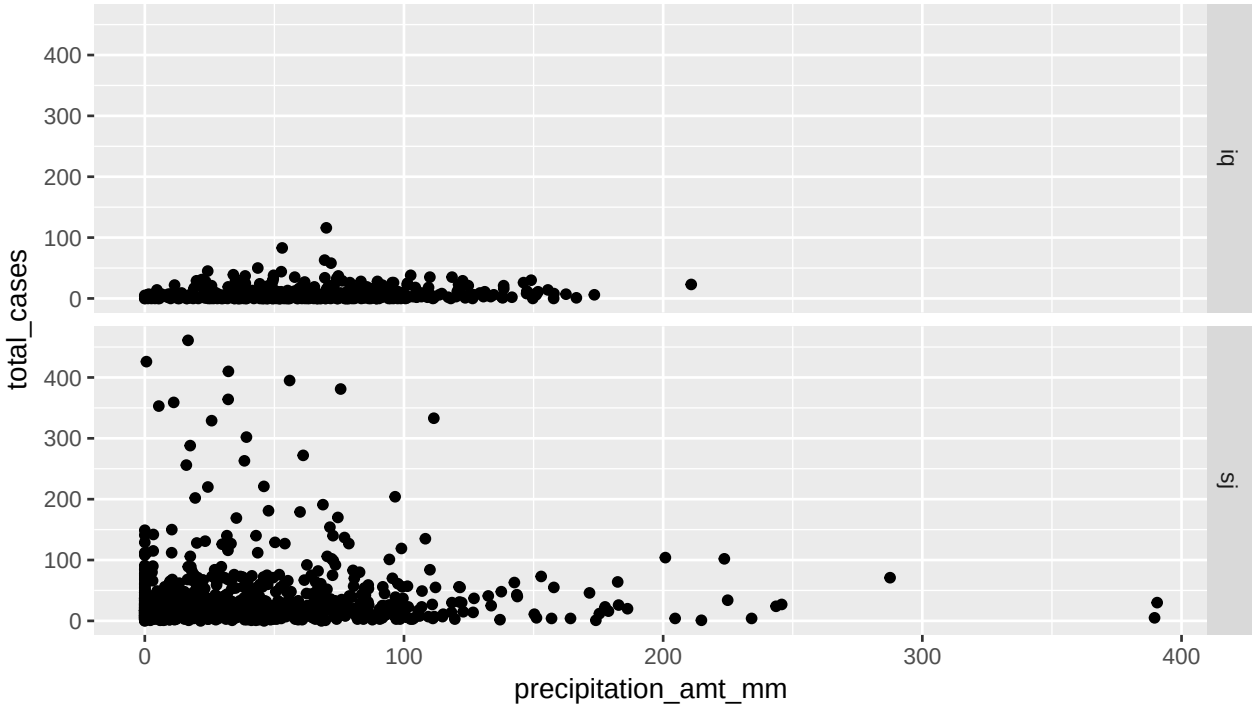
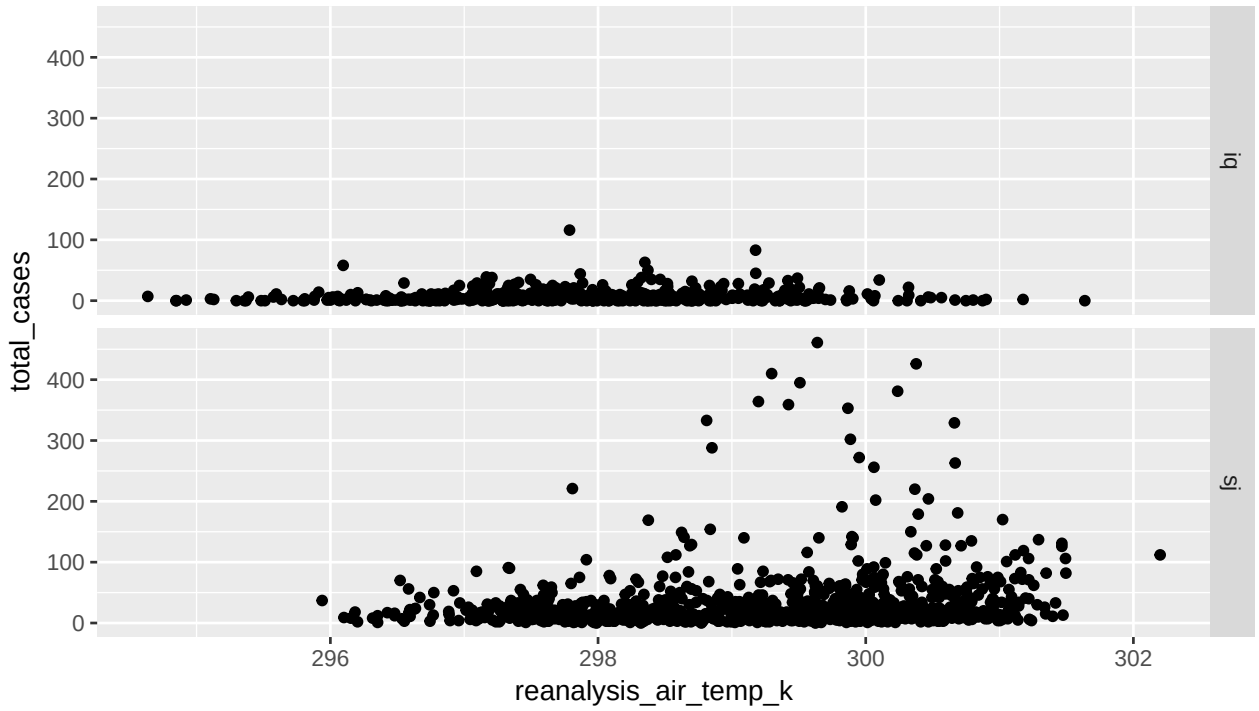
Data Exlporation

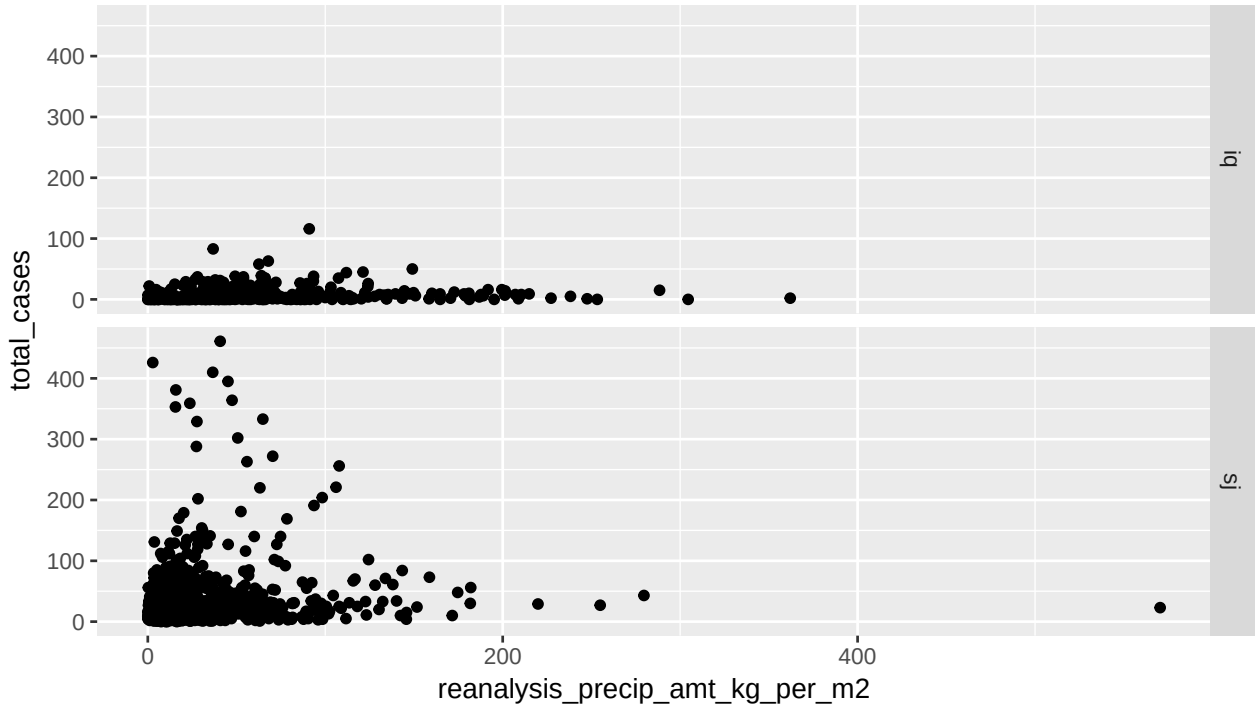
Jenn Havens and Madison Hobbs

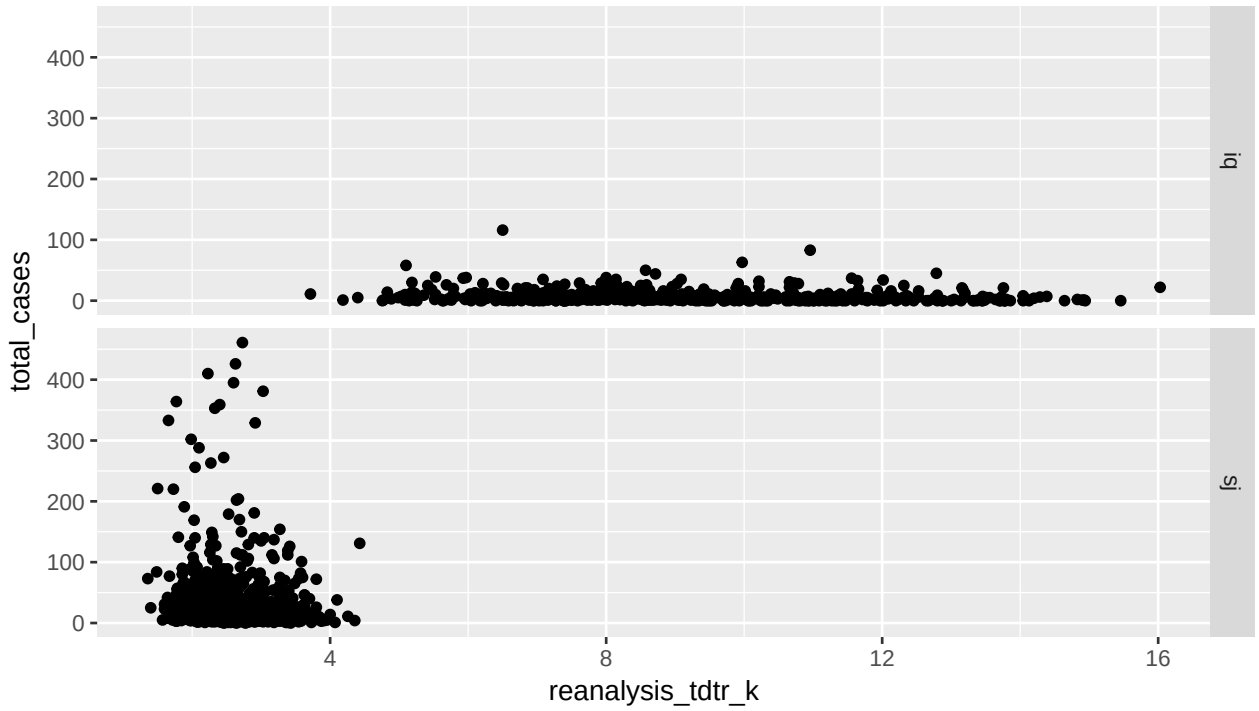
November & December 2017

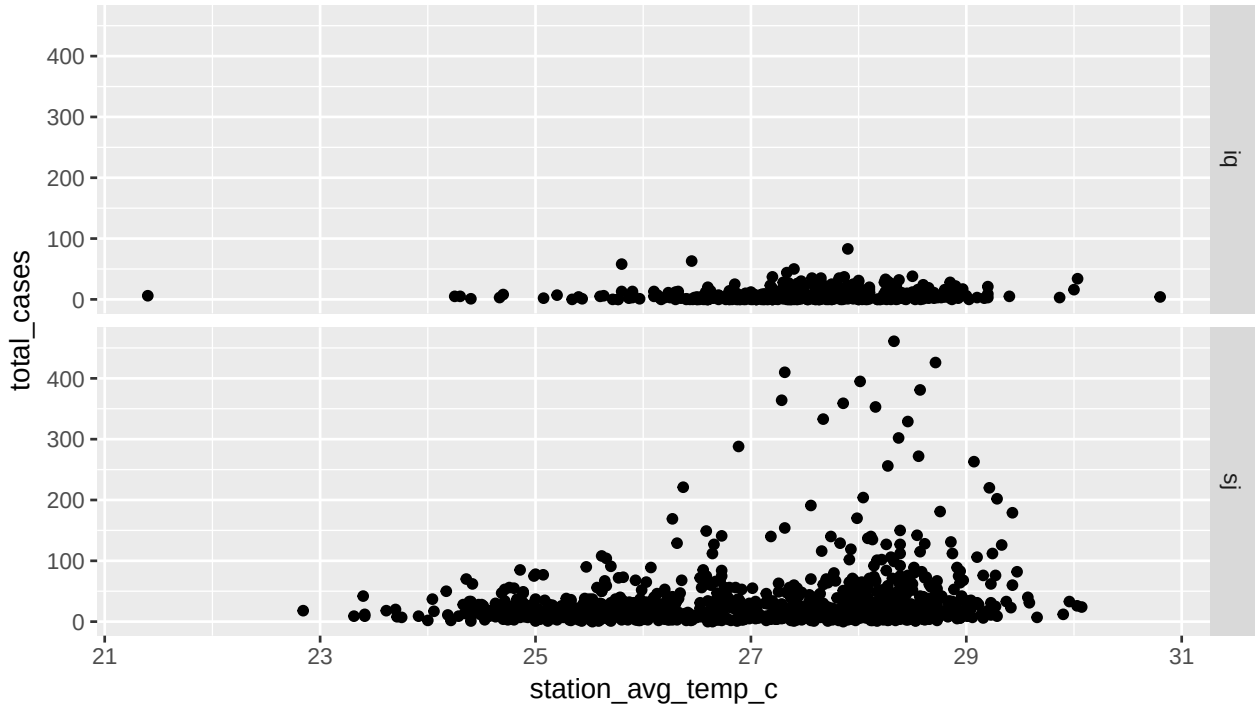
Preliminary Data Exploration

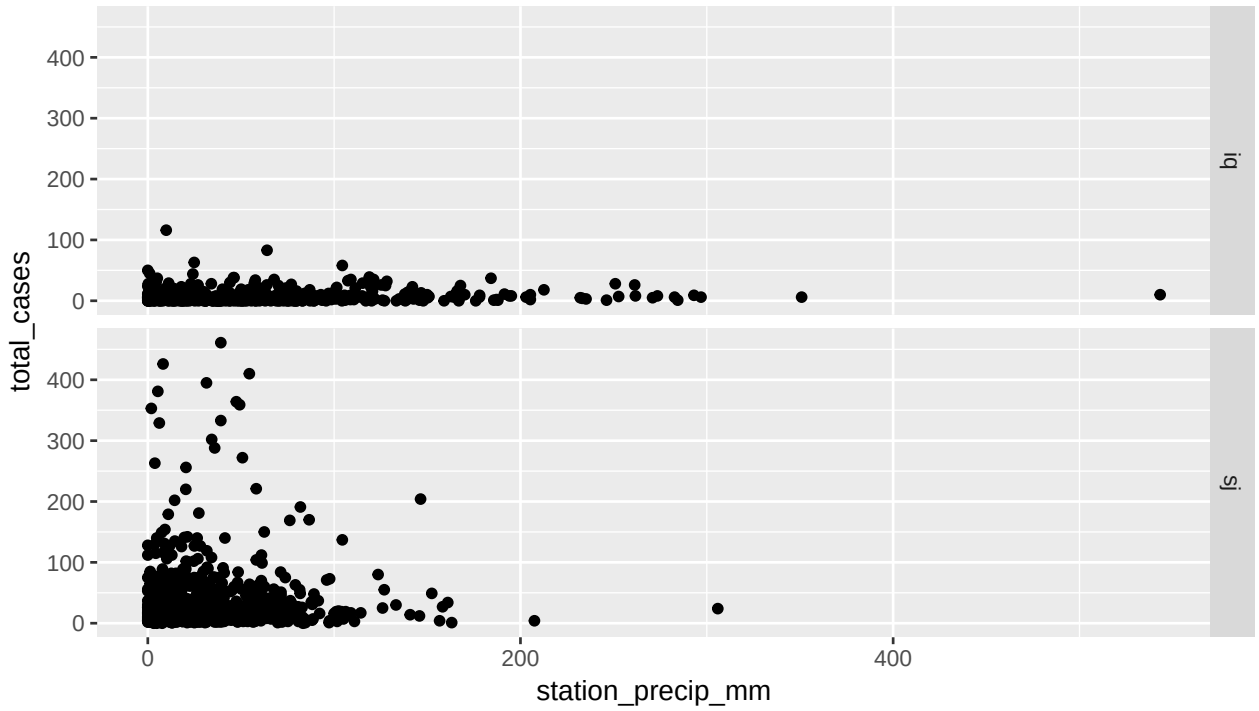


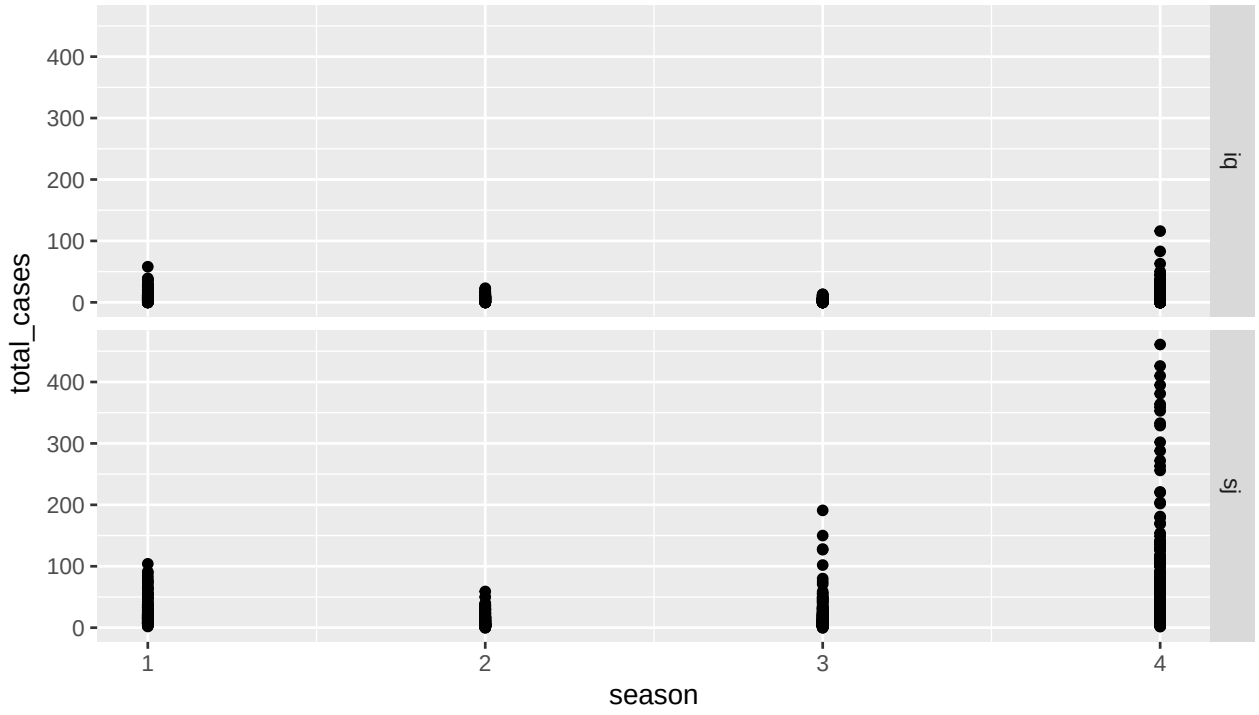

total_cases
400
300
200
100
0
iq
sj
296
298
300
302
reanalysis_air_temp_k

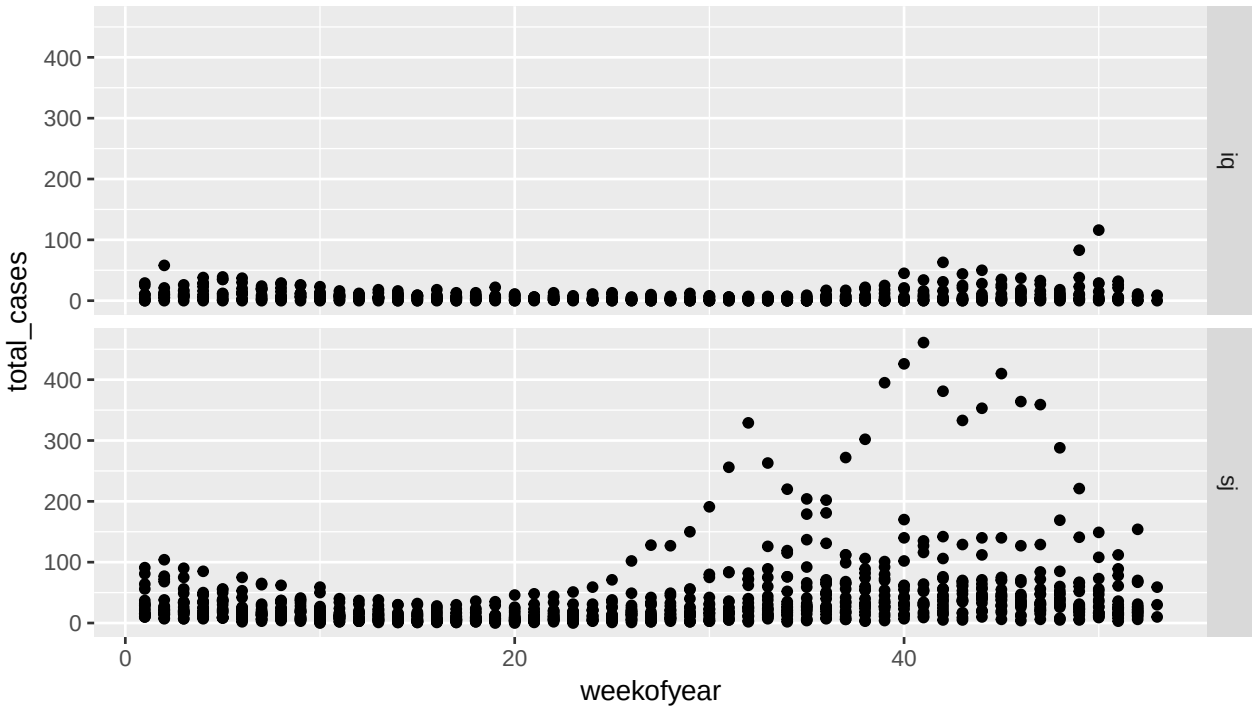

total_cases
400
300
200
100
0
iq
sj
0
200
400
reanalysis_precip_amt_kg_per_m2

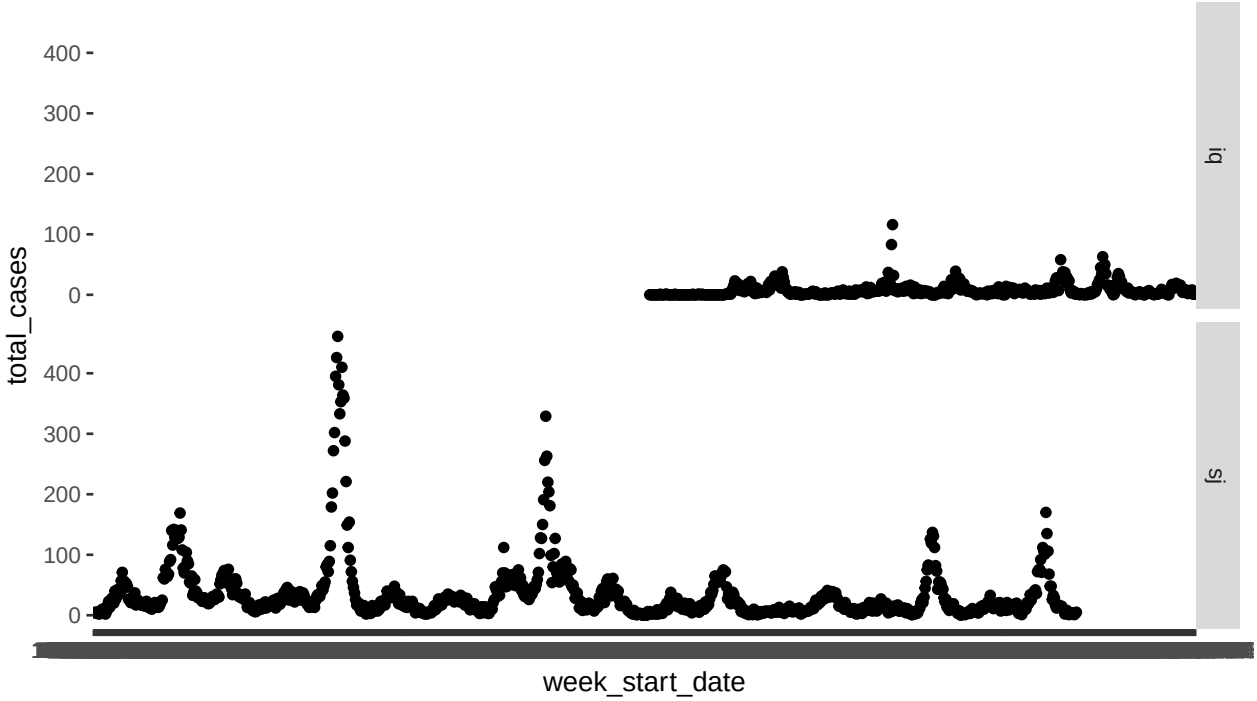

total_cases
400
300
200
100
0
4
8
12
16
reanalysis_tdtr_k
iq
sj


total_cases
400
300
200
100
0
21
23
25
27
29
31
station_avg_temp_c
iq
sj


total_cases
400
300
200
100
0
iq
sj
0
200
400
station_precip_mm


total_cases
400
300
200
100
0
iq
sj
1
2
3
4
season





Missing Values

We impute missing values using imputation via bagging (from the caret package). According to their documentation: " Imputation via bagging fits a bagged tree model for each predictor (as a function of all the others). This method is simple, accurate and accepts missing values, but it has much higher computational cost. "

```r
set.seed(6)
impTrainFull <- preProcess(TrainFull, "bagImpute")
TrainFull_noNA <- predict(impTrainFull, TrainFull)
head(TrainFull_noNA)
```

```
##   city year weekofyear total_cases week_start_date   ndvi_ne   ndvi_nw
## 1   sj 1990         18           4      1990-04-30 0.1226000 0.1037250
## 2   sj 1990         19           5      1990-05-07 0.1699000 0.1421750
## 3   sj 1990         20           4      1990-05-14 0.0322500 0.1729667
## 4   sj 1990         21           3      1990-05-21 0.1286333 0.2450667
## 5   sj 1990         22           6      1990-05-28 0.1962000 0.2622000
## 6   sj 1990         23           2      1990-06-04 0.1526769 0.1748500
##     ndvi_se   ndvi_sw precipitation_amt_mm reanalysis_air_temp_k
## 1 0.1984833 0.1776167               12.42              297.5729
## 2 0.1623571 0.1554857               22.82              298.2114
## 3 0.1572000 0.1708429               34.54              298.7814
## 4 0.2275571 0.2358857               15.36              298.9871
## 5 0.2512000 0.2473400                7.52              299.5186
## 6 0.2543143 0.1817429                9.58              299.6300
##   reanalysis_avg_temp_k reanalysis_dew_point_temp_k
## 1              297.7429                    292.4143
## 2              298.4429                    293.9514
## 3              298.8786                    295.4343
## 4              299.2286                    295.3100
## 5              299.6643                    295.8214
## 6              299.7643                    295.8514
##   reanalysis_max_air_temp_k reanalysis_min_air_temp_k
## 1                     299.8                     295.9
## 2                     300.9                     296.4
## 3                     300.5                     297.3
## 4                     301.4                     297.0
## 5                     301.9                     297.5
## 6                     302.4                     298.1
##   reanalysis_precip_amt_kg_per_m2 reanalysis_relative_humidity_percent
## 1                           32.00                              73.36571
## 2                           17.94                              77.36857
## 3                           26.10                              82.05286
## 4                           13.90                              80.33714
## 5                           12.20                              80.46000
## 6                           26.49                              79.89143
##   reanalysis_sat_precip_amt_mm reanalysis_specific_humidity_g_per_kg
## 1                        12.42                               14.01286
## 2                        22.82                               15.37286
## 3                        34.54                               16.84857
## 4                        15.36                               16.67286
## 5                         7.52                               17.21000
## 6                         9.58                               17.21286
##   reanalysis_tdtr_k station_avg_temp_c station_diur_temp_rng_c
## 1          2.628571           25.44286                6.900000
## 2          2.371429           26.71429                6.371429
## 3          2.300000           26.71429                6.485714
## 4          2.428571           27.47143                6.771429
## 5          3.014286           28.94286                9.371429
## 6          2.100000           28.11429                6.942857
```

```
##   station_max_temp_c station_min_temp_c station_precip_mm
## 1               29.4               20.0              16.0
## 2               31.7               22.2               8.6
## 3               32.2               22.8              41.4
## 4               33.3               23.3               4.0
## 5               35.0               23.9               5.8
## 6               34.4               23.9              39.1
```

```
anyNA(TrainFull_noNA)
```

```
## [1] FALSE
```

```
all_rf <- train(total_cases ~ ., data = TrainFull_noNA, method = "rf", na.action = na.pass, trControl =

all_rf
```

```
## Random Forest
##
## 1456 samples
##   24 predictor
##
## No pre-processing
## Resampling results across tuning parameters:
##
##   mtry  RMSE      Rsquared
##    1    43.46109  0.005496716
##    2    42.91572  0.030299008
##    3    41.78722  0.080626306
##    4    41.32945  0.100659319
##    5    40.74963  0.125716248
##    6    40.45380  0.138364318
##    7    40.25097  0.146982788
##    8    40.01611  0.156908294
##    9    39.70588  0.169929898
##   10    39.61550  0.173704649
##   11    39.49733  0.178626923
##   12    39.38332  0.183361718
##   13    39.11734  0.194354869
##   14    39.07597  0.196058208
##   15    38.85424  0.205156128
##   16    38.85580  0.205092088
##   17    38.66956  0.212694138
##   18    38.62754  0.214403938
##   19    38.48725  0.220099990
##   20    38.28585  0.228241154
##   21    37.92884  0.242567171
##   22    38.00809  0.239398437
##   23    38.02435  0.238747656
##   24    37.79381  0.247950357
##
## RMSE was used to select the optimal model using  the smallest value.
## The final value used for the model was mtry = 24.
```

Variable Selection

Temperature

We think that the station-measured temperature would be preferable, because these are not measured from satellite or estimated from a model.

We should also think about whether to choose minimum temperature, maximum temperature, average temperature, or diurnal temperature range. We suspect that these variables are redundant, and we can clearly see from the data below:

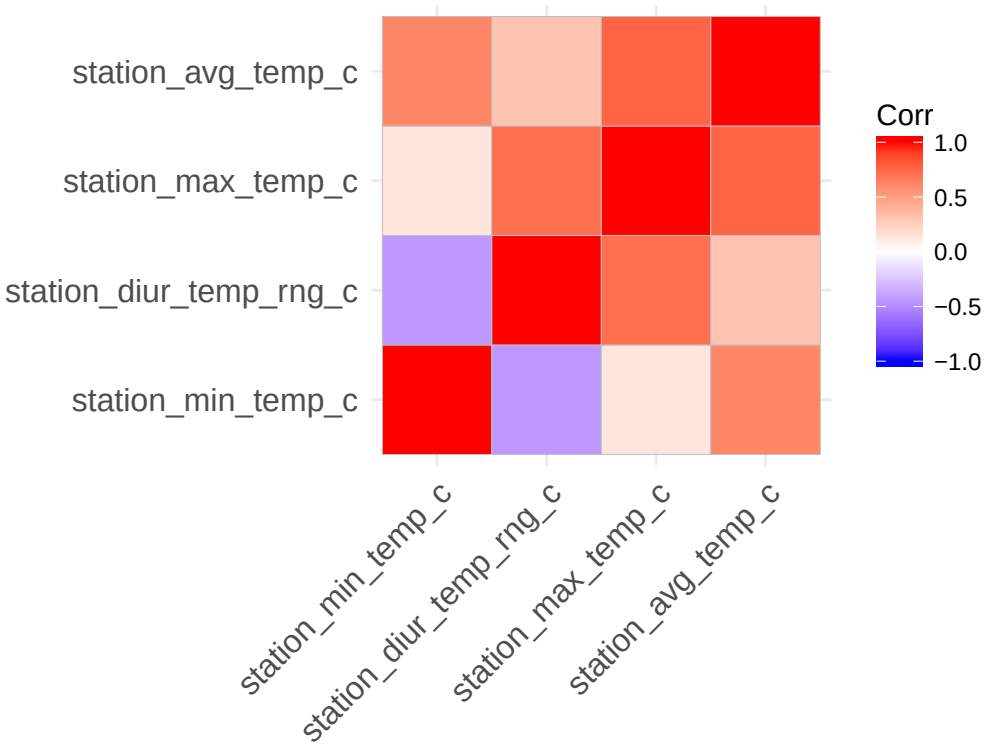
```
temps <- TrainFull_noNA %>% select(station_max_temp_c,station_avg_temp_c, station_min_temp_c, station_di

corr_matrix <- cor(temps)
corr_pmat <- cor_pmat(temps)
# they are all significantly correlated
all(corr_pmat < 0.05)
```

```
## [1] TRUE
```

All pairwise correlations betweent the four measures of temperature are significant. We can see these strong correlations represented in the correlation plot below:

```
ggcorrplot(corr_matrix, hc.order = TRUE)
```



Note that Choi et. al say : "Mean temperature was significantly associated with dengue incidence in all three provinces, but incidence did not correlate well with maximum temperature in Banteay Meanchey, nor with minimum temperature in Kampong Thom at a lag of three months in the negative binomial model." We are inclined to only use mean temperature.

Precipitation

Variable descriptions https://www.drivendata.org/competitions/44/dengai-predicting-disease-spread/page/82/.

There are three measures of precipitation. One is station_precip_mm which is the total daily precipitation as measured by NOAA's GHCN weather stations (https://www.ncdc.noaa.gov/ghcn-daily-description). Another is precipitation_amt_mm which represents total precipitation as measured by PERSIANN satellites. The third and forth, reanalysis_sat_precip_amt_mm and reanalysis_precip_amt_kg_per_m2 are both generated by NOAA's NCEP Climate Forecast System Reanalysis (https://climatedataguide.ucar.edu/climate-data/climate-forecast-system-reanalysis-cfsr).

We should choose one of these precipitation measures for the model, since these four precipitation measures all measure approximately the same thing. According to the NOAA Dengue Forecasting recommendations (http://dengueforecasting.noaa.gov), "remotely sensed observations are generally an excellent observation of precipitation and vegetation conditions for a location." With the multiple sources for total precipitation, we decide to use only the satellite measured total precipitation for each city to build our model.

Which Air Temperature Measures Should Be Included?

Note that Choi et. al say : "Mean temperature was significantly associated with dengue incidence in all three provinces, but incidence did not correlate well with maximum temperature in Banteay Meanchey, nor with minimum temperature in Kampong Thom at a lag of three months in the negative binomial model." I'm inclined to only use mean temperature.

Should humidity and Dewpoint be included?

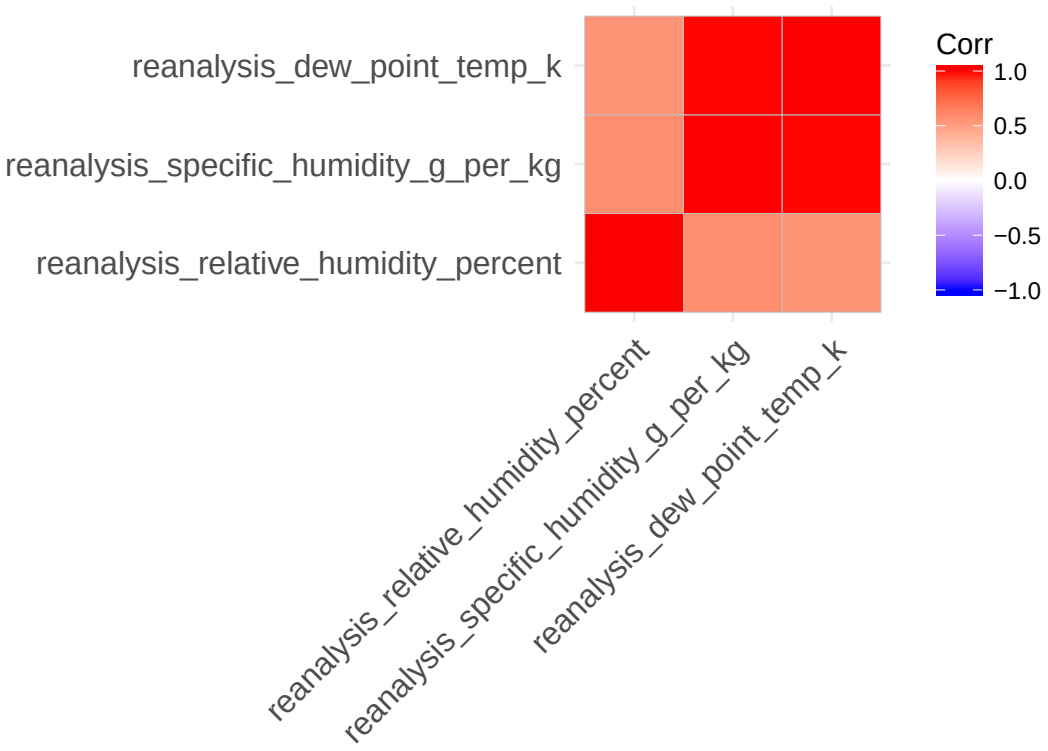
Specific humidity, relative humidity, and dew point are all provided and are all measured using NOAA's NCEP Climate Forecast System Reanalysis. All three measures are significantly correlated, as seen below.

```
humids <- TrainFull_noNA %>% select(reanalysis_relative_humidity_percent, reanalysis_specific_humidity_g

corr_matrix <- cor(humids)
corr_pmat <- cor_pmat(humids)
# they are all significantly correlated
all(corr_pmat < 0.05)
```

```
## [1] TRUE
```

```
ggcorrplot(corr_matrix, hc.order = TRUE)
```



Since these measure roughly the same information, we opt to include only one in our model. Relative humidity is the measure most often used in literature we have read. It is also an easy-to-find measure, making our model more user-friendly. Therefore, we use only relative humidity in the model.

Creating Data for Model and Lagged Data

We separate San Juan and Iquitos data to produce two models, one to predict weekly dengue fever cases in San Juan and the other in Iquitos. This is because weather and vegetation will behave differently in relation to time between both locations, because these are locations separated by distance, climate, and ecosystem.

At the same time, we create two lag variables for each site: temperature lag and precipitation lag. The lagged temperature and precipitation variables record, respectively, what the temperature or precipitation was 12 weeks prior for the observation at hand.

Time laged environmental variables have been shown to be a signifigant predictors. Specifically we consider temperature and relative humidity (Wu et al 2007).

```
lag_num_weeks = 12

#make on col which gives year and week of year info
tempData <- mutate(TrainFull_noNA, year_week_temp = paste(as.character(year), "_", as.character(weekofy

#make year_week and lag_year_week of yyyy_ww format
tempData <- mutate(tempData, year_week = ifelse(nchar(year_week_temp) == 7, year_week_temp, sub("_", "_

tempDataIQ <- filter(tempData, city == "iq")
tempDataSJ <- filter(tempData, city == "sj")

### San Juan Data for Model
#want lag for: precipitation_amt_mm and station_avg_temp_c for SJ
lag_week_temp <- c()
```

```
lag_week_precip <- c()
lag_week_humid <- c()
for (i in 1:length(tempDataSJ$lag_year_week)){
  if (sum(grepl(tempDataSJ$lag_year_week[i], tempDataSJ$year_week)) == 1){
    #find correct index for lag data and add that data to vectors
    lagWeeki <- grep(tempDataSJ$lag_year_week[i], tempDataSJ$year_week)
    lag_week_temp <- c(lag_week_temp, tempDataSJ$station_avg_temp_c[lagWeeki])
    lag_week_precip <- c(lag_week_precip, tempDataSJ$precipitation_amt_mm[lagWeeki])
    lag_week_humid <- c(lag_week_humid, tempDataSJ$reanalysis_relative_humidity_percent[lagWeeki])
  }else {
    lag_week_temp <- c(lag_week_temp, NA)
    lag_week_precip <- c(lag_week_precip, NA)
    lag_week_humid <- c(lag_week_humid, NA)
  }
}

tempDataSJ$temp_lag <- lag_week_temp
tempDataSJ$precip_lag <- lag_week_precip
tempDataSJ$humidity_lag <- lag_week_humid

# select only the predictors we want for the model
sj_data_for_model <- select(tempDataSJ, total_cases, year, weekofyear, ndvi_ne, ndvi_nw, ndvi_se, ndvi_


### Iquitos Data for Model
#want lag for: precipitation_amt_mm and station_avg_temp_c for IQ
lag_week_temp <- c()
lag_week_precip <- c()
lag_week_humid <- c()
for (i in 1:length(tempDataIQ$lag_year_week)){
  if (sum(grepl(tempDataIQ$lag_year_week[i], tempDataIQ$year_week)) == 1){
    #find correct index for lag data and add that data to vectors
    lagWeeki <- grep(tempDataIQ$lag_year_week[i], tempDataIQ$year_week)
    lag_week_temp <- c(lag_week_temp, tempDataIQ$station_avg_temp_c[lagWeeki])
    lag_week_precip <- c(lag_week_precip, tempDataIQ$precipitation_amt_mm[lagWeeki])
    lag_week_humid <- c(lag_week_humid, tempDataIQ$reanalysis_relative_humidity_percent[lagWeeki])
  }else {
    lag_week_temp <- c(lag_week_temp, NA)
    lag_week_precip <- c(lag_week_precip, NA)
    lag_week_humid <- c(lag_week_humid, NA)
  }
}

tempDataIQ$temp_lag <- as.numeric(lag_week_temp)
tempDataIQ$precip_lag <- as.numeric(lag_week_precip)
tempDataIQ$humidity_lag <- as.numeric(lag_week_humid)

# select only the predictors we want for the model
iq_data_for_model <- select(tempDataIQ, total_cases, year, weekofyear, ndvi_ne, ndvi_nw, ndvi_se, ndvi_
```

```
iq_data_for_model <- filter(iq_data_for_model, humidity_lag != "NA")
sj_data_for_model <- filter(sj_data_for_model, humidity_lag != "NA")

# get good names for columns
```

```
iq_data_for_model <- iq_data_for_model %>% rename(week_of_year = weekofyear)
sj_data_for_model <- sj_data_for_model %>% rename(week_of_year = weekofyear)
```

Building a Model

The Driven Data team uses Mean Absolute Error to measure error, so we will, too.

Model Assessment

```
# returns Mean Absolute Error
# error is defined as actual - predicted
MAE <- function(error)
{
    mean(abs(error))
}

# returns Root Mean Squared Error
# error is defined as actual - predicted
RMSE <- function(error)
{
  sqrt(mean(error^2))
}
```

As Kane et. al used in their time series Random Forest model, the model was assessed by doing the following: "The simulation begins by using data from the last 30 weeks of data from 2006 to predict the outbreak for the week of 2007 (2007-01-07). The simulation proceeds by iteratively adding a new week of data, training a new model based on the updated data, and predicting the number of outbreaks for the following week."

Moving in steps of 30 weeks, we train the model on 30 weeks, then predict the next week.

We'll assess our model's performance by computing the error (MSE) between the model's predicted number of cases and the actual number of cases for each week.

San Juan Model Assessment

```
step <- 30
predictions <- rep(NA, step)

index_to_predict = 31
while(index_to_predict <= length(sj_data_for_model$total_cases)) {

  # use previous 30 weeks to build model
  data <- slice(sj_data_for_model, (index_to_predict-step):(index_to_predict-1))

  model <- train(total_cases ~ ., data = data, method = "rf", trControl =  trainControl(method = "oob")

  # predict on single observation:  the next week, the index_to_predict
  predicted_num_cases <- predict(model, slice(sj_data_for_model, index_to_predict))

  predictions <- c(predictions, predicted_num_cases)
```

```
  index_to_predict = index_to_predict + 1
}

actual <- sj_data_for_model$total_cases[31:length(sj_data_for_model$total_cases)]
predicted <- predictions[31:length(predictions)]

RMSE(predicted - actual)
```

```
## [1] 26.83507
```

```
MAE(predicted - actual)
```

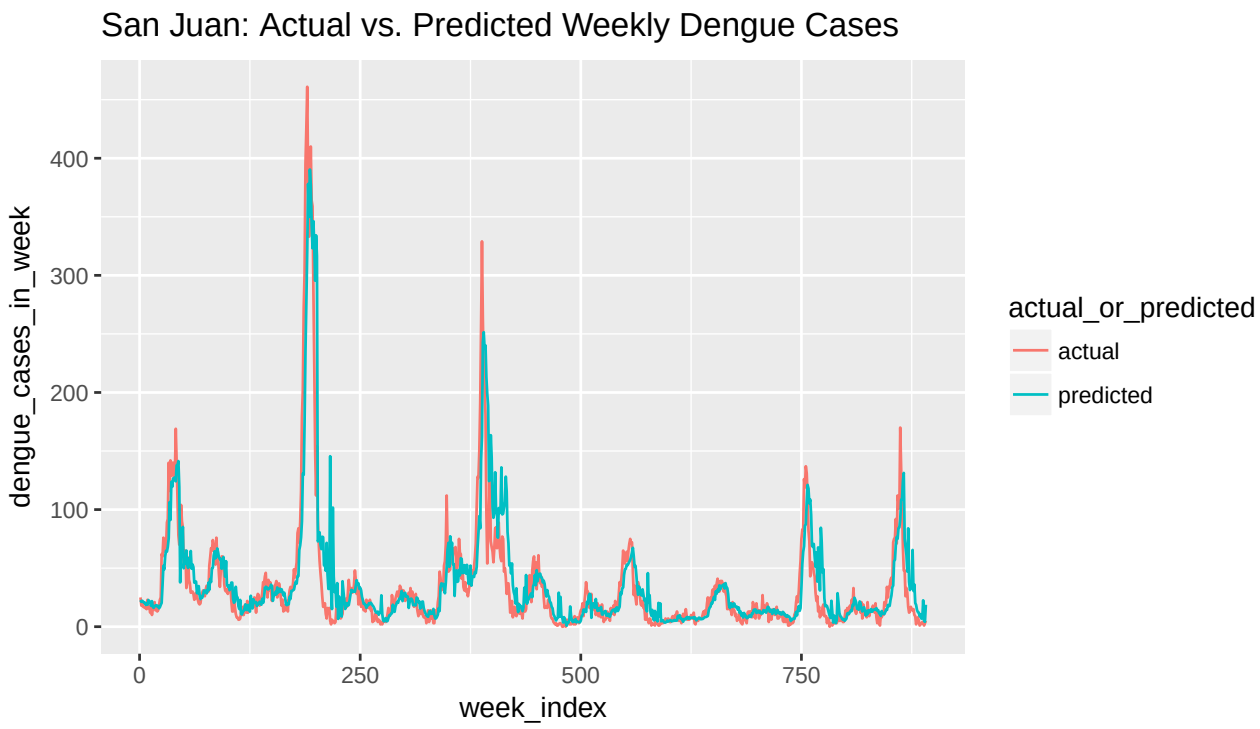
```
## [1] 14.35892
```

```
# ranges from 0 to 100, scaled to data
nrmse(predicted, actual)
```

```
## [1] 51.2
```

```
sj_prospective_preds <- data.frame(sj_data_for_model$year[31:length(sj_data_for_model$total_cases)], sj
colnames(sj_prospective_preds) = c("year", "week_of_year", "actual", "predicted")
sj_prospective_preds <- sj_prospective_preds %>% mutate(week_index = row_number()) %>% gather(actual_or

ggplot(sj_prospective_preds, aes(x = week_index, y = dengue_cases_in_week, col = actual_or_predicted)) +
```



Iquitos Model Assessment

```
step <- 30
predictions <- rep(NA, step)
```

```
index_to_predict = 31
while(index_to_predict <= length(iq_data_for_model$total_cases)) {

  # use previous 30 weeks to build model
  data <- slice(iq_data_for_model, (index_to_predict-step):(index_to_predict-1))

  model <- train(total_cases ~ ., data = data, method = "rf", trControl =  trainControl(method = "oob")

  # predict on single observation:  the next week, the index_to_predict
  predicted_num_cases <- predict(model, slice(iq_data_for_model, index_to_predict))

  predictions <- c(predictions, predicted_num_cases)

  index_to_predict = index_to_predict + 1
}

actual <- iq_data_for_model$total_cases[31:length(iq_data_for_model$total_cases)]
predicted <- predictions[31:length(predictions)]

RMSE(predicted - actual)
```

```
## [1] 9.271702
```

```
MAE(predicted - actual)
```

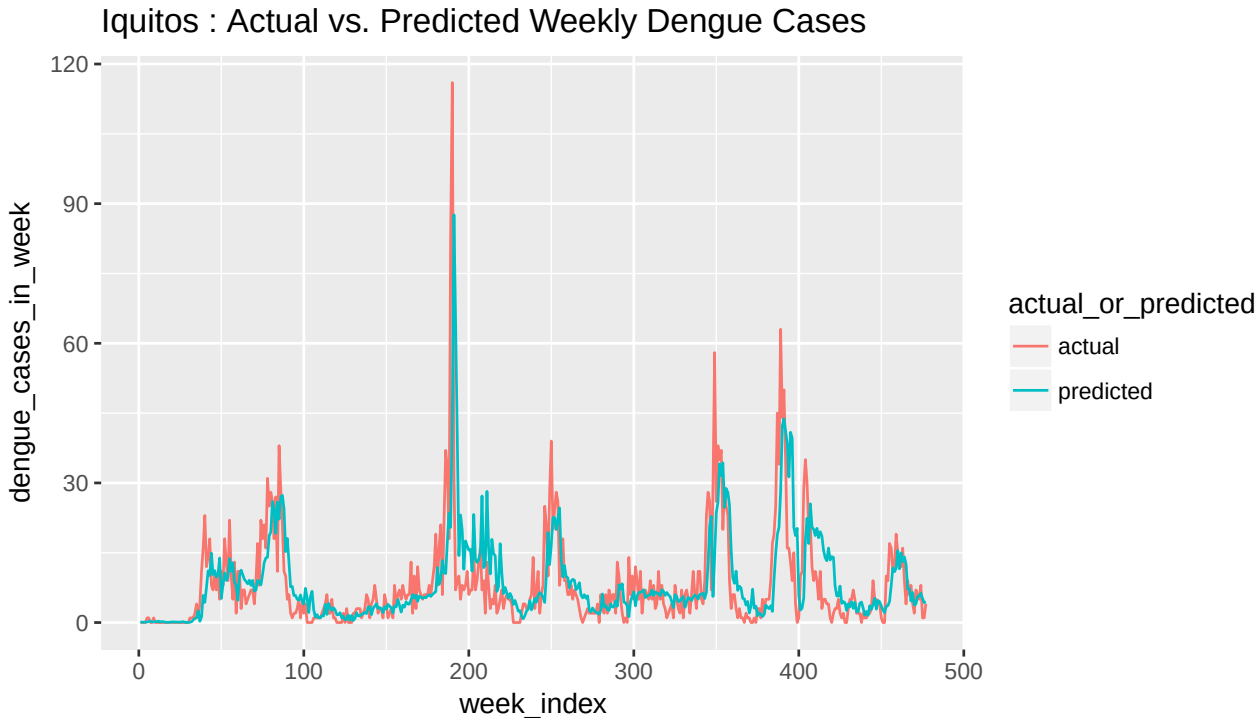
```
## [1] 5.192284
```

```
nrmse(predicted, actual)
```

```
## [1] 84.3
```

```
iq_prospective_preds <- data.frame(iq_data_for_model$year[31:length(iq_data_for_model$total_cases)], iq
colnames(iq_prospective_preds) = c("year", "week_of_year", "actual", "predicted")
iq_prospective_preds <- iq_prospective_preds %>% mutate(week_index = row_number()) %>% gather(actual_or

ggplot(iq_prospective_preds, aes(x = week_index, y = dengue_cases_in_week, col = actual_or_predicted)) +
```


Iquitos : Actual vs. Predicted Weekly Dengue Cases
dengue_cases_in_week
120
90
60
30
0
0
100
200
300
400
500
week_index
actual_or_predicted
actual
predicted