Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC2115 – Programación como Herramienta para la Ingeniería (II/2018)

Laboratorio 6: API, *Web Scraping* y Geopandas

Objetivos

- Utilizar conceptos básicos de *Web Scraping* para extraer información de una página web.
- Utilizar la librería Geopandas para la visualización de mapa.
- Utilizar la librería Pandas para el procesamiento de datos y responder consultas acerca de ellos.

Entrega

- **Lenguaje a utilizar:** Python 3.6
- **Lugar:** repositorio GitHub
- **Fecha límite de entrega: Jueves 15 de noviembre de 2018 a las 23:59 horas**
- **Formato de entrega:** dos archivos con la solución propuesta, uno en formato **.py** y el otro en **.ipynb**. Deben ser exactamente iguales (el ayudante revisará cualquiera de ellos). Adicionalmente, incluya un archivo **README.md** con información útil para el ayudante corrector.

Introducción

¡Micro de mI@rd@ que no pasa nunca! Cansado de vivir en la incertidumbre de no saber cuánto tiempo tendrás que esperar para subirte en el bendito Transantiago, decides utilizar tus asombrosas habilidades de programación. Has escuchado a tu amigo Hans hablar de la API Transantiago API que contiene información del tiempo de llegada de buses del Transantiago. Algunas de las informaciones que se pueden consultar en la API son las siguientes:

- Tiempo de llegada los próximos buses a un paradero.
- Conocer cuando un bus en específico se encuentra fuera de operación.
- Distancia del bus al paradero.

Al ver todo el potencial de conocimiento que puedes extraer de esta API, decides que ha llegado la hora de recolectar la información de la página mediante *requests*. Para esto deberás realizar una *request* en forma de *GET* a este link: `https://api.scltrans.it/v2/stops/[PARADERO]/next_arrivals` donde [PARADERO] debe ser reemplazado por el número de algún paradero válido. Por ejemplo: `https://api.scltrans.it/v2/stops/PF15/next_arrivals`. Esto retornará un diccionario con la información de llegada de los buses de dicho paradero.

1 Parte 1: extracción de la información (2.5 pts.)

La primera parte del laboratorio consiste en realizar llamados a la API a todos los paraderos de un servicio de bus. Lamentablemente, no cuentas con la lista de todos los paraderos de bus dado un servicio, y no se ha encontrado ninguna API que sirva para este proposito. Sin embargo, existe la página TyggerMilenio que cuenta con varias tablas con la información buscada. Por lo tanto, deberás utilizar tus conocimiento de *web scraping* para recolectar solamente la información necesaria para este laboratorio. Algunas consideraciones a tener en cuenta:

- Cada página cuenta con un elemento HTML $< a >$ que posee el *link* a la siguiente página con más tablas. Deberás buscar este elemento en la página para obtener el enlace a la siguiente. Habrá una penalización en tu nota si tu código tiene de forma predeterminada los *links* de cada página de TyggerMilenio
- Hace tiempo, TyggerMilenio fue atacado por los malvados alumnos de Programación Avanzada que agregaron tablas falsas a la página. Para nuestro beneficio, esas tablas incluyen una clase (HTML class) llamada **fake**, mientras que las tablas verdaderas tienen una clase llaman **true**. Por lo tanto, deberás realizar *scrapping* exclusivamente de las tablas cuya clase son **true**.
- Las tablas de la página, cuentan con 3 columnas: Código, NServicios y Servicios, la primera corresponde al paradero, la segunda al número del servicio asociado a dicho paradero y la tercera indica cada bus que pasa por dicho paradero. Cada bus está separado por un punto y coma (";") y poseen una letra al final, la cual puede ser **I** o **R**, que significan Ida y Regreso respectivamente.
- Para este laboratorio, solo trabajaremos con cantidades pares de NServicios.[1]. Por lo tanto, tu *scrapping* solo guardará información de cada paradero y bus asociado a un servicio par.

Dada esta descripción, deberás crear 2 *script* en esta parte. El primero será uno que realice *web scrapping* de TyggerMilenio y guarde en un archivo CSV la información recolectada. Esta debe ser guardada en un archivo llamado `paraderos.csv` y las columnas deben ser:

1. "servicio": número del servicio.
2. "paradero": código del paradero.
3. "bus": número del bus.
4. "dirección": debe ser "I" si va de ida o "R" si va de regreso.

Podrás notar que en la página, cada fila representa 1 paradero con todos sus buses, mientras que en este nuevo CSV, cada fila solo muestra la información de 1 solo bus por paradero. Por lo tanto, una fila de la tabla que tiene 5 buses van a corresponder a 5 filas en el nuevo CSV.

Finalmente, luego de obtener la información de cada paradero. Deberás crear el segundo *script* que consuma la API de Transantiago API y que realice las *requests* para cada paradero existente en `paraderos.csv`. Para cada paradero, deberás realizar 3 *requests* con intervalo de al menos 5 minutos. De este modo, obtendremos la información de cada paradero en 3 momentos distintos. Deberás guardar los datos recolectados en un archivo denominado `tiempo_estimado.csv`, que siga la siguiente estructura:

`timestamp,paradero,bus,patente,tiempo_de_demora,distancia_del_paradero`.

[1] Uno de los ayudantes tuvo muchas pesadillas con los números impares y por eso no quiere verlos nunca más

La columna `timestamp` corresponde al tiempo cuando se realizó el llamado a la API, con dicho tiempo se debe ser capaz de identificar como mínimo el mes, día, hora, minuto y segundo. Se recomienda utilizar la codificación UNIX para almacenar el tiempo.

2 Parte 2: visualización de información (2.5 pts)

Advertencia: Parte importante de esta sección requiere del archivo `tiempo_estimado.csv` obtenido en la parte 1.

¡Felicitaciones! Has logrado obtener información de los tiempos estimados para buses del Transantiago. Decides compartir tu éxito con tu familia... y nadie te entiende nada acerca de *timestamps* y archivos .csv. Requieres de una manera más intuitiva de comprender la misma información. "Dibújame un mapa de Santiago con puntitos para las micros y te entenderé", dicen en tu hogar. Ese es un buen resumen de lo que harás en esta sección.

Para esta parte es importante que investiguen sobre shapefile. En palabras simples, un *shapefile* es un estándar multiarchivos que se utiliza para intercambio de información geográfica. Para esta tarea, utilizarán este estándar para ver el mapa de Santiago con el trazado de la ruta de un recorrido en específico. Además, podrán ubicar los puntos obtenidos en la parte anterior para que tengan una referencia visual de las posiciones de las micros en cada instante, así como la ubicación de los paraderos.

2.1 Archivos a utilizar

Para la visualización, se les entregarán dos carpetas con archivos:

- `Shape Paraderos`: el *shapefile* que contiene las ubicaciones de cada paradero. Contiene los archivos `Paradas.shp`, `Paradas.shx`, `Paradas.dbf` y `Paradas.prj`
- `Shape Trazados`: aquí estará la ubicación de las rutas de los buses. Contiene los archivos `Trazados.shp`, `Trazados.shx`, `Trazados.dbf` y `Trazados.prj`

Advertencia: no debe subir estas carpetas a sus repositorios, sino tendrá una penalización por no seguir formato de entrega. Los ayudantes agregarán dichas carpetas al momento de corregir. Si es que llegan a subirla, deben asegurarse de eliminarlas antes de la entrega final.

Con la información anterior, se les pide que muestren un mapa de Santiago y marquen en él todos los paraderos y trazado del servicio. Por ejemplo, dado el recorrido 210, debes mostrar de un color todas las 210, de otro color sus paraderos, y con un tercer color, la ruta de la micro. Para mostrar el mapa de Santiago, deben utilizar el *shapefile* que se utiliza en el material de clases y sobre este *shapefile* incluir los paraderos, buses y trazado de la ruta. Es decir, superponer el archivo del material de clases, con los archivos entregados en este laboratorio.

Se desea también ver el estado de las demás micros obtenido en la parte anterior, por lo tanto, en el mapa también deberán ir marcadas todas las demás micros que se detienen en algún paradero del recorrido principal indicado. Para esto debes utilizar un color diferente a los usados anteriormente. El resultado final debe ser una visualización que permita entender cómo cambió el estado de cada bus relacionado con el recorrido principal en los tres momentos en los que se llamó a la API en la parte 1.

Podrás notar que gracias a la API, vas a disponer de la distancia del bus a un paradero en especifico y con los *shapefile* obtendrás la posición de cada paradero. Dada esa información, tendrás que crear una forma de posicionar cada bus en el mapa para lograr diferenciar entre qué paraderos está cada bus y si hay

buses más cerca del paradero o más lejos.

La explicación de cómo trabajar con los mapas (llámese cargarlos o hacer marcas en ellos) estará disponible en el material de esta semana.

2.2 Zoom del mapa

Un aspecto importante a considerar es el *zoom* del mapa. Santiago es muy grande y, por lo mismo, cuando ubiquen buses y trazados en el mapa podrían verse muy pequeños (o incluso no observarse). Se les pide que hagan un zoom adecuado al mapa para que se ajuste al trazado de la micro que se les indique.

3 Parte 3: procesamiento y consultas de la información (1 pts)

Dado el archivo `tiempo_estimado.csv`, debes crear funciones que mediante Pandas, permitan obtener información adicional.

- `avance_total(bus)`: dada la patente de un bus, deberán tomar el primer y último momento obtenido por la API y utilizar la posición que tiene para retornar una estimación del avance total que tuvo dicho bus. Para esta función, deberán realizar algunos supuestos, los cuales deben ser explicitados en el README.md. Si la patente no existe, debes retornar -1.
- `buses_llegando_a_paradero(paradero, n)`: dada una distancia n en metros, se desea encontrar los buses que estén a menos de n metros del paradero entregado como parámetro de la función
- `buses_mas_cercanos(n)`: dada una distancia n en metros, retorna todos los pares de buses que se encuentran a n metros o menos entre ellos.[2]

En esta sección se evaluará que incluyan todos los supuestos que consideren necesarios en el Readme.md. El no seguir esta indicación podría llevarlos a una penalización en su nota final.

Avance parcial

Para la revisión de avance, se espera que las parejas hayan logrado extraer la información de TyggerMilenio, es decir, el *web scrapping* de la parte 1 completa.

Política de Integridad Académica

Los alumnos de la Escuela de Ingeniería deben mantener un comportamiento acorde al Código de Honor de la Universidad:

"Como miembro de la comunidad de la Pontificia Universidad Católica de Chile me comprometo a respetar los principios y normativas que la rigen. Asimismo, prometo actuar con rectitud y honestidad en las relaciones con los demás integrantes de la comunidad y en la realización de todo trabajo, particularmente en aquellas actividades vinculadas a la docencia, el aprendizaje y la creación, difusión y transferencia del conocimiento. Además, velaré por la integridad de las personas y cuidaré los bienes de la Universidad."

[2] Esta información podría ser útil para quienes administran el sistema de Transantiago

En particular, se espera que mantengan altos estándares de honestidad académica. Cualquier acto deshonesto o fraude académico está prohibido; los alumnos que incurran en este tipo de acciones se exponen a un procedimiento sumario. Ejemplos de actos deshonestos son la copia, el uso de material o equipos no permitidos en las evaluaciones, el plagio, o la falsificación de identidad, entre otros. Específicamente, para los cursos del Departamento de Ciencia de la Computación, rige obligatoriamente la siguiente política de integridad académica en relación a copia y plagio: Todo trabajo presentado por un alumno (grupo) para los efectos de la evaluación de un curso debe ser hecho individualmente por el alumno (grupo), sin apoyo en material de terceros. Si un alumno (grupo) copia un trabajo, se le calificará con nota 1.0 en dicha evaluación y dependiendo de la gravedad de sus acciones podrá tener un 1.0 en todo ese ítem de evaluaciones o un 1.1 en el curso. Además, los antecedentes serán enviados a la Dirección de Docencia de la Escuela de Ingeniería para evaluar posteriores sanciones en conjunto con la Universidad, las que pueden incluir un procedimiento sumario. Por "copia" o "plagio" se entiende incluir en el trabajo presentado como propio, partes desarrolladas por otra persona. Está permitido usar material disponible públicamente, por ejemplo, libros o contenidos tomados de Internet, siempre y cuando se incluya la cita correspondiente.