UWPCE DataScience3 KaggleWNVP Project Report

Kaggle Team 3

2 June 2015

Background

WNVP: "West Nile Virus Prediction"

URL of Kaggle Competition.

Goal of Competition: "Predict West Nile virus in mosquitos across the city of Chicago"

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.

Team Members

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

Data Preparation

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
- 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."
```

Appendix

General Setup: Clear environment, set working directory, load libraries

```
# 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
```

Dataset download and unpacking

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

Alternatively, files can be downloaded manually.

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

# 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("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 Data 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()
}
```

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)
```