

Universidad de San Sebastián
Taller de Programación II
Profesora: PhD. Liset González
Clase 14: Aplicaciones Web usando Streamlit

Guía de Ejercicio: Crear una Aplicación Web de Uber Pickups en NYC

Objetivo: Desarrollar una aplicación web interactiva en Python usando Streamlit que visualice y analice datos de recogidas de Uber en la ciudad de Nueva York.

Instrucciones:

1. Revise el siguiente código base y entienda cómo funciona cada sección.
2. Implemente y ejecute el código en su entorno de desarrollo con el nombre `uber_example.py`.
3. Responda las preguntas al final de la guía para demostrar su comprensión y aplicar los conocimientos adquiridos.

```
import streamlit as st
import pandas as pd
import numpy as np

st.title('Uber pickups in NYC')

DATE_COLUMN = 'date/time'
DATA_URL = ('https://s3-us-west-2.amazonaws.com/'
            'streamlit-demo-data/uber-raw-data-sep14.csv.gz')

@st.cache_data
def load_data(nrows):
    data = pd.read_csv(DATA_URL, nrows=nrows)
    lowercase = lambda x: str(x).lower()
    data.rename(lowercase, axis='columns', inplace=True)
    data[DATE_COLUMN] = pd.to_datetime(data[DATE_COLUMN])
    return data

data_load_state = st.text('Loading data...')
data = load_data(10000)
data_load_state.text("Done! (using st.cache_data)")

if st.checkbox('Show raw data'):
    st.subheader('Raw data')
    st.write(data)

st.subheader('Number of pickups by hour')
hist_values = np.histogram(data[DATE_COLUMN].dt.hour, bins=24, range=(0,24))[0]
st.bar_chart(hist_values)

# Some number in the range 0-23
hour_to_filter = st.slider('hour', 0, 23, 17)
filtered_data = data[data[DATE_COLUMN].dt.hour == hour_to_filter]

st.subheader('Map of all pickups at %s:00' % hour_to_filter)
st.map(filtered_data)
```

Parte 1: Entendiendo el Código Base

1. ¿Qué hace la función `@st.cache_data` y por qué es importante en una aplicación de Streamlit?
2. ¿Qué tipo de datos estamos cargando desde la URL proporcionada y cómo se procesan al leerlos con `pd.read_csv()`?
3. Explica cómo se usa la función lambda en el código y cuál es su propósito.
4. ¿Qué papel juega la conversión de `DATE_COLUMN` a formato `datetime` y por qué es útil?

Parte 2: Interactividad y Visualización

5. ¿Qué ocurre cuando se marca la casilla de verificación `Mostrar datos en bruto` en la aplicación?
6. ¿Cómo se utiliza `st.slider` en el código y qué efecto tiene en la visualización de los datos?
7. Explica cómo se construye el histograma de `Número de recogidas por hora`. ¿Qué significa cada parte de `np.histogram`?
8. ¿Qué muestra el mapa generado con `st.map(filtered_data)` y cómo se filtran los datos que se muestran?

Parte 3: Modificación y Extensión

9. Agrega una funcionalidad para que los usuarios puedan seleccionar un rango de horas en lugar de una sola hora. ¿Cómo lo harías?
10. Añade un control de selección para filtrar las recogidas de Uber por día de la semana. ¿Qué cambios tendrías que hacer en el código para incluir esta funcionalidad?
11. ¿Cómo podrías modificar el código para que se muestre el número de recogidas por día de la semana en un gráfico de barras?

Parte 4: Análisis y Evaluación

12. ¿Qué ventajas tiene usar Streamlit para este tipo de visualizaciones interactivas en comparación con un script estático de Python?
13. Imagina que los datos se cargan lentamente. ¿Qué alternativas podrías explorar para mejorar el rendimiento de la aplicación?
14. ¿Qué cambios harías en la función `load_data` si quisieras cargar datos desde un archivo local en vez de una URL?
15. Agrega un gráfico de líneas que muestre cómo varía el número de recogidas a lo largo de un día completo. ¿Qué pasos seguirías para hacerlo?
16. Implementa un filtro para mostrar solo las recogidas que ocurrieron durante el fin de semana. ¿Cómo ajustarías el código?
17. Investiga y añade un `st.selectbox` que permita al usuario cambiar entre diferentes tipos de gráficos (por ejemplo, barras, líneas, área).