

Assignment 11 Visualization

×

Select the provinces

Choose an option

☒ Select all provinces

Select the start date

2017/08/01

Select the end date

2017/08/31

Deploy

Analysis of the Thailand Rain Daily Dataset

This is a simple example of a Streamlit app that displays the Thailand Rain Daily dataset.

| | code | name | latitude | longitude | tambon | amphoe | province | date |
|---|------|----------------|----------|-----------|----------|---------------|----------|---------------------|
| 0 | ABRT | อนุบาลร้อยเอ็ด | 16.0532 | 103.6598 | รอนเมือง | เมืองร้อยเอ็ด | ร้อยเอ็ด | 2017-08-01 00:00:00 |
| 1 | ABRT | อนุบาลร้อยเอ็ด | 16.0532 | 103.6598 | รอนเมือง | เมืองร้อยเอ็ด | ร้อยเอ็ด | 2017-08-02 00:00:00 |
| 2 | ABRT | อนุบาลร้อยเอ็ด | 16.0532 | 103.6598 | รอนเมือง | เมืองร้อยเอ็ด | ร้อยเอ็ด | 2017-08-03 00:00:00 |
| 3 | ABRT | อนุบาลร้อยเอ็ด | 16.0532 | 103.6598 | รอนเมือง | เมืองร้อยเอ็ด | ร้อยเอ็ด | 2017-08-04 00:00:00 |
| 4 | ABRT | อนุบาลร้อยเอ็ด | 16.0532 | 103.6598 | รอนเมือง | เมืองร้อยเอ็ด | ร้อยเอ็ด | 2017-08-05 00:00:00 |

The dataset has the following description:

| | latitude | longitude | date | rain |
|-------|----------|-----------|---------------------|--------|
| count | 11,284 | 11,284 | 11284 | 11,284 |
| mean | 15.6112 | 100.98 | 2017-08-16 00:00:00 | 5.7256 |
| min | 5.772 | 97.9177 | 2017-08-01 00:00:00 | 0 |

×

Select the provinces

Choose an option

☒ Select all provinces

Select the start date

2017/08/01

Select the end date

2017/08/31

Deploy

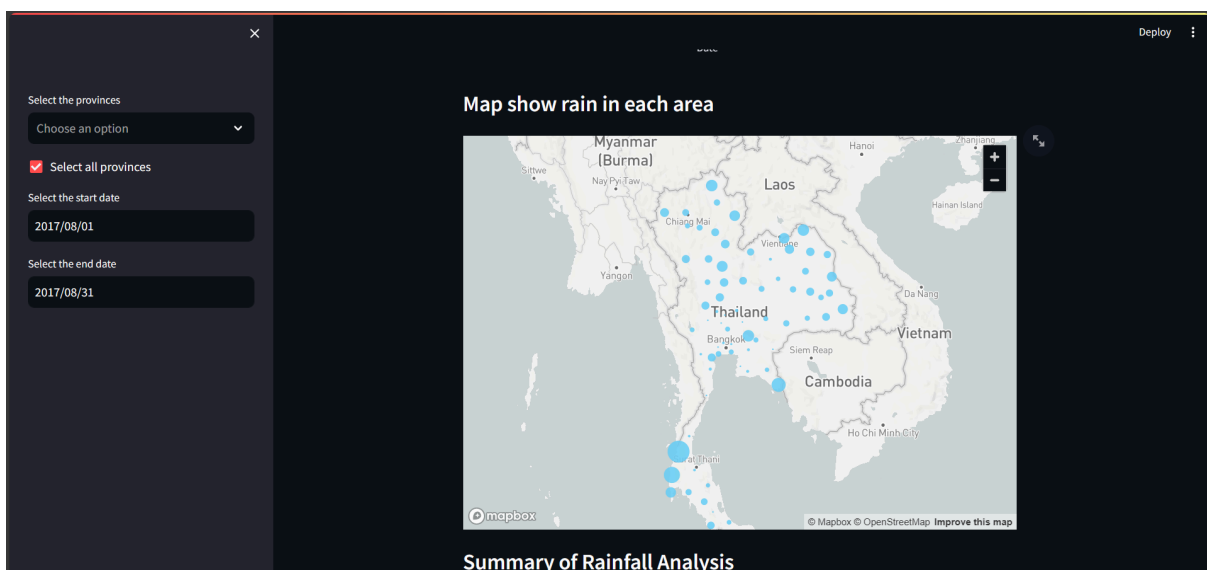
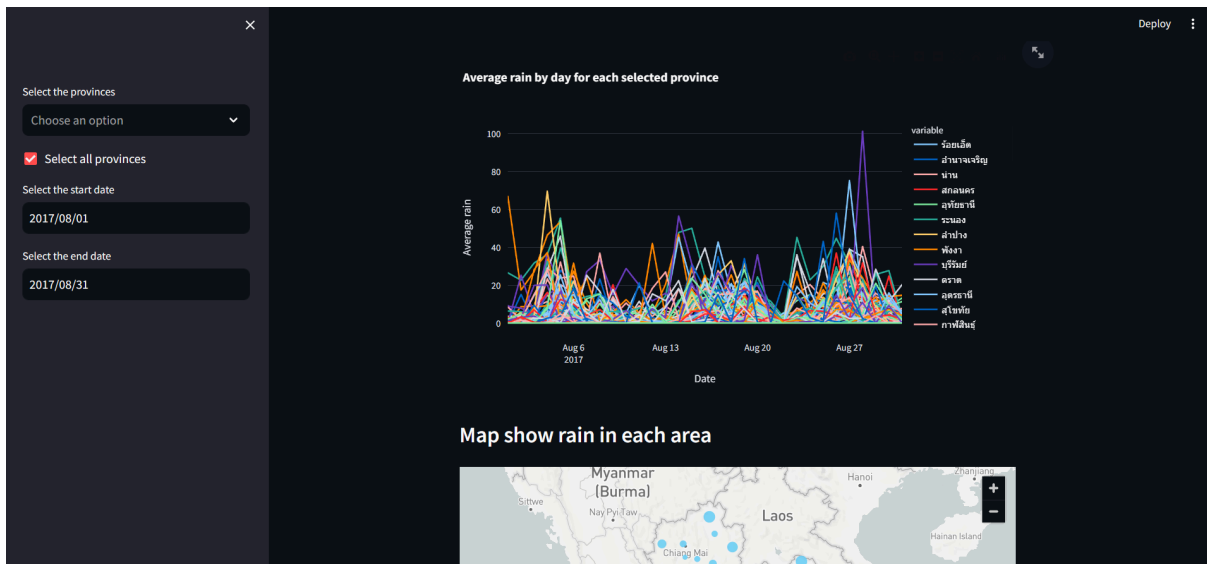
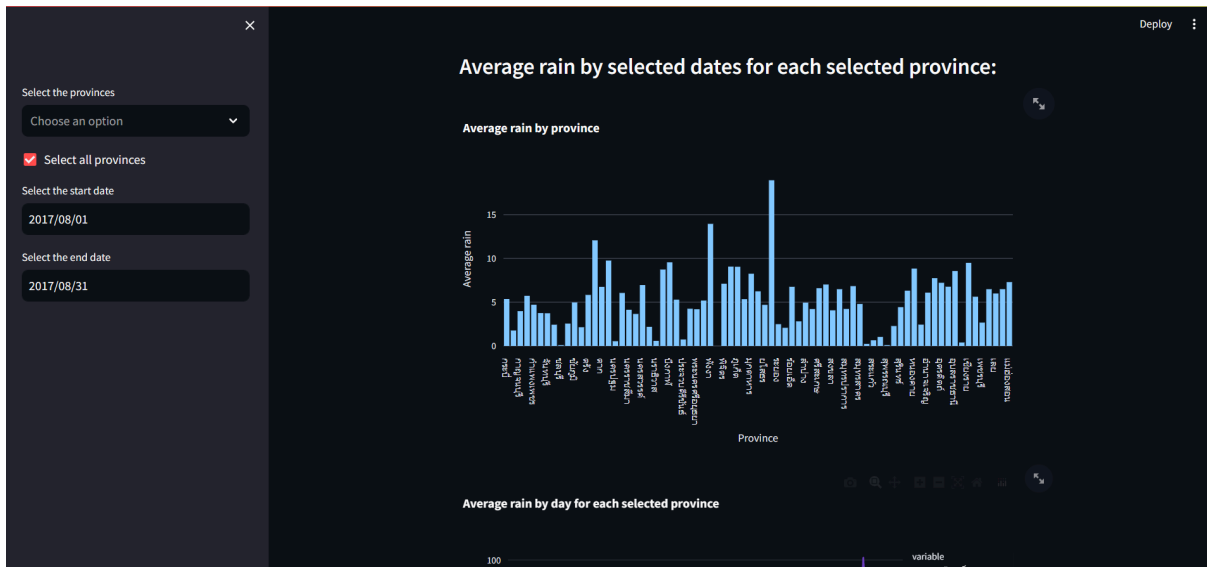
| | | | | | | | | |
|---|------|----------------|---------|----------|----------|---------------|----------|---------------------|
| 0 | ABRT | อนุบาลร้อยเอ็ด | 16.0532 | 103.6598 | รอนเมือง | เมืองร้อยเอ็ด | ร้อยเอ็ด | 2017-08-01 00:00:00 |
| 2 | ABRT | อนุบาลร้อยเอ็ด | 16.0532 | 103.6598 | รอนเมือง | เมืองร้อยเอ็ด | ร้อยเอ็ด | 2017-08-03 00:00:00 |
| 3 | ABRT | อนุบาลร้อยเอ็ด | 16.0532 | 103.6598 | รอนเมือง | เมืองร้อยเอ็ด | ร้อยเอ็ด | 2017-08-04 00:00:00 |
| 4 | ABRT | อนุบาลร้อยเอ็ด | 16.0532 | 103.6598 | รอนเมือง | เมืองร้อยเอ็ด | ร้อยเอ็ด | 2017-08-05 00:00:00 |

The dataset has the following description:

| | latitude | longitude | date | rain |
|-------|----------|-----------|---------------------|---------|
| count | 11,284 | 11,284 | 11284 | 11,284 |
| mean | 15.6112 | 100.98 | 2017-08-16 00:00:00 | 5.7256 |
| min | 5.772 | 97.9177 | 2017-08-01 00:00:00 | 0 |
| 25% | 14.4343 | 99.6547 | 2017-08-08 00:00:00 | 0 |
| 50% | 16.0186 | 100.439 | 2017-08-16 00:00:00 | 0.2 |
| 75% | 17.8037 | 102.2691 | 2017-08-24 00:00:00 | 5.6 |
| max | 20.4032 | 105.5039 | 2017-08-31 00:00:00 | 194.8 |
| std | 3.0245 | 1.8022 | None | 11.9629 |

Average rain by selected dates for each selected province:

Average rain by province



×

Select the provinces

Choose an option

☒ Select all provinces

Select the start date

2017/08/01

Select the end date

2017/08/31

Source Code

```
import streamlit as st
import pandas as pd
import pydeck as pdk
import plotly.express as px

st.title("Analysis of the Thailand Rain Daily Dataset")
st.write("This is a simple example of a Streamlit app that displays the Thailand R

@st.cache_data
def load_data():
    df = pd.read_csv("RainDaily_Tabular.csv")
    return df

# Load the dataset
df = load_data()
df['date'] = pd.to_datetime(df['date'])

# Get the unique values of the province and date columns
province = df['province'].unique()
date = df['date'].unique()

provinces = st.sidebar.multiselect("Select the provinces", province)

# Select all provinces
selectAll_provinces = st.sidebar.checkbox("Select all provinces", value=False)
if selectAll_provinces:
    provinces = province
```

Deploy

×

Select the provinces

Choose an option

☒ Select all provinces

Select the start date

2017/08/01

Select the end date

2017/08/31

Source Code

```
provinces = st.sidebar.multiselect("Select the provinces", province)

# Select all provinces
selectAll_provinces = st.sidebar.checkbox("Select all provinces", value=False)
if selectAll_provinces:
    provinces = province

start_date = st.sidebar.date_input("Select the start date", date[0])
end_date = st.sidebar.date_input("Select the end date", date[-1])

dates = pd.date_range(start_date, end_date)

# Display the dataset
st.write(df.head())

# Display the dataset description
st.subheader("The dataset has the following description:")
st.write(df.describe())

# Filter the dataset by selected provinces and dates
df_filtered = df[df['province'].isin(provinces)]
df_filtered = df_filtered[df_filtered['date'].isin(dates)]

# Display the filtered dataset
st.subheader("Average rain by selected dates for each selected province:")

# Plot the average rain by province
# Bar chart
```

Deploy

×

Select the provinces

Choose an option

☒ Select all provinces

Select the start date

2017/08/01

Select the end date

2017/08/31

Source Code

```
# Bar chart
if df_filtered.empty:
    st.write("No data available for the selected provinces and dates.")
else:
    fig = px.bar(df_filtered.groupby('province')['rain'].mean(), y='rain', title='Average rain by province',
                xaxis_title="Province",
                yaxis_title="Average rain",
                )
    st.plotly_chart(fig)

# Line chart
if df_filtered.empty:
    st.write("No data available for the selected provinces and dates.")
else:
    # Create an empty list to store data for each province
    data_for_plot = []

    # Iterate over selected provinces
    for province in provinces:
        df_filtered_province = df_filtered[df_filtered['province'] == province]
        # Append data for this province to the list
        data_for_plot.append(df_filtered_province.groupby('date')['rain'].mean().r

    # Concatenate the data into a single DataFrame
    df_for_plot = pd.concat(data_for_plot, axis=1)

    # Plot using Streamlit's line chart
    fig = px.line(df_for_plot, title='Average rain by day for each selected provin
```

Deploy

×

Select the provinces

Choose an option

☒ Select all provinces

Select the start date

2017/08/01

Select the end date

2017/08/31

```
fig = px.line(df_for_plot, title='Average rain by day for each selected provin
xaxis_title="Date",
yaxis_title="Average rain",
)
st.plotly_chart(fig)

avg_rain_by_province = df_filtered.groupby('province')

#pydeck map for the average rain by province
province_map = df_filtered.groupby('province')[['longitude', 'latitude','rain']].m

# Display the map
st.subheader("Map show rain in each area")

# Map show rain in each area
layer = pdk.Layer(
    "ScatterplotLayer",
    province_map,
    pickable=True,
    opacity=0.8,
    filled=True,
    radius_scale=2500,
    radius_min_pixels=1,
    radius_max_pixels=100,
    line_width_min_pixels=1,
    get_position= ["longitude", "latitude"],
    get_radius = ["rain"] ,
    get_fill_color=[108, 206, 245]
)
```

Deploy

×

Select the provinces

Choose an option

☒ Select all provinces

Select the start date

2017/08/01

Select the end date

2017/08/31

```
if df_filtered.empty:
    view_state = pdk.ViewState(
        longitude=df['longitude'].mean(),
        latitude=df['latitude'].mean(),
        zoom=4.5,
    )
    r = pdk.Deck(
        map_style="mapbox://styles/mapbox/light-v9",
        initial_view_state= view_state,
    )
    st.pydeck_chart(r)
else:
    view_state = pdk.ViewState(
        longitude=province_map['longitude'].mean(),
        latitude=province_map['latitude'].mean(),
        zoom=4.5,
    )
    r = pdk.Deck(
        map_style="mapbox://styles/mapbox/light-v9",
        layers=[layer],
        initial_view_state=view_state,
        tooltip={"text": "Rain: {rain} mm"},
    )
    st.pydeck_chart(r)

# Display summary
st.subheader("Summary of Rainfall Analysis")

if df_filtered.empty:
```

Deploy

×

Select the provinces

Choose an option

☒ Select all provinces

Select the start date

2017/08/01

Select the end date

2017/08/31

```
        tooltip={"text": "Rain: {rain} mm"},
    )
    st.pydeck_chart(r)

# Display summary
st.subheader("Summary of Rainfall Analysis")

if df_filtered.empty:
    summary_text = "The analysis of the rainfall data shows that there is no data
st.write(summary_text)
else:
    highest_rain = avg_rain_by_province['rain'].mean().max()
    lowest_rain = avg_rain_by_province['rain'].mean().min()

    summary_text = f"The analysis of the rainfall data shows that the province wit
st.write(summary_text)

# Display the source code
st.subheader("Source Code")
with open('app.py', "r", encoding="utf-8") as f:
    code = f.read()
st.code(code, language="python")
```

Deploy