Kaggle West Nile Virus Competition

Team UWKT3

9 June 2015

Contents

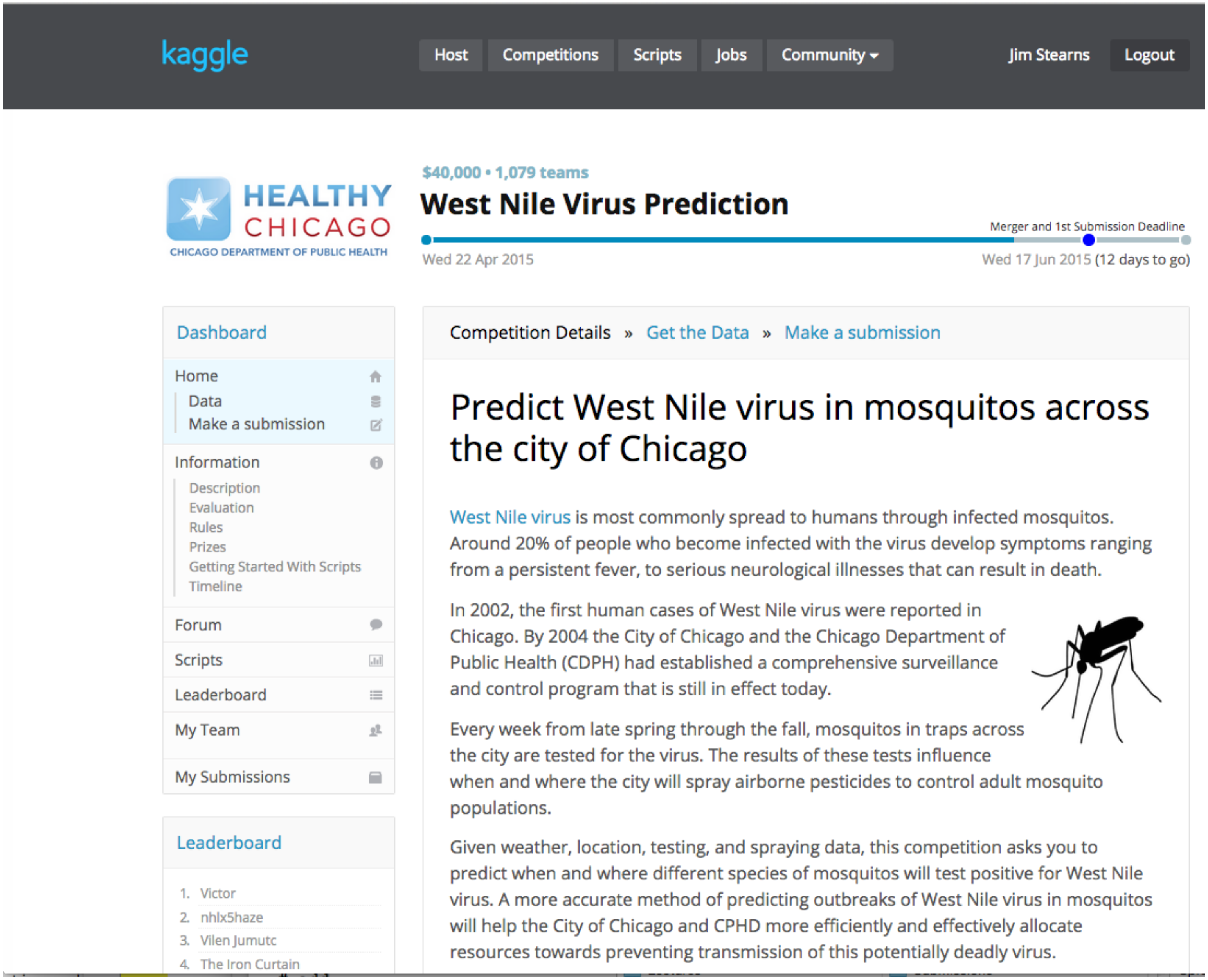
0. Introduction

The culminating project of the Spring 2015 UW PCE Data Science course “Data at Scale” was to participate in a Kaggle data science competition. Instructor Dr. Barga chose the West Nile Virus Prediction competition.

The class was broken up into three teams of roughly 8 students. This is the report of the third team with the Kaggle name of UWKT3.

0.1 Background

The West Nile Virus Prediction (WNVP) competition’s goal was to “Predict West Nile virus in mosquitos across the city of Chicago.”:



Predict West Nile virus in mosquitos across the city of Chicago

West Nile virus is most commonly spread to humans through infected mosquitos. Around 20% of people who become infected with the virus develop symptoms ranging from a persistent fever, to serious neurological illnesses that can result in death.

In 2002, the first human cases of West Nile virus were reported in Chicago. By 2004 the City of Chicago and the Chicago Department of Public Health (CDPH) had established a comprehensive surveillance and control program that is still in effect today.

Every week from late spring through the fall, mosquitos in traps across the city are tested for the virus. The results of these tests influence when and where the city will spray airborne pesticides to control adult mosquito populations.

Given weather, location, testing, and spraying data, this competition asks you to predict when and where different species of mosquitos will test positive for West Nile virus. A more accurate method of predicting outbreaks of West Nile virus in mosquitos will help the City of Chicago and CPHD more efficiently and effectively allocate resources towards preventing transmission of this potentially deadly virus.

The WNVP contest started on April 22nd and will end 17 June 2015. UWKT3’s first submission was on May 12th. Its last so far was on June 4th.

0.2 Team Goals

1. Experiment with modelling alternative on a real-world dataset.

2. Learn how to participate in a Kaggle competition.
3. Non-Goal: win the competition, or even score highly.

0.3 Team Members

- Bethene Britt
- Andrew Ewing
- Gregory Hogue
- Patrick Leahy
- Linghua Qiu
- Chris Ross
- Robert Russell
- Jim Stearns

1. Data Preparation

A goal of the team project was to create a "golden" train and test dataset that could form the basis of many modeling experiments.

1.1 Obtain Original Datasets from Kaggle Website

Download the training, test, spray, and weather data from the Kaggle web site page for West Nile Virus Prediction.

Please see Appendix for R (and Python) code that:

- General Setup: Clears environment, sets working directory, loads libraries, define utility functions
- Downloads the data from the Kaggle site and unzips them.

```
stopifnot(allKaggleFilesArePresent())
print("All unzipped Kaggle datasets found in PWD. Proceeding.")
```

```
## [1] "All unzipped Kaggle datasets found in PWD. Proceeding."
```

1.2 Read Kaggle Train/Test Files

- Read in test and train csv-format files into data frames.

```
test_df <- read.csv(paste0(dataSubDir, "/", wnvpTestFilename))
train_df <- read.csv(paste0(dataSubDir, "/", wnvpTrainFilename))
# Quick sanity check: got right number of records?
stopifnot(nrow(train_df) == wnvpTrainFileNRecs)
stopifnot(nrow(test_df) == wnvpTestFileNRecs)
```

1.3 Feature Selection/Creation

- Make the train and test datasets have the same attributes:
  - Train: Convert the WnvPresent column from numeric to factor with levels "Yes" and "No".
  - Train: Add an Id attribute, set to zero.
  - Train: Remove NumMosquitos attribute. Potentially useful, but not available in Test dset.
  - Test: Add a WnvPresent factor column, all with "No" level.
  - Both: Remove the address attributes of little use compared to Lat/Long:
    - Address, Block, Street, AddressNumberAndStreet, AddressAccuracy.
  - Both: Add bit vectors of each of the levels of the Species factor (a new column for each of the factor levels, with a zero or 1 value). Leave the Species as well.
  - Both: Convert date into date format, add "Year", "Month", and "Week" factor attributes.

```r
# WnvPresent. Train: convert to factor. Test: add as factor, default value of "No".
train_df$WnvPresent <- factor(train_df$WnvPresent, labels=c("No", "Yes"))
WnvPresent <- factor("No", levels=c("No","Yes"))
test_df <- cbind(test_df, WnvPresent)

# Train: Add Id attribute to match that in Test. Set to 0. Id in Test is 1-relative.
train_df["Id"] <- 0
train_df <- moveColsToFirst(train_df, "Id")

# Train: Remove NumMosquitos attribute. Potentially useful, but not available in Test dset.
train_df$NumMosquitos <- NULL

# Both: Remove the block attributes of little use:
attrsToRemove <- c("Address", "Block", "Street", "AddressNumberAndStreet", "AddressAccuracy")
train_df <- train_df[,!names(train_df) %in% attrsToRemove]
test_df <- test_df[,!names(test_df) %in% attrsToRemove]

# For creation of factor attributes, temporarily combine train and test into one dataset
# so that factor levels are the same when both are written out as separate files.
# Keeps Weka happy.
# Add a temporary column distinquishing train from test dataset entries.
train_df$DsetType <- "Train"
test_df$DsetType <- "Test"

combined_df = rbind(train_df, test_df)

# Both (in Combined): Add bit vectors for Species, one column for each factor level
# TODO: "UNSPECIFIED CULEX" needs attention.
combined_df <- with(combined_df, cbind(model.matrix( ~ 0 + Species, combined_df), combined_df))

# Both (in Combined): Convert date into date format,
# add "Year", "Month", and "Week" factor attributes.
# as.Date() tries %Y-%m-%d by default, but what the heck, explicitly state the format.
combined_df$Date <- as.Date(combined_df$Date, format="%Y-%m-%d")

combined_df$Year <- as.factor(format(combined_df$Date, "%Y"))
combined_df$Month <- as.factor(format(combined_df$Date, "%m"))
combined_df$Week <- as.factor(format(combined_df$Date, "%U"))
```

```
# Move temporary dsetType and date-related attributes to left, leaving WnvPresent last
combined_df <- moveColsToFirst(combined_df, c("DsetType", "Id", "Date", "Year", "Month", "Week")

# Do not remove the Species attribute - not all models will use the bit vectors.
#train$Species <- NULL
#test$Species <- NULL
```

1.4 Feature Creation (ctd): Merge Weather Data with Trap Observations in Train/Test Datasets

Calculate Distance of Trap from the Two Weather Stations

- Both (in Combined): Calculate the distance (using lat/long) of the trap from the two weather stations, adding attributes with the value in kilometers. Patience: This takes a while (~5 minutes).
- Both (in Combined): Add a nearest weather station attribute.

Using function *distCosine* in R Geosphere Package to calculate distance on a sphere.

```
# Station 1: O'Hare
station1LongLat <- c(-87.933, 41.995)

# Station 2: Midway
station2LongLat <- c(-87.752, 41.786)

# Patience. This takes a while (~5 minutes)
for (i in 1:nrow(combined_df)) {
    combined_df$Station1DistKm[i] <- distCosine(
        c(combined_df$Longitude[i], combined_df$Latitude[i]), station1LongLat) / 1000
    combined_df$Station2DistKm[i] <- distCosine(
        c(combined_df$Longitude[i], combined_df$Latitude[i]), station2LongLat) / 1000
}
combined_df$NearestStation <- ifelse(
    combined_df$Station1DistKm <= combined_df$Station2DistKm, 1, 2)
```

- Both (in Combined): Merge in temperature and wind data from nearest station on that date.

```
weather_df <- read.csv(paste0(dataSubDir, "/", wnvpWeatherFilename), stringsAsFactors=FALSE)
colsToKeep <- c("Station", "Date", "Tmax", "Tmin", "Tavg", "AvgSpeed")
weatherData <- weather_df[,names(weather_df) %in% colsToKeep]
weatherData$Date <- as.Date(weatherData$Date, format="%Y-%m-%d")
# Tmax and Tmin come in as type int. Tavg, however, comes in as chr.
weatherData$Tavg <- as.integer(weatherData$Tavg)
```

```
## Warning: NAs introduced by coercion
```

```
# So does AvgSpeed.
weatherData$AvgSpeed <- as.numeric(weatherData$AvgSpeed)
```

```
## Warning: NAs introduced by coercion
```

```
combinedww <- merge(combined_df, weatherData,
                             by.x=c("Date", "NearestStation"), by.y=c("Date", "Station"),
                             all.x=TRUE)
#str(combinedww)

# Make "Id" the first column and "WnvPresent" the last.
combinedww <- moveColsToFirst(combinedww, "Id")
combinedww <- moveColsToLast(combinedww, "WnvPresent")

stopifnot(nrow(combinedww) == (wnvpTrainFileNRecs + wnvpTestFileNRecs))
```

Weather data does have some NA fields (warning above: “Warning: NAs introduced by coercion”), but not the subset merged into train and test dset. Throw an exception if that ever proves not to be the case.

```
stopifnot(sum(is.na(combinedww$Tmin)) == 0)
stopifnot(sum(is.na(combinedww$Tmax)) == 0)
stopifnot(sum(is.na(combinedww$Tavg)) == 0)
stopifnot(sum(is.na(combinedww$AvgSpeed)) == 0)
```

1.5 Write “master” train and test files, complete (all attributes), to CSV and ARFF

- Write the train and test datasets - including weather data - as CSV.

```
stopifnot(sum(combinedww$DsetType == "Train") == wnvpTrainFileNRecs)
stopifnot(sum(combinedww$DsetType == "Test") == wnvpTestFileNRecs)

trainWithWeather <- combinedww[combinedww$DsetType == "Train",]
stopifnot(nrow(trainWithWeather) == wnvpTrainFileNRecs)
trainWithWeather$DsetType <- NULL

write.csv(trainWithWeather,
          paste0(workingSubDir, "/", "train", "Master", ".csv"),
          eol = '\n')
str(trainWithWeather)
```

```
## 'data.frame':    10506 obs. of  25 variables:
##  $ Id                         : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Date                       : Date, format: "2007-05-29" "2007-05-29" ...
##  $ NearestStation             : num  1 1 1 1 1 1 1 2 2 2 ...
##  $ Year                       : Factor w/ 8 levels "2007","2008",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Month                      : Factor w/ 6 levels "05","06","07",..: 1 1 1 1 1 1 1 1 1 1 ..
##  $ Week                       : Factor w/ 20 levels "21","22","23",..: 1 1 1 1 1 1 1 1 1 1 .
##  $ SpeciesCULEX ERRATICUS       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ SpeciesCULEX PIPIENS         : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ SpeciesCULEX PIPIENS/RESTUANS: num  1 0 0 1 0 1 1 0 1 0 ...
##  $ SpeciesCULEX RESTUANS        : num  0 1 1 0 1 0 0 1 0 1 ...
##  $ SpeciesCULEX SALINARIUS      : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ SpeciesCULEX TARSALIS        : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ SpeciesCULEX TERRITANS       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ SpeciesUNSPECIFIED CULEX     : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Species                      : Factor w/ 8 levels "CULEX ERRATICUS",..: 3 4 4 3 4 3 3 4 3 4
##  $ Trap                         : Factor w/ 149 levels "T001","T002",..: 2 2 7 14 14 95 90 34
##  $ Latitude                     : num  42 42 42 42 42 ...
##  $ Longitude                    : num  -87.8 -87.8 -87.8 -87.8 -87.8 ...
##  $ Station1DistKm               : num  11.81 11.81 13.55 9.25 9.25 ...
##  $ Station2DistKm               : num  19.2 19.2 23.3 21.8 21.8 ...
##  $ Tmax                         : int  88 88 88 88 88 88 88 88 88 88 ...
##  $ Tmin                         : int  60 60 60 60 60 60 60 65 65 65 ...
##  $ Tavg                         : int  74 74 74 74 74 74 74 77 77 77 ...
##  $ AvgSpeed                     : num  6.5 6.5 6.5 6.5 6.5 6.5 6.5 7.4 7.4 7.4 ...
##  $ WnvPresent                   : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...
```

```
testWithWeather <- combinedww[combinedww$DsetType == "Test",]
stopifnot(nrow(testWithWeather) == wnvpTestFileNRecs)
# Make sure test dataset is still ordered by Id
testWithWeather <- testWithWeather[order(testWithWeather$Id),]
trainWithWeather$DsetType <- NULL

write.csv(testWithWeather,
          paste0(workingSubDir, "/", "test", "Master", ".csv"),
          eol = '\n')
#str(testWithWeather)
```

- Write ARFF versions as well. Advantage over CSV: levels of factors are maintained, even if no observations have that level. CSV builds levels from usage.

```
write.arff(trainWithWeather,
          paste0(workingSubDir, "/", "train", "Master", ".arff"),
          eol = '\n')

write.arff(testWithWeather,
          paste0(workingSubDir, "/", "test", "Master", ".arff"),
          eol = '\n')
```

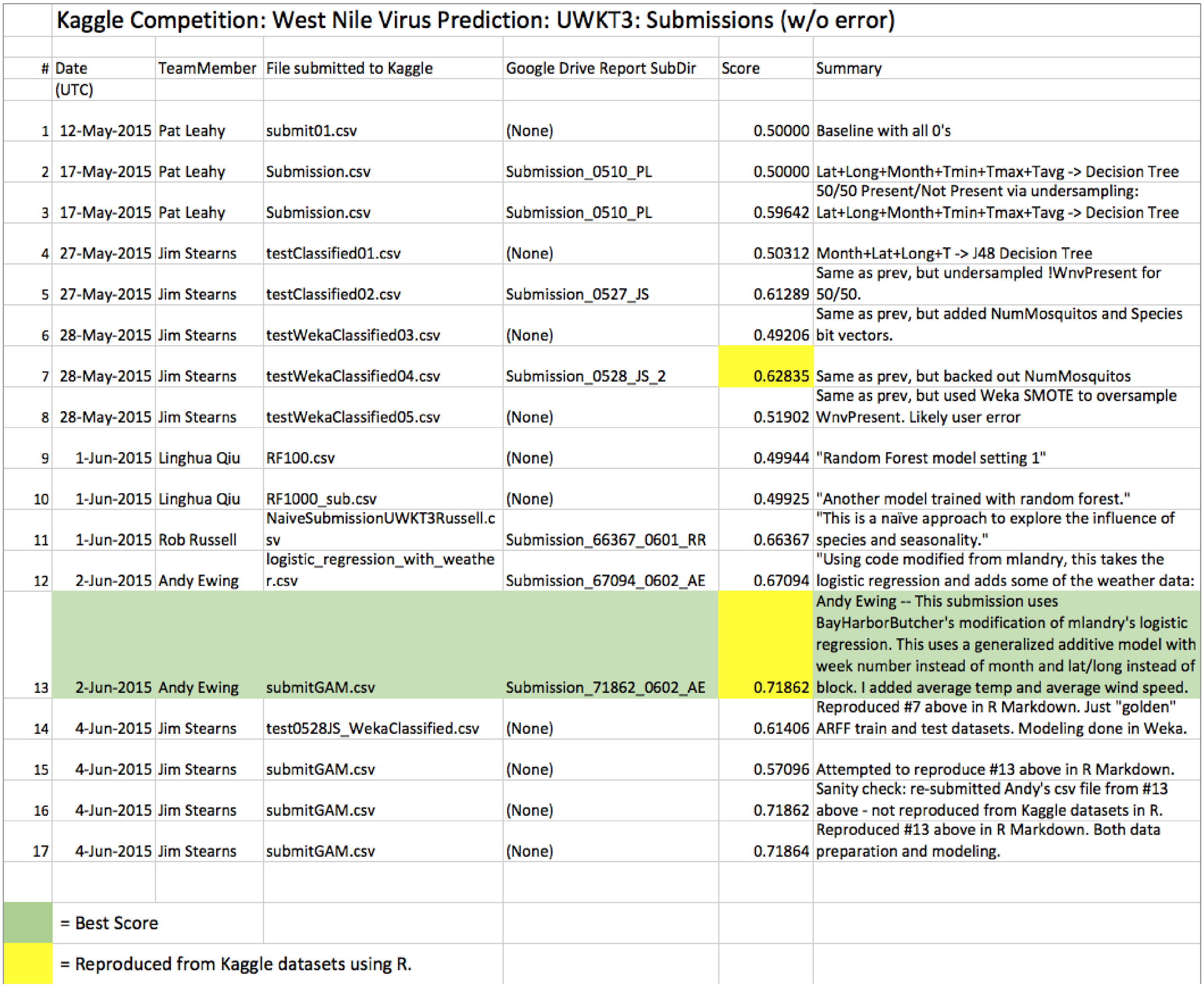
2. Data Exploration and Analysis

TBD

3. Features for Modeling

TBD

4. Kaggle Submissions



Kaggle Competition: West Nile Virus Prediction: UWKT3: Submissions (w/o error)

#	Date (UTC)	TeamMember	File submitted to Kaggle	Google Drive Report SubDir	Score	Summary
1	12-May-2015	Pat Leahy	submit01.csv	(None)	0.50000	Baseline with all 0's
2	17-May-2015	Pat Leahy	Submission.csv	Submission_0510_PL	0.50000	Lat+Long+Month+Tmin+Tmax+Tavg -> Decision Tree
3	17-May-2015	Pat Leahy	Submission.csv	Submission_0510_PL	0.59642	50/50 Present/Not Present via undersampling: Lat+Long+Month+Tmin+Tmax+Tavg -> Decision Tree
4	27-May-2015	Jim Stearns	testClassified01.csv	(None)	0.50312	Month+Lat+Long+T -> J48 Decision Tree
5	27-May-2015	Jim Stearns	testClassified02.csv	Submission_0527_JS	0.61289	Same as prev, but undersampled !WnvPresent for 50/50.
6	28-May-2015	Jim Stearns	testWekaClassified03.csv	(None)	0.49206	Same as prev, but added NumMosquitos and Species bit vectors.
7	28-May-2015	Jim Stearns	testWekaClassified04.csv	Submission_0528_JS_2	0.62835	Same as prev, but backed out NumMosquitos
8	28-May-2015	Jim Stearns	testWekaClassified05.csv	(None)	0.51902	Same as prev, but used Weka SMOTE to oversample WnvPresent. Likely user error
9	1-Jun-2015	Linghua Qiu	RF100.csv	(None)	0.49944	"Random Forest model setting 1"
10	1-Jun-2015	Linghua Qiu	RF1000_sub.csv	(None)	0.49925	"Another model trained with random forest."
11	1-Jun-2015	Rob Russell	NaiveSubmissionUWKT3Russell.csv	Submission_66367_0601_RR	0.66367	"This is a naïve approach to explore the influence of species and seasonality."
12	2-Jun-2015	Andy Ewing	logistic_regression_with_weather.csv	Submission_67094_0602_AE	0.67094	"Using code modified from mlandry, this takes the logistic regression and adds some of the weather data:
13	2-Jun-2015	Andy Ewing	submitGAM.csv	Submission_71862_0602_AE	0.71862	Andy Ewing -- This submission uses BayHarborButcher's modification of mlandry's logistic regression. This uses a generalized additive model with week number instead of month and lat/long instead of block. I added average temp and average wind speed.
14	4-Jun-2015	Jim Stearns	test0528JS_WekaClassified.csv	(None)	0.61406	Reproduced #7 above in R Markdown. Just "golden" ARFF train and test datasets. Modeling done in Weka.
15	4-Jun-2015	Jim Stearns	submitGAM.csv	(None)	0.57096	Attempted to reproduce #13 above in R Markdown.
16	4-Jun-2015	Jim Stearns	submitGAM.csv	(None)	0.71862	Sanity check: re-submitted Andy's csv file from #13 above - not reproduced from Kaggle datasets in R.
17	4-Jun-2015	Jim Stearns	submitGAM.csv	(None)	0.71864	Reproduced #13 above in R Markdown. Both data preparation and modeling.

🟩 = Best Score

🟨 = Reproduced from Kaggle datasets using R.

4.1 May 17 Pat Leahy Submittal (Decision Tree, Score 0.59642)

Pat's summary of data preparation, feature selection, and model:

Data Preparation

We joined the weather data provided by Kaggle to the training and test records. This resulted in two new tables which contained the test and training data along with a set of weather attributes from the nearest weather station for the date in question.

We carried out our data preparation in Excel. We copied the files train.csv, test.csv and weather.csv into tabs in an Excel workbook. There were two weather stations in the weather data. We calculated the distance from each observation point to each of the two weather stations. We used an Excel macro copied from http://www.codecodex.com/wiki/Calculate_distance_between_two_points_on_a_globe#Excel to calculate the distances given latitude and longitude. We then determined which weather station was closer to each point. We used the weather station ID and date as a key to join test and training records to the weather records. We used Excels VLOOKUP function to implement a join.

Feature Selection

Once we had the training and test data joined to the weather data we selected some features to generate a model. A team member studied mosquitos and reported the following

"Culex mosquitoes lay their eggs usually at night on the surface of fresh or stagnant water; usually lay their eggs at night; a mosquito may lay a raft of eggs every third night during its life span.

Culex usually live only a few weeks during the warm summer months; those females which emerge in late summer search for sheltered areas where they hibernate (diapause) until spring; warm weather brings them out in search of water on which to lay their eggs. "

Given this knowledge we selected the Month as a feature.

Chicago has one large body of water with a coastline which runs in approximately a straight line. We therefore concluded that Latitude and Longitude would also be useful features.

We also selected three temperature measures from the weather data. They were Minimum Temperature, Maximum Temperature and Average Temperature. We selected these temperature features because they didn't contain any missing values.

The full set of features we selected were Latitude, Longitude, Month, Minimum Temperature, Maximum Temperature and Average Temperature.

Model

We decided to over sample the test data to include the same number of positive observations for West Nile Virus as negative observations. We did this by selecting all the positive observations together with an equal number of negative observations randomly selected. We carried out the random selection in Excel by adding a new column of randomly generated values using the RAND function and then sorting using that column. We created two new CSV files, a training and test file, containing only our selected features. The training set only contained the equally represented subset of positive and negative observations.

We opened the training set in Weka and generated a Decision Tree using the J48 classifier. We tuned some of the parameters until we settled on the following settings, "-C 0.5 -M 2". We had to reformat the class column in the training file to be Yes/No instead for 1/0 for Weka to recognize it as a class. We then used the test.csv we created with only our specific features. We had some difficulty using the test file until we added a class column. This we just set to No for all records.

Weka failed to run the model if we tried to output the results of the test to a file regardless of the file type. To work around this we turned off output to a file. Instead we right clicked on the results in the result list and selected "Visualize classification errors". We could then save the predictions in the window which opened as an ARFF file. We converted this to a CSV, changed some of the columns and this gave me a submission file to upload to Kaggle. We uploaded this submission and achieved an accuracy of 0.59642. This is better than the accuracy of 0.5 we achieved when prediction no West Nile Virus for every test record.

Reproduce ARFF datasets for Use as Model Input in Weka

Read in the train and test golden "Master" datasets. Use the ARFF versions so that the factor types are preserved with the same levels, even if some levels are not present in any record in the file.

```
trainRecs <- read.arff(paste0(workingSubDir, "/", "train", "Master", ".arff"))
testRecs <- read.arff(paste0(workingSubDir, "/", "test", "Master", ".arff"))
stopifnot(nrow(trainRecs) == wnvpTrainFileNRecs)
stopifnot(nrow(testRecs) == wnvpTestFileNRecs)
```

Perform any subsetting here so that train and test formats look the same to Weka.

```
colsToKeep=c("Latitude", "Longitude", "Month", "Tmin", "Tmax", "Tavg", "WnvPresent")
trainRecs <- trainRecs [,names(trainRecs) %in% colsToKeep]
testRecs <- testRecs [,names(testRecs) %in% colsToKeep]
```

Undersample: use all the WnvPresent==True samples. Randomly select an equal number of False samples. Use that for the test data set.

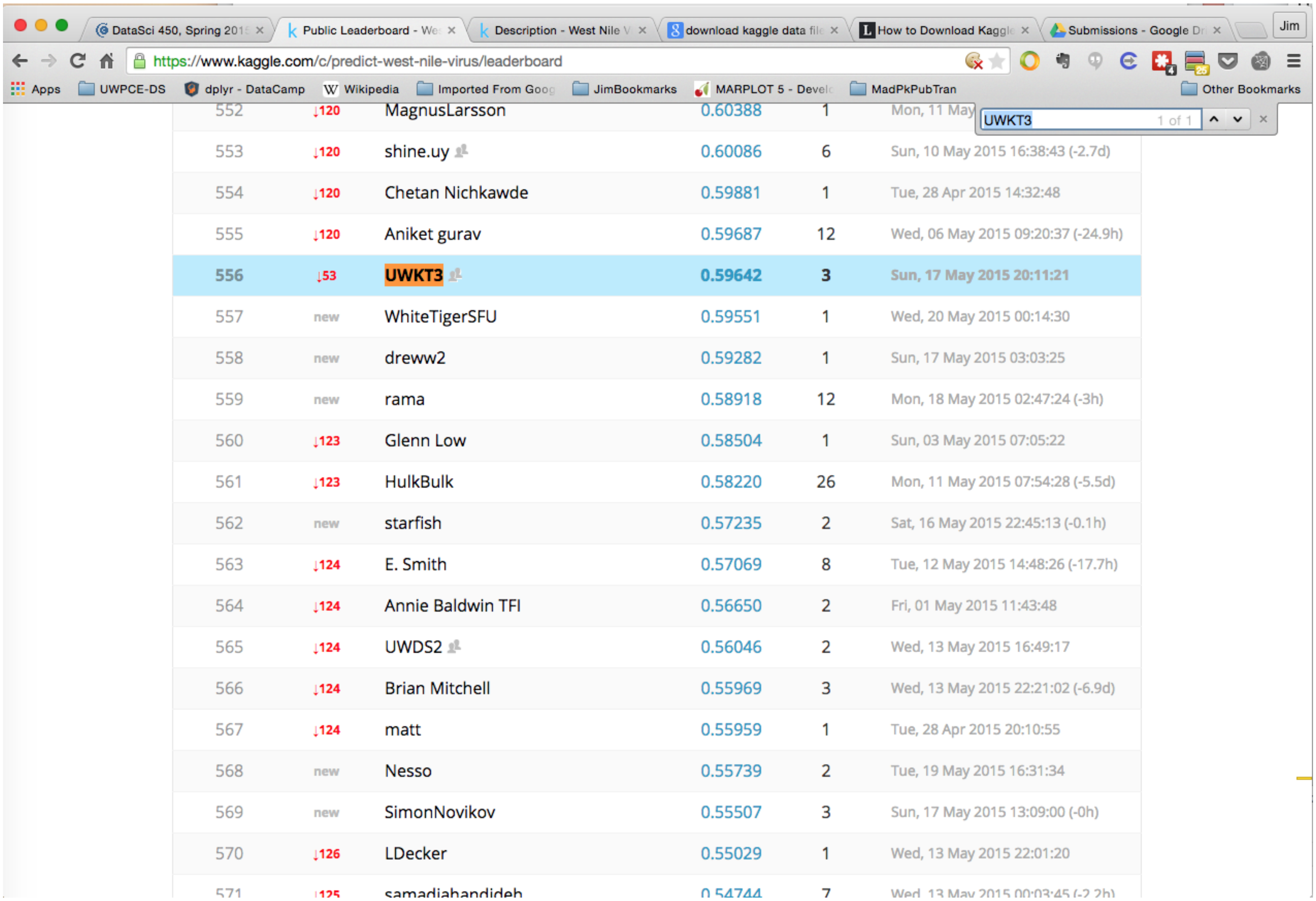
```
curModelIdx <- "0517PL"
undersample_df <- trainRecs[trainRecs$WnvPresent=="Yes",]
nFalseObservationsToUse <- nrow(undersample_df)
wnvNotPresent <- trainRecs[trainRecs$WnvPresent=="No",]
undersample_df <- rbind(undersample_df,
                       wnvNotPresent[sample(nrow(wnvNotPresent), nFalseObservationsToUse),])

write.arff(undersample_df,
           paste0(workingSubDir, "/", "train", curModelIdx, ".arff"),
           eol = '\n', relation="WNVPTrainDataset")
str(undersample_df)
```

```
## 'data.frame':    1102 obs. of  7 variables:
##  $ Month     : Factor w/ 6 levels "05","06","07",..: 3 3 3 3 3 3 4 4 4 4 ...
##  $ Latitude  : num  41.7 41.7 41.7 41.7 41.7 ...
##  $ Longitude : num  -87.5 -87.6 -87.6 -87.6 -87.6 ...
##  $ Tmax      : num  85 83 83 83 83 83 92 92 92 92 ...
##  $ Tmin      : num  69 70 70 70 70 70 69 69 69 69 ...
##  $ Tavg      : num  77 77 77 77 77 77 81 81 81 81 ...
##  $ WnvPresent: Factor w/ 2 levels "No","Yes": 2 2 2 2 2 2 2 2 2 2 ...
```

```
write.arff(testRecs,
           paste0(workingSubDir, "/", "test", curModelIdx, ".arff"),
           eol = '\n', relation="WNVPTestDataset")
str(testRecs)
```

```
## 'data.frame':    116293 obs. of  7 variables:
##  $ Month     : Factor w/ 5 levels "06","07","08",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Latitude  : num  42 42 42 42 42 ...
##  $ Longitude : num  -87.8 -87.8 -87.8 -87.8 -87.8 ...
##  $ Tmax      : num  86 86 86 86 86 86 86 86 86 86 ...
##  $ Tmin      : num  61 61 61 61 61 61 61 61 61 61 ...
##  $ Tavg      : num  74 74 74 74 74 74 74 74 74 74 ...
##  $ WnvPresent: Factor w/ 1 level "No": 1 1 1 1 1 1 1 1 1 1 ...
```

Screenshot of Leaderboard



552	↓120	MagnusLarsson	0.60388	1	Mon, 11 May
553	↓120	shine.uy	0.60086	6	Sun, 10 May 2015 16:38:43 (-2.7d)
554	↓120	Chetan Nichkawde	0.59881	1	Tue, 28 Apr 2015 14:32:48
555	↓120	Aniket gurav	0.59687	12	Wed, 06 May 2015 09:20:37 (-24.9h)
556	↓53	UWKT3	0.59642	3	Sun, 17 May 2015 20:11:21
557	new	WhiteTigerSFU	0.59551	1	Wed, 20 May 2015 00:14:30
558	new	dreww2	0.59282	1	Sun, 17 May 2015 03:03:25
559	new	rama	0.58918	12	Mon, 18 May 2015 02:47:24 (-3h)
560	↓123	Glenn Low	0.58504	1	Sun, 03 May 2015 07:05:22
561	↓123	HulkBulk	0.58220	26	Mon, 11 May 2015 07:54:28 (-5.5d)
562	new	starfish	0.57235	2	Sat, 16 May 2015 22:45:13 (-0.1h)
563	↓124	E. Smith	0.57069	8	Tue, 12 May 2015 14:48:26 (-17.7h)
564	↓124	Annie Baldwin TFI	0.56650	2	Fri, 01 May 2015 11:43:48
565	↓124	UWDS2	0.56046	2	Wed, 13 May 2015 16:49:17
566	↓124	Brian Mitchell	0.55969	3	Wed, 13 May 2015 22:21:02 (-6.9d)
567	↓124	matt	0.55959	1	Tue, 28 Apr 2015 20:10:55
568	new	Nesso	0.55739	2	Tue, 19 May 2015 16:31:34
569	new	SimonNovikov	0.55507	3	Sun, 17 May 2015 13:09:00 (-0h)
570	↓126	LDecker	0.55029	1	Wed, 13 May 2015 22:01:20
571	↓125	samadiahandideh	0.54744	7	Wed, 13 May 2015 00:03:45 (-2.2h)

4.2 May 28 Jim Stearns Submittal (Decision Tree, Score 0.62835)

Based upon May 17 Pat Leahy Submittal (Score 0.59642), with one additional predictor: Species bit vectors.

Feature Selection

Input:

- Attributes: Lat/Long, Month, Tmin/Tmax/Tavg, Species bit vectors.
- Classified Attribute: WnvPresent. Two-level factor: “No”, “Yes”.
- Observations: all WnvPresent records, plus an equal number of !WnvPresent records, randomly sampled.

Output: train and test datasets in ARFF format for modeling in Weka.

```
trainRecs <- read.arff(paste0(workingSubDir, "/", "train", "Master", ".arff"))
testRecs <- read.arff(paste0(workingSubDir, "/", "test", "Master", ".arff"))
stopifnot(nrow(trainRecs) == wnvpTrainFileNRecs)
stopifnot(nrow(testRecs) == wnvpTestFileNRecs)
```

```
colsToKeep=c("Latitude", "Longitude", "Month", "Tmin", "Tmax", "Tavg", "WnvPresent")
colsToKeep=c(colsToKeep, "SpeciesCULEX ERRATICUS", "SpeciesCULEX PIPIENS",
             "SpeciesCULEX PIPIENS/RESTUANS", "SpeciesCULEX RESTUANS", "SpeciesCULEX SALINARIUS",
             "SpeciesCULEX TARSALIS", "SpeciesCULEX TERRITANS", "SpeciesUNSPECIFIED CULEX")
trainRecs <- trainRecs [,names(trainRecs) %in% colsToKeep]
testRecs <- testRecs [,names(testRecs) %in% colsToKeep]
```

Undersample: use all the WnvPresent==True samples. Randomly select an equal number of False samples. Use that for the training data set.

```
curModelIdx <- "0528JS"
allWnvPresentTrainRecs <- trainRecs[trainRecs$WnvPresent=="Yes",]
nFalseObservationsToUse <- nrow(allWnvPresentTrainRecs)
allNotWnvPresentTrainRecs <- trainRecs[trainRecs$WnvPresent=="No",]
sampleNotWnvPresentTrainRecs <- allNotWnvPresentTrainRecs[
    sample(nrow(allNotWnvPresentTrainRecs), nFalseObservationsToUse),]

train_0528JS <- rbind(allWnvPresentTrainRecs, sampleNotWnvPresentTrainRecs)
write.arff(train_0528JS,
           paste0(workingSubDir, "/", "train", curModelIdx, ".arff"),
           eol = '\n', relation="WNVPTrainDataset")
str(train_0528JS)
```

```
## 'data.frame':    1102 obs. of  15 variables:
##  $ Month                        : Factor w/ 6 levels "05","06","07",..: 3 3 3 3 3 3 4 4 4 4 ..
##  $ SpeciesCULEX ERRATICUS       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ SpeciesCULEX PIPIENS         : num  0 0 0 1 1 1 0 0 0 0 ...
##  $ SpeciesCULEX PIPIENS/RESTUANS: num  1 1 1 0 0 0 1 1 1 1 ...
##  $ SpeciesCULEX RESTUANS        : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ SpeciesCULEX SALINARIUS      : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ SpeciesCULEX TARSALIS        : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ SpeciesCULEX TERRITANS       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ SpeciesUNSPECIFIED CULEX     : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Latitude                     : num  41.7 41.7 41.7 41.7 41.7 ...
##  $ Longitude                    : num  -87.5 -87.6 -87.6 -87.6 -87.6 ...
##  $ Tmax                         : num  85 83 83 83 83 83 92 92 92 92 ...
##  $ Tmin                         : num  69 70 70 70 70 70 69 69 69 69 ...
##  $ Tavg                         : num  77 77 77 77 77 77 81 81 81 81 ...
##  $ WnvPresent                   : Factor w/ 2 levels "No","Yes": 2 2 2 2 2 2 2 2 2 2 ...
```

```
# Write all the test recs
write.arff(testRecs,
           paste0(workingSubDir, "/", "test", curModelIdx, ".arff"),
           eol = '\n', relation="WNVPTestDataset")
#str(testRecs)
```

Model

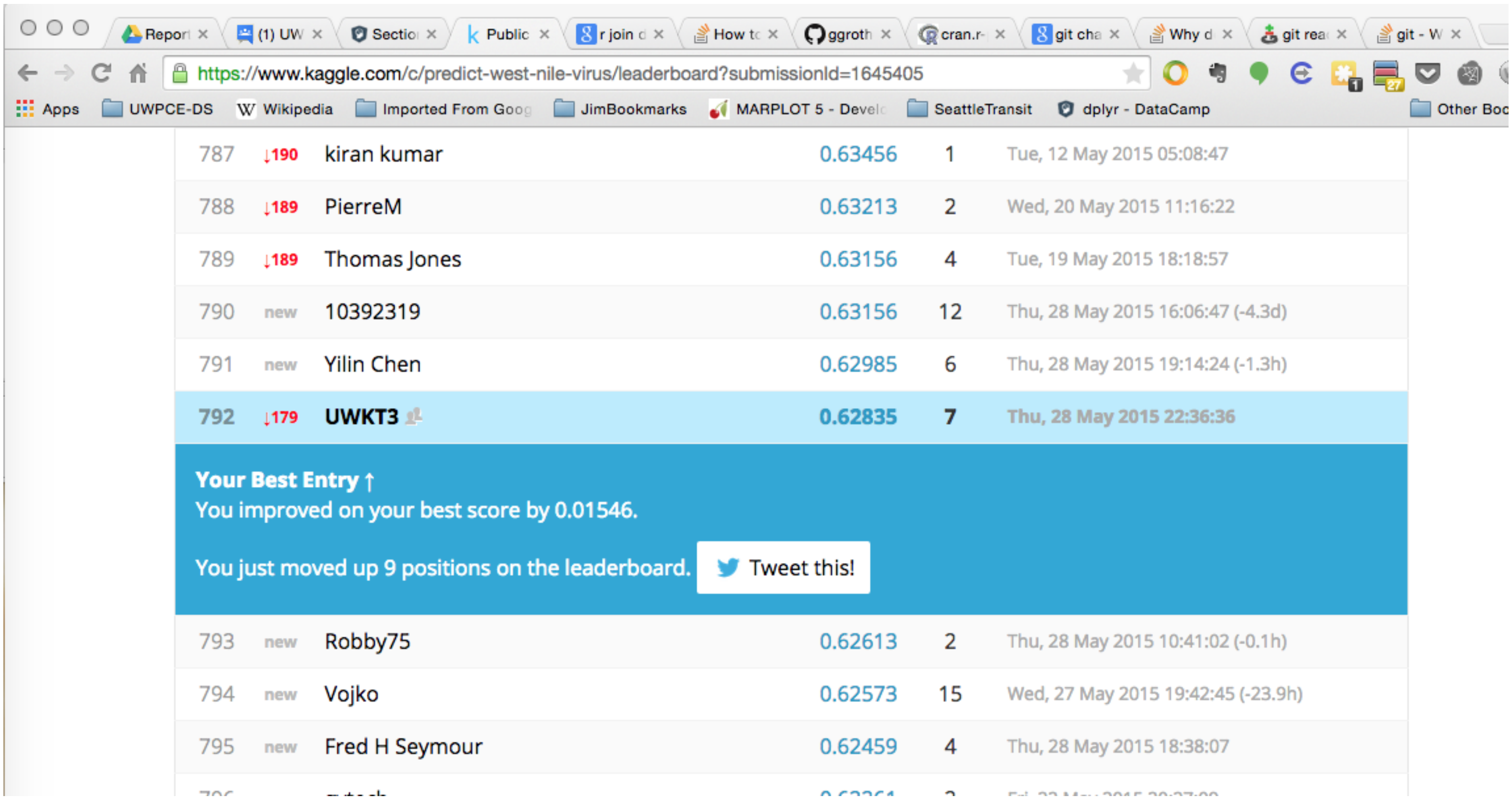
Weka Modeling: same as Pat Leahy using on May 17th:

- Opened the training set in Weka. Had
- Generated a Decision Tree using the J48 classifier.
- Used Pat's parameter settings, "-C 0.5 -M 2".

Notes

- NumMosquitos degraded score. It's not an attribute in test dataset. Not using.
- Dataset ideosynchrocy not dealt with: Dataset rolls over to a new record if number of mosquitos reaches 50.
    - TODO: Combine records with same date, same lat/long, same Species. Sum NumMosquitos, set WnvPresent if any record is WnvPresent.

Screenshot of Leaderboard



https://www.kaggle.com/c/predict-west-nile-virus/leaderboard?submissionId=1645405

787	↓190	kiran kumar	0.63456	1	Tue, 12 May 2015 05:08:47
788	↓189	PierreM	0.63213	2	Wed, 20 May 2015 11:16:22
789	↓189	Thomas Jones	0.63156	4	Tue, 19 May 2015 18:18:57
790	new	10392319	0.63156	12	Thu, 28 May 2015 16:06:47 (-4.3d)
791	new	Yilin Chen	0.62985	6	Thu, 28 May 2015 19:14:24 (-1.3h)
792	↓179	UWKT3	0.62835	7	Thu, 28 May 2015 22:36:36
793	new	Robby75	0.62613	2	Thu, 28 May 2015 10:41:02 (-0.1h)
794	new	Vojko	0.62573	15	Wed, 27 May 2015 19:42:45 (-23.9h)
795	new	Fred H Seymour	0.62459	4	Thu, 28 May 2015 18:38:07

Your Best Entry ↑
You improved on your best score by 0.01546.

You just moved up 9 positions on the leaderboard. Tweet this!

4.3 June 2 Andy Ewing Submittal (Generalized Additive Model (GAM), Score 0.71862)

Submitted by Andy Ewing. Used a sample from the Kaggle WNVP forum: baby steps: breach 0.71 with GAM

Added weather data: daily average temperature and wind speed.

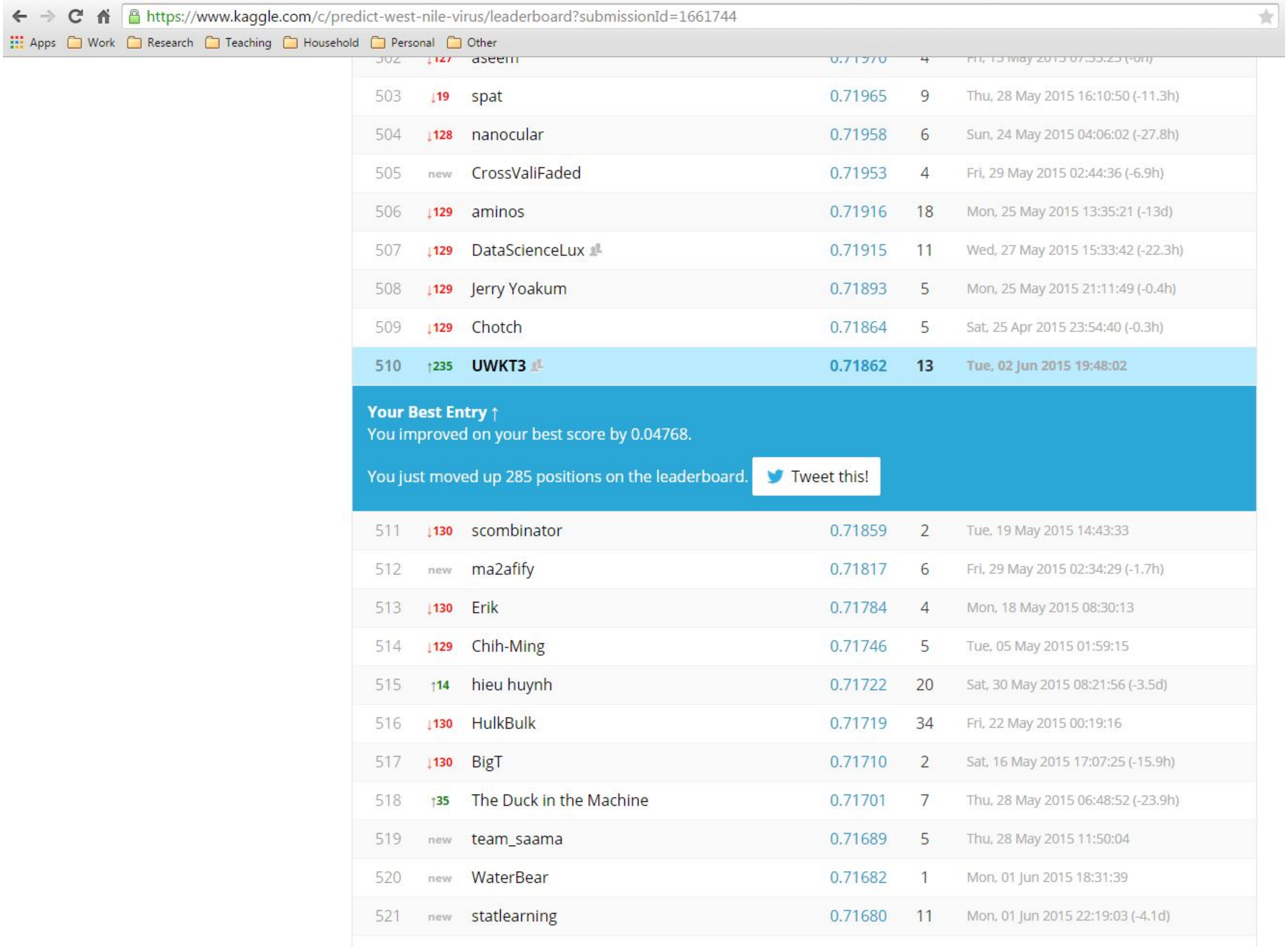
Modeling is done in R, not Weka.

```
source("src/starter_GAM.R", echo=TRUE, verbose=FALSE, print.eval=FALSE,
       prompt.echo=" ", continue.echo="  ")
```

```
##
##  library(Metrics)
##
```

```
##  library(data.table)
##
##  x <- fread("working/trainMaster.csv")
##
##  test <- fread("working/testMaster.csv")
##
##  x$WnvPresent <- ifelse(x$WnvPresent == "No", 0, 1)
##
##  test$WnvPresent <- ifelse(test$WnvPresent == "No",
##       0, 1)
##
##  vSpecies <- c(as.character(x$Species), as.character(test$Species))
##
##  vSpecies[vSpecies == "UNSPECIFIED CULEX"] <- "CULEX ERRATICUS"
##
##  vSpecies[-which(vSpecies == "CULEX PIPIENS" | vSpecies ==
##       "CULEX PIPIENS/RESTUANS" | vSpecies == "CULEX RESTUANS")] = "CULEX OTHER"
##
##  vSpecies <- factor(vSpecies, levels = unique(vSpecies))
##
##  x[, `:=`(Species2, factor(vSpecies[1:nrow(x)], levels = unique(vSpecies)))]
##
##  test[, `:=`(Species2, factor(vSpecies[(nrow(x) + 1):length(vSpecies)],
##       levels = unique(vSpecies)))]
##
##  x[, `:=`(dMonth, as.factor(paste(substr(x$Date, 6,
##       7))))]
##
##  x[, `:=`(dYear, as.factor(paste(substr(x$Date, 1,
##       4))))]
##
##  x$Date = as.Date(x$Date)
##
##  xsDate = as.Date(paste0(x$dYear, "0101"), format = "%Y%m%d")
##
##  x$dWeek = as.numeric(paste(floor((x$Date - xsDate +
##       1)/7)))
##
##  test[, `:=`(dMonth, as.factor(paste(substr(test$Date,
##       6, 7))))]
##
##  test[, `:=`(dYear, as.factor(paste(substr(test$Date,
##       1, 4))))]
##
##  test$Date = as.Date(test$Date)
##
##  tsDate = as.Date(paste0(test$dYear, "0101"), format = "%Y%m%d")
##
##  test$dWeek = as.numeric(paste(floor((test$Date - tsDate +
##       1)/7)))
##
##  my.x = data.frame(x[, list(WnvPresent, dWeek, Species2,
##       Latitude, Longitude, Tavg, AvgSpeed)])
##
##  x1 <- my.x[x$dYear != 2011, ]
##
##  x2 <- my.x[x$dYear == 2011, ]
##
##  require(gam)
##
##  fitCv = gam(WnvPresent ~ s(dWeek) + Species2 + lo(Latitude,
##       Longitude) + Tavg + AvgSpeed, data = x1, family = "binomial")
##
##  p2 <- predict(fitCv, newdata = x2, type = "response")
##
##  auc(x2$WnvPresent, p2)
##
##  fitSubmit <- update(fitCv, data = my.x)
##
##  pSubmit <- predict(fitSubmit, newdata = test, type = "response")
##
##  summary(pSubmit)
##
##  submissionFile <- cbind(test$Id, pSubmit)
##
##  colnames(submissionFile) <- c("Id", "WnvPresent")
##
##  options(scipen = 100, digits = 8)
##
##  write.csv(submissionFile, paste0("working/", "submitGAM.csv"),
##       row.names = FALSE, quote = FALSE)
```

Screenshot of Leaderboard


https://www.kaggle.com/c/predict-west-nile-virus/leaderboard?submissionId=1661744
Apps
Work
Research
Teaching
Household
Personal
Other

502	↓127	aseem	0.71970	4	Fri, 15 May 2015 07:33:23 (-6h)
503	↓19	spat	0.71965	9	Thu, 28 May 2015 16:10:50 (-11.3h)
504	↓128	nanocular	0.71958	6	Sun, 24 May 2015 04:06:02 (-27.8h)
505	new	CrossValiFaded	0.71953	4	Fri, 29 May 2015 02:44:36 (-6.9h)
506	↓129	aminos	0.71916	18	Mon, 25 May 2015 13:35:21 (-13d)
507	↓129	DataScienceLux	0.71915	11	Wed, 27 May 2015 15:33:42 (-22.3h)
508	↓129	Jerry Yoakum	0.71893	5	Mon, 25 May 2015 21:11:49 (-0.4h)
509	↓129	Chotch	0.71864	5	Sat, 25 Apr 2015 23:54:40 (-0.3h)
510	↑235	**UWKT3**	**0.71862**	**13**	**Tue, 02 Jun 2015 19:48:02**

Your Best Entry ↑
You improved on your best score by 0.04768.

You just moved up 285 positions on the leaderboard. Tweet this!

511	↓130	scombinator	0.71859	2	Tue, 19 May 2015 14:43:33
512	new	ma2afify	0.71817	6	Fri, 29 May 2015 02:34:29 (-1.7h)
513	↓130	Erik	0.71784	4	Mon, 18 May 2015 08:30:13
514	↓129	Chih-Ming	0.71746	5	Tue, 05 May 2015 01:59:15
515	↑14	hieu huynh	0.71722	20	Sat, 30 May 2015 08:21:56 (-3.5d)
516	↓130	HulkBulk	0.71719	34	Fri, 22 May 2015 00:19:16
517	↓130	BigT	0.71710	2	Sat, 16 May 2015 17:07:25 (-15.9h)
518	↑35	The Duck in the Machine	0.71701	7	Thu, 28 May 2015 06:48:52 (-23.9h)
519	new	team_saama	0.71689	5	Thu, 28 May 2015 11:50:04
520	new	WaterBear	0.71682	1	Mon, 01 Jun 2015 18:31:39
521	new	statlearning	0.71680	11	Mon, 01 Jun 2015 22:19:03 (-4.1d)

5. Modeling Strategy

TBD. Unclear: discuss modeling strategy here, or in discussion of submittals?

6. Ensemble Model Opportunities

Appendix

A.1 General Setup: Clear environment, set working directory, load libraries, utilities

```
# Clear the working environment of variables, data, functions
rm(list=ls())

# Set working directory for this Kaggle project. Default: pwd.
```

```
#kaggleProjHomeDir <- "."
kaggleProjHomeDir <- "/Users/jimstearns/GoogleDrive/Learning/Courses/UWPCE-DataScience/Course3_Da
setwd(kaggleProjHomeDir)
getwd()

#install.packages("rPython")  # For download from web site with login/pwd.
library(rPython) # For calling python function to download file w/login+pwd
# Package for writing Weka ARFF file format
stopifnot(require("foreign"))
library("foreign")
# Package for calculating great circle distances
stopifnot(require("geosphere"))
library("geosphere")

# Return a data frame with the named column(s) moved to last position.
# Intended usage: move the output classification, WnvPresent, to the last column position.
moveColsToLast <- function(df, colsToMove) {
    df[c(setdiff(names(df), colsToMove), colsToMove)]
}

moveColsToFirst <- function(df, colsToMove) {
    df[c(colsToMove, setdiff(names(df), colsToMove))]
}
```

A.2 Dataset download and unpacking

This R and Python code downloads the WNVP datasets from Kaggle.

Why use Python instead of R? Why not just use read.csv("https://www.gaggle.com/datalocation")?

The read.csv() function does not support SSL when reading from a URL.

R's download.file(, , method="curl") does download HTTPS URLs, but has no facility for establishing authentication credentials for a session. I.e. Kaggle requires a Kaggle login in order to download data files.

There may be a way in R, but I found a way in Python, and using it from R using RPython works.

Some setup is required:

- One's Kaggle username and password must be defined as environment variables where R is running.
- Easiest way to set environment variable for R: Create (add to) ~/.Renviron file (kaggleUsername="XXXX" and kagglePassword="YYYY").

```
wnvpTrainFilename <- "train.csv"
wnvpTestFilename <- "test.csv"
wnvpWeatherFilename <- "weather.csv"
wnvpSprayFilename <- "spray.csv"
kaggleDatasets = c(
    wnvpTrainFilename,
    wnvpTestFilename,
    wnvpWeatherFilename,
    wnvpSprayFilename)
dataSubDir <- "input"  # Kaggle convention
```

```r
workingSubDir <- "working" # Kaggle convention: massaged datasets - and output - go here.

wnvpTrainFileNRecs <- 10506 # Observation records in training file. Excludes header record.
wnvpTestFileNRecs <- 116293 # Records in test file supplied by Kaggle. Submission record cnt mu
# If download from Kaggle required, and user and pwd are empty (default),
# then user will be prompted for these two values.
kaggleUsername <- ""
kagglePassword <- ""

allKaggleFilesArePresent <- function() {
    filesAllFound <- TRUE
    for (file in kaggleDatasets) {
        if (!file.exists(paste0(dataSubDir, "/", file))) {
            print(paste("Error: could not find unzipped Kaggle file in PWD:", file))
            filesAllFound <- FALSE
        }
    }
    return(filesAllFound)
}

downloadMissingKaggleFiles <- function() {
    python.load("src/UrlFileDownloaderWithLogin.py")

    kaggleUsername = Sys.getenv("kaggleUsername")
    kagglePassword = Sys.getenv("kagglePassword")
    if (kaggleUsername == "" || kagglePassword == "") {
        print("Please assign kaggleUsername and kagglePassword environment variables.")
        print("Place in ~/.Renviron entries such as kaggleUsername='YourName'.")
    }
    stopifnot(!(kaggleUsername == ""))
    stopifnot(!(kagglePassword == ""))

    wnvpKaggleDataUrl <-
        "https://www.kaggle.com/c/predict-west-nile-virus/download/"

    for (file in kaggleDatasets) {
        if (file.exists(file))
            next

        urlOfZip <- paste0(wnvpKaggleDataUrl, file, ".zip")
        print(urlOfZip)
        # Use a python method to download from URL with login and password.
        # Download to subdirectory "input" and filename w/o the .zip suffix.
        python.call("Download", urlOfZip,
                    kaggleUsername, kagglePassword ,
                    paste0(dataSubDir, "/", file, ".zip"))
    }
}

unzipDownloadedFiles <- function() {
    for (file in kaggleDatasets) {
        zippedFile <- paste0(dataSubDir, "/", file, ".zip")
        print(paste0("Unzip: ", zippedFile))
        if (file.exists(zippedFile)) {
            if (file.exists(file)) {
                print(sprintf("Warning: removing existing file %s\n", file))
                file.remove(file)
            }
            unzip(zippedFile, exdir=dataSubDir)
            print(sprintf("Unzipped: %s\n", zippedFile))
        }
    }
}

if (!allKaggleFilesArePresent()) {
    print(paste("Not all needed Kaggle datasets are present in PWD;",
                "attempting to download from Kaggle web site."))
    downloadMissingKaggleFiles()
    unzipDownloadedFiles()
}
```

Alternatively, files can be downloaded manually.

File UrlFileDownloaderWithLogin.py:

```
__author__ = 'jimstearns'
""" Download a file at a URL at a web site that requires a user name and password.
"""

import logging
import os       # File utilities

# Python package "requests": "Python HTTP for Humans" by Kenneth Reitz. Current version: 2.7.0.
# Documented at http://docs.python-requests.org/en/latest/
# To install from the command line: "pip install requests"
# (On Mac, sudo may be required. Also pip2.7 instead of pip, depending on default Python version)

import requests # Http GET, POST

def Download(url, username, password, local_filename):
    # Login to web site such as Kaggle and retrieve the data. Use POST rather than GET as as to
    # send login info in body of HTTP request rather than in query string portion of URL.

    # Limitation: when used by Python version < 2.7.9, an "InsecureRequestWarning" is generated.
    # TODO: Fix. Details: https://urllib3.readthedocs.org/en/latest/security.html#insecureplatfor
    # Workaround: log warnings to file, not stdout.
    logging.captureWarnings(True)

    if (os.path.exists(local_filename)):
        os.remove(local_filename)

    # This won't get the file, but use the return value URL in a follow-on POST:
```

```
    r = requests.get(url)

    login_info = {'UserName': '{0}'.format(username), 'Password': '{0}'.format(password) }
    print(login_info)
    r = requests.post(r.url, data = login_info)
    print("POST (w/login info): {0}\n".format(r.status_code))

    # Write the data to a local file one chunk at a time.
    chunk_size = 512 * 1024 # Reads 512KB at a time into memory
    with open(local_filename, 'wb') as fd:
        for chunk in r.iter_content(chunk_size): # Reads 512KB at a time into memory
            if chunk: # filter out keep-alive new chunks
                fd.write(chunk)

    if (os.path.exists(local_filename)):
        return(True)
    else:
        return(False)
```

A.3 Prepare Weka results as ARFF file as submittal file to Kaggle as CSV

File PrepareWekaArffResultsForKaggleCsvSubmittal:

```
# Script to read in ARFF file created by Weka modeler,
# strip all attributes except the predicted classification (here, "WnvPresent"),
# add an Id column with a sequence number equal to the record number; and
# write as a CSV file.

library("foreign") # For read.arff
wnvpTestFileNRecs <- 116293 # Records in test file supplied by Kaggle. Submission record cnt must

dataSubDir <- "../Submissions/Submission_0604_JS_1/" # Modify as needed
fileBaseName <- "test0528JS_WekaClassified"  # Change for your filename. Note: no suffix.

fileBasePath <- paste0(dataSubDir, fileBaseName)
testClassified_df <- read.arff(paste0(fileBasePath, ".arff"))
stopifnot(nrow(testClassified_df) == wnvpTestFileNRecs)

Id <- seq(1:wnvpTestFileNRecs)
colsToKeep <- c("predicted WnvPresent")
testClassified_df <- cbind(Id, testClassified_df[names(testClassified_df) %in% colsToKeep])
names(testClassified_df) <- c("Id", "WnvPresent")
# Write "No" as 0 and "Yes" as 1
testClassified_df$WnvPresent <- ifelse(testClassified_df$WnvPresent == "No", 0, 1)
str(testClassified_df)

write.csv(testClassified_df, paste0(fileBasePath, ".csv"), row.names=FALSE)
```