forecast.R

BrewJR

Tue Apr 28 12:30:23 2015

```
library(plyr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
##
## The following objects are masked from 'package:plyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
##
## The following object is masked from 'package:stats':
##
##     filter
##
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(weatherData)
library(randomForest)
```

```
## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.
```

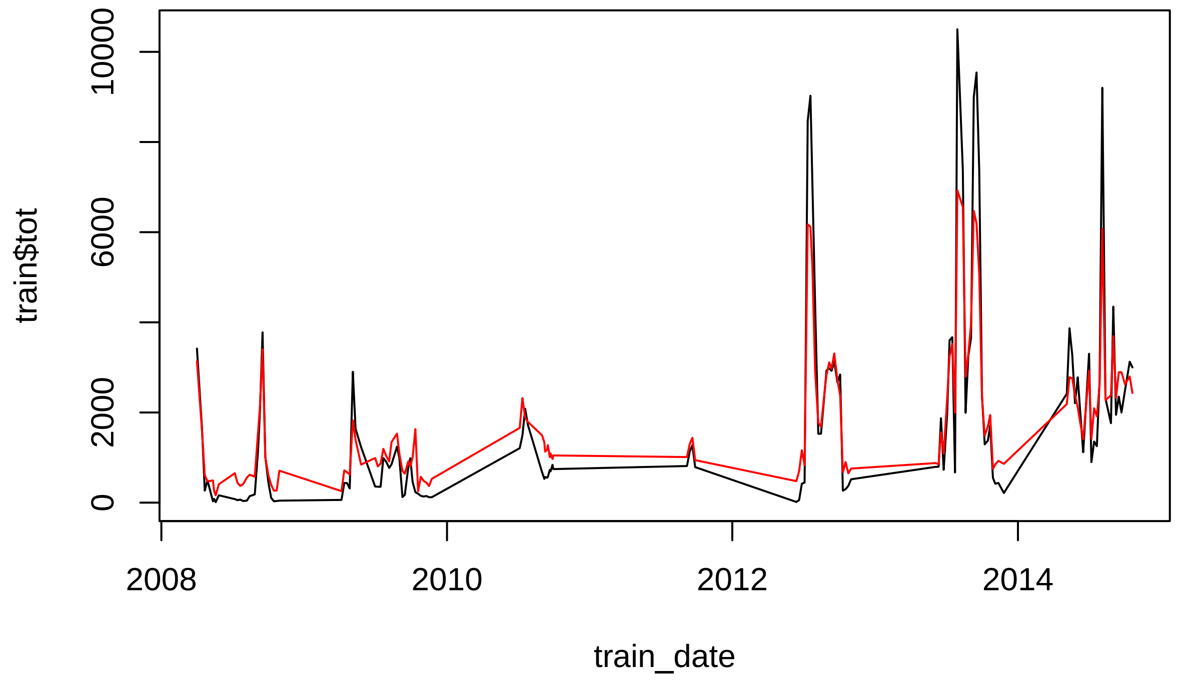
```
#####
# DEFINE WORKING DIRECTORIES
#####
if(Sys.info()['sysname'] == 'Windows'){
  trap_dir <- 'C:/Users/BrewJR/Documents/fdoh/public/mosquito_forecast/trap_data'
  weather_dir <- 'C:/Users/BrewJR/Documents/fdoh/public/mosquito_forecast/weather_data'

} else {
  trap_dir <- '/home/joebrew/Documents/fdoh/public/mosquito_forecast/trap_data'
  weather_dir <- '/home/joebrew/Documents/fdoh/public/mosquito_forecast/weather'

}
setwd(trap_dir)

####
# READ IN ALL DATA INTO LIST
####
temp = list.files(pattern="*.csv")
myfiles = lapply(temp, read.csv, skip=1)
myfileswithdate = lapply(temp, read.csv, skip=0)
```

```r
####
# MAKE A NUMBER OF TRAPS COLUMN
####
for (i in 1:length(myfiles)){
  myfiles[[i]]$nTraps <-
    ncol(myfiles[[i]]) - 2
}

####
#ADD DATE COLUMN
####
for (i in 1:length(myfiles)){
  myfiles[[i]]$date <-
    as.Date(gsub("X", "",colnames(myfileswithdate[[i]])[1]),
            format="%m.%d.%Y")
}

####
#CBIND ALL ELEMENTS OF THE MYFILES LIST INTO ONE DF
####
raw_data <- rbind.fill(myfiles)

####
#CODE ALL "OTHER" MOSQUITOES AS 100
####
raw_data$code[which(grepl("other",
                          tolower(raw_data$mosquito)))] <-  100

#####
# MAKE A TIME SERIES DATAFRAME
#####
ts <- raw_data %>%
  group_by(date) %>%
  summarise(n_traps = mean(nTraps, na.rm = TRUE),
            tot =
              sum(site1, na.rm = TRUE) +
              sum(site2, na.rm = TRUE) +
              sum(site3, na.rm = TRUE) +
              sum(site4, na.rm = TRUE) +
              sum(site5, na.rm = TRUE) +
              sum(site6, na.rm = TRUE) +
              sum(site7, na.rm = TRUE) +
              sum(site8, na.rm = TRUE) +
              sum(site9, na.rm = TRUE) +
              sum(site10, na.rm = TRUE))

# ! BUG
ts$n_traps[which(ts$n_traps > 10 )] <- 10

############

#####
# GET RAINFALL DATA
```

```r
#####

setwd(weather_dir)
if(file.exists('weather.csv')){
  weather <- read.csv('weather.csv')
} else {

  weather_list <- list()
  min_year <- as.numeric(format(min(ts$date), '%Y'))
  max_year <- as.numeric(format(Sys.Date(), '%Y'))
  year_seq <- min_year:max_year
  for (i in 1:length(year_seq) ){

    # Define end dates
    if(year_seq[i] == max_year){
      end <-  Sys.Date() - 1
    } else {
      end <-  paste0(year_seq[i], '-12-31')
    }

    # Get data
    rf <- getSummarizedWeather("GNV",
                               start_date = paste0(year_seq[i], '-01-01'),
                               end_date = end,
                               opt_custom_columns = TRUE,
                               custom_columns = c(2,4,20))

    # Put into list
    weather_list[[i]] <- rf
    print(year_seq[i])
  }

  # Combine all weather into one dataframe
  weather <- do.call("rbind", weather_list)
  rm(min_year, max_year, year_seq, i, weather_list, rf)

  # Write csv
  setwd(weather_dir)
  write.csv(weather, 'weather.csv', row.names = FALSE)
  weather <- read.csv('weather.csv')
}

# Format date
weather$Date <- as.Date(weather$Date)

#####
# GET MORE RECENT WEATHER IF NEEDED
#####
if(max(weather$Date) < (Sys.Date() - 1)){
  new_weather <- getSummarizedWeather("GNV",
                                      start_date = max(weather$Date) +1,
                                      end_date = Sys.Date() - 1,
                                      opt_custom_columns = TRUE,
                                      custom_columns = c(2,4,20))
  weather <- rbind(weather, new_weather)

  # Clean up
  # Format date
  weather$Date <- as.Date(weather$Date)


  # Write csv
  setwd(weather_dir)
  write.csv(weather, 'weather.csv', row.names = FALSE)
  weather <- read.csv('weather.csv')
}

#####
# CLEAN UP A BIT MORE
#####

# Format date
weather$Date <- as.Date(weather$Date)


# Get right structure for columns
for (j in 2:ncol(weather)){
  suppressWarnings(weather[,j] <- as.numeric(as.character(weather[,j])))
  weather[,j][which(is.na(weather[,j]))] <- 0
}

# Order
weather <- weather[order(weather$Date),]

##############################################################
##############################################################
##############################################################

#####
# CREATE AN AUGMENTED WEATHER DATAFRAME
# WITH TS FEATURES
#####
weather_augmented <- weather

# Create dataframe for the next two weeks
next_two_weeks <- as.data.frame(matrix(NA,
                                       ncol = ncol(weather),
                                       nrow = 14))
colnames(next_two_weeks) <- colnames(weather)
next_two_weeks$Date <- as.Date(seq(Sys.Date(), (Sys.Date()+13), 1))

# Combine with weather_augmented
weather_augmented <- rbind(weather_augmented, next_two_weeks)


for (column in colnames(weather)[2:4]){
```

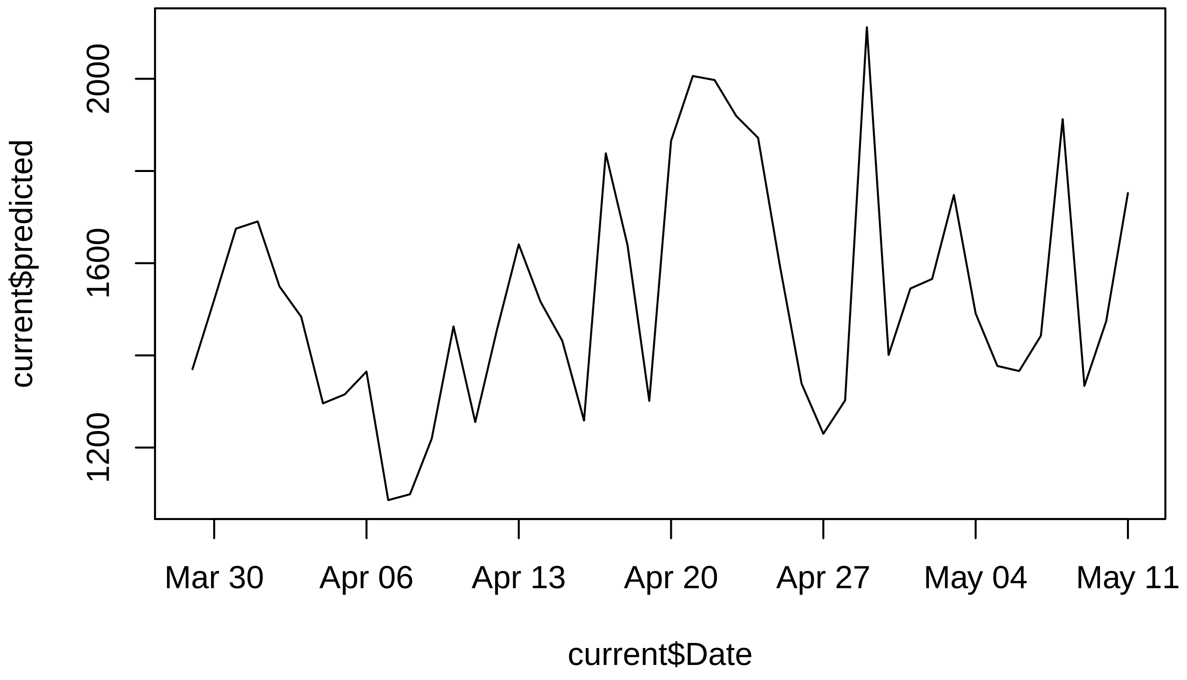
```r
  # looking only at the period 14 to 30 days before
  for (day in 14:30){
    weather_augmented[,paste0(column, '_minus_', day)] <- NA
  }
}

for (i in 1:nrow(weather_augmented)){
  for (column in colnames(weather)[2:4]){
    for (day in 14:30){
      if(i > 30){
      date <- weather_augmented$Date[i]
      val <- weather_augmented[(i-day),column]
      weather_augmented[i,paste0(column, '_minus_', day)] <- val
      }
    }
  }
}

# Remove the non-historic observations
weather_augmented <- weather_augmented[,!colnames(weather_augmented) %in% c('Max_TemperatureF',
                                                                              'Min_TemperatureF',
                                                                              'PrecipitationIn')]


#####
# CREATE DATAFRAME WITH GENERATED FEATURES FOR MODEL TRAINING
#####
train <- left_join(ts, weather_augmented, by = c('date' = 'Date'))

# Remove date so that it's not used in modeling
train_date <- train$date
train$date <- NULL

#####
# TRAIN MODEL
#####
fit <- randomForest(tot ~ . ,
                    data = train)

# Also get a predictions matrix
obs_mat <- predict(fit,
                   train,
                   predict.all = TRUE)
# And error matrix
temp <- as.matrix(obs_mat$individual)
error_mat <- apply(temp, 2, function(x){
  x - train$tot
})
rm(temp)

# Predict on train
train$predicted <- predict(fit, train)
```

```
plot(train_date, train$tot, type = 'l')
lines(train_date, train$predicted, col = 'red')
```



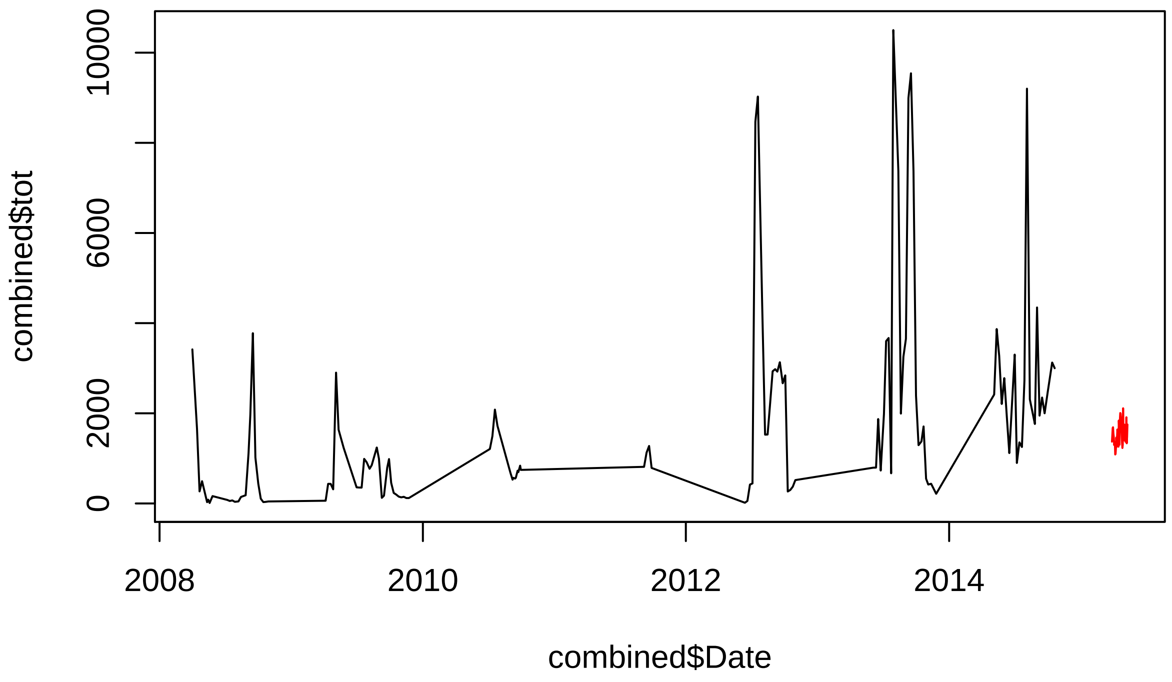
```
#####
# PREDICT ON CURRENT
#####

current <- weather_augmented[which(weather_augmented$Date >= (Sys.Date() - 30)),]
current$n_traps <- 10

current$predicted <- predict(fit, current)

# Also get a matrix of each tree's results
pred_mat <- predict(fit, current, predict.all = TRUE)

# Add error terms from the observed data:
temp <- as.matrix(pred_mat$individual)
for (j in 1:ncol(temp)) {
  errors <- sample(error_mat[, j], nrow(temp),
                   replace = TRUE)
  # temp[,j] <- temp[,j] + errors # ! too big - currently not adding errors
  temp[,j] <- temp[,j] + errors

}
pred_mat_with_error <- temp
```

```r
rm(temp)

# Now that we've introduced fundamental uncertainy,
# we can draw up individual-level predicition intervals,

# Get point estimate
current$predicted <- pred_mat$aggregate

# Get confidence bounds
current$lwr <- apply(pred_mat_with_error, 1,
                     function(x){
                       quantile(x, probs = 0.025)
                     })
current$upr <- apply(pred_mat_with_error, 1,
                     function(x) {
                       quantile(x, probs = 0.975)
                     })

plot(current$Date,
     current$predicted,
     type = 'l')
```



```r
#####
# COMBINE TRAIN AND CURRENT
#####
```

```
train_small <- train[,'tot']
train_small$Date <- train_date
train_small$observed <- TRUE

current_small <- current[,c('predicted', 'Date')]
current_small$observed <- TRUE
names(current_small)[1] <- 'tot'

combined <- rbind(train_small, current_small)

plot(combined$Date, combined$tot, type = 'n')

lines(train_small$Date, train_small$tot)
lines(current_small$Date, current_small$tot, col = 'red')
```

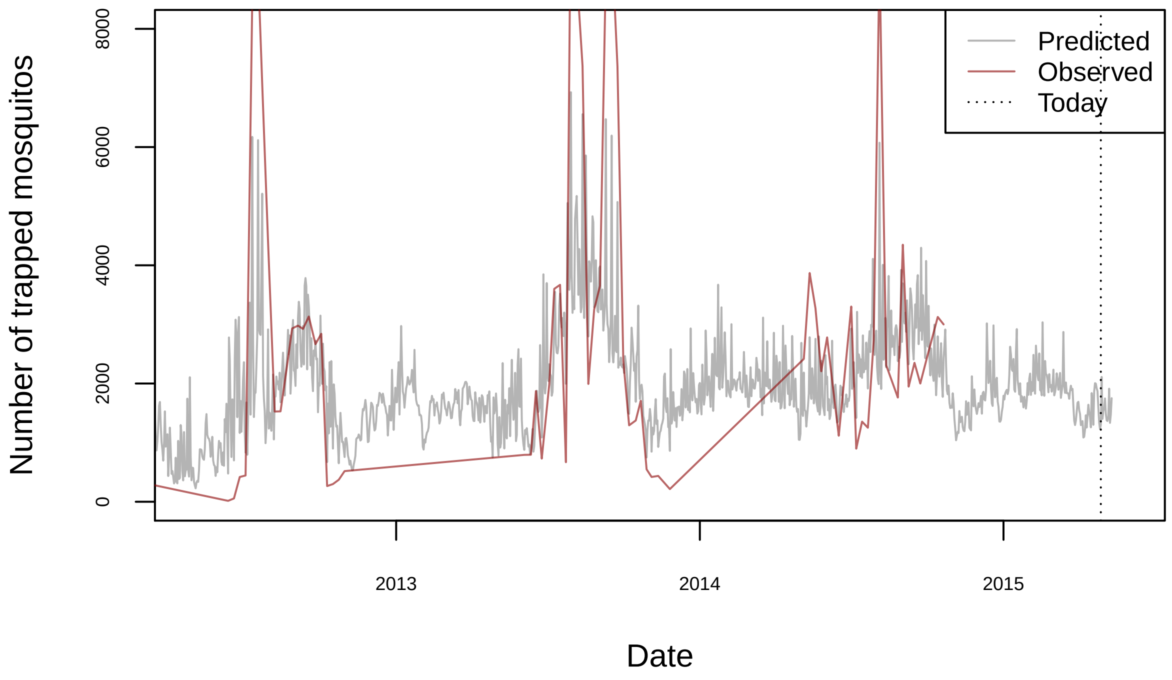


```
#####
# PREDICT ON WEATHER AUGMENTED
#####
weather_augmented <- left_join(weather_augmented, train)
```

```
## Joining by: c("Max_TemperatureF_minus_14", "Max_TemperatureF_minus_15", "Max_TemperatureF_minus_16",
```

```
# Get number of traps
weather_augmented$n_traps[1] <- 6
```

```r
for (i in 1:nrow(weather_augmented)){
  while(is.na(weather_augmented$n_traps[i])){
    weather_augmented$n_traps[i] <- weather_augmented$n_traps[i-1]
  }
}

weather_augmented$predicted <- predict(fit, weather_augmented)

plot(weather_augmented$Date, weather_augmented$predicted, type = 'l',
     col = adjustcolor('black', alpha.f = 0.3),
     ylim = c(0, 8000),
     xlim = as.Date(c('2012-05-01', '2015-05-30')),
     xlab = 'Date',
     ylab = 'Number of trapped mosquitos',
     cex.axis = 0.6)
lines(train_small$Date, train_small$tot,
      col = adjustcolor('darkred', alpha.f = 0.6))
abline(v = Sys.Date(), lty = 3)

legend('topright',
       lty = c(1, 1, 3),
       col = c(adjustcolor('black', alpha.f = 0.3),
               adjustcolor('darkred', alpha.f = 0.6),
               'black'),
       legend = c('Predicted',
                  'Observed',
                  'Today'),
       cex = 0.8)
```


Number of trapped mosquitos
0
2000
4000
6000
8000
2013
2014
2015
Date
Predicted
Observed
Today