

Real-Time Weather Condition Analysis

May 6th, 2022, presented By

Kangrui Li(kl3350), Xiaohang He(xh2509),

Yuhang Wang(yw3733), Linyang Han(lh3096)



- I. Problem Definition
- II. Data Processing
- III. Visualization
- IV. Results Demo
- V. Future Work



- I. Problem Definition**
- II. Data Processing
- III. Visualization
- IV. Results Demo
- V. Future Work



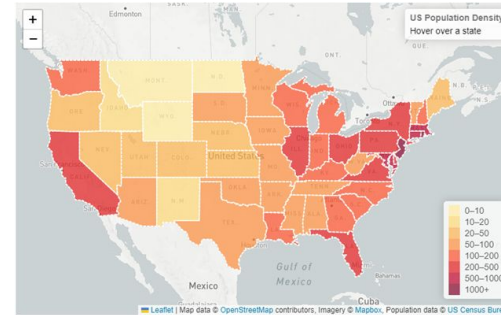
Problem

- Idea:

- Obtain local weather information via devices
- Cannot search easily from different places

- What we want:

- A heat map showing weather condition nationally
- Updating frequently & simple to use



Problem

- How we implement:

- Access weather data via NOAA
- Process Data
- Apply Django to visualize

- The outcomes:

- A heat map with all the states in America
- Weather condition displayed on the map

- I. Problem Definition
- II. Data Processing**
- III. Visualization
- IV. Results Demo
- V. Future Work



Data Processing

Raw Data:

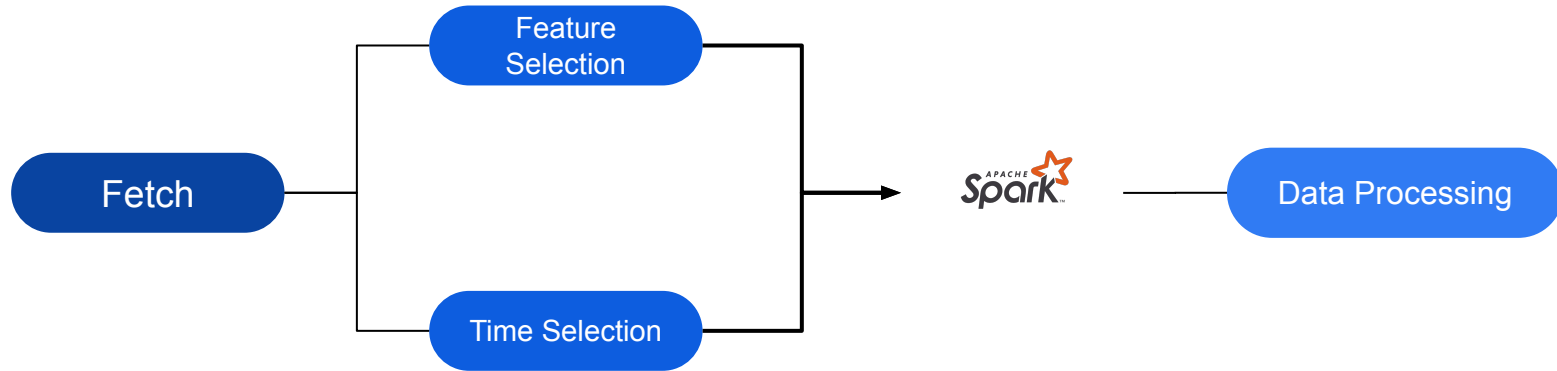
1. 13 rows containing various weather information;
2. Forecast for the next seven days, one hour apart;
3. Confused data types and data structures not suitable for streaming

```
[{'detailedForecast': '',  
  'endTime': '2022-05-06T00:00:00-04:00',  
  'icon': 'https://api.weather.gov/icons/land/night/bkn?size=small',  
  'isDaytime': False,  
  'name': '',  
  'number': 1,  
  'shortForecast': 'Mostly Cloudy',  
  'startTime': '2022-05-05T23:00:00-04:00',  
  'temperature': 62,  
  'temperatureTrend': None,  
  'temperatureUnit': 'F',  
  'windDirection': 'S',  
  'windSpeed': '6 mph'},  
 {'detailedForecast': '',  
  'endTime': '2022-05-06T01:00:00-04:00',  
  'icon': 'https://api.weather.gov/icons/land/night/bkn?size=small',  
  'isDaytime': False,  
  'name': '',  
  'number': 2,  
  'shortForecast': 'Mostly Cloudy',  
  'startTime': '2022-05-06T00:00:00-04:00',  
  'temperature': 59,  
  'temperatureTrend': None,
```



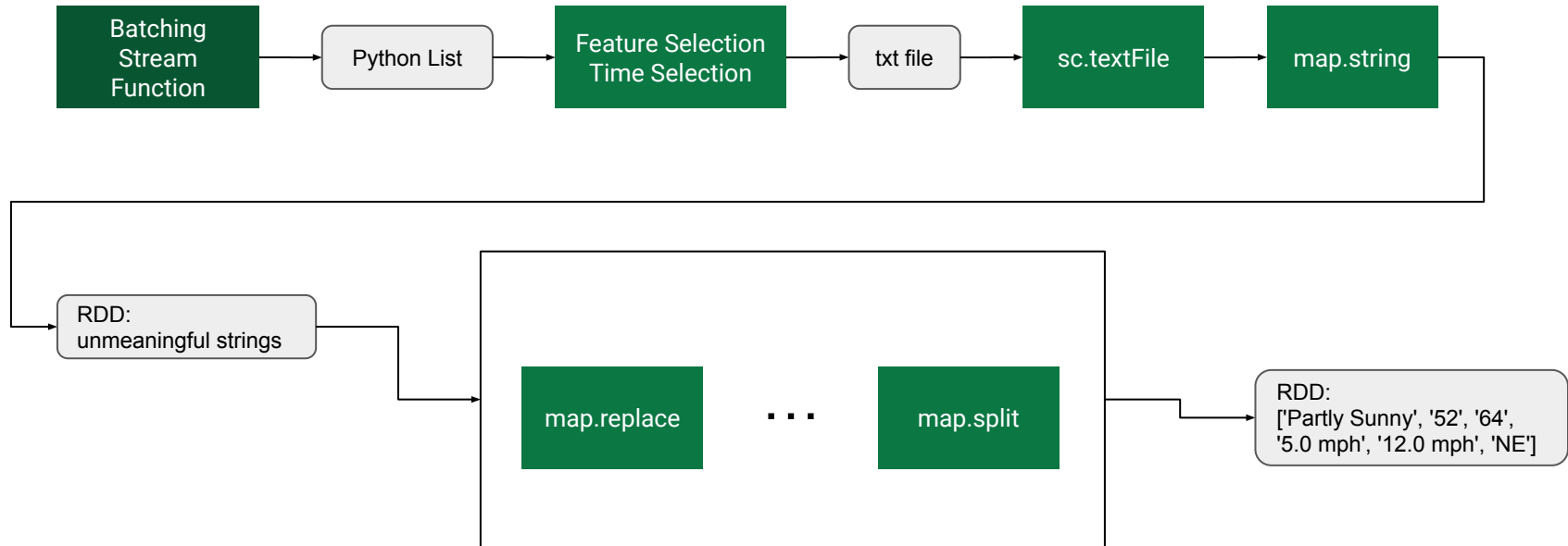
Data Processing

Data Preprocessing Procedure:



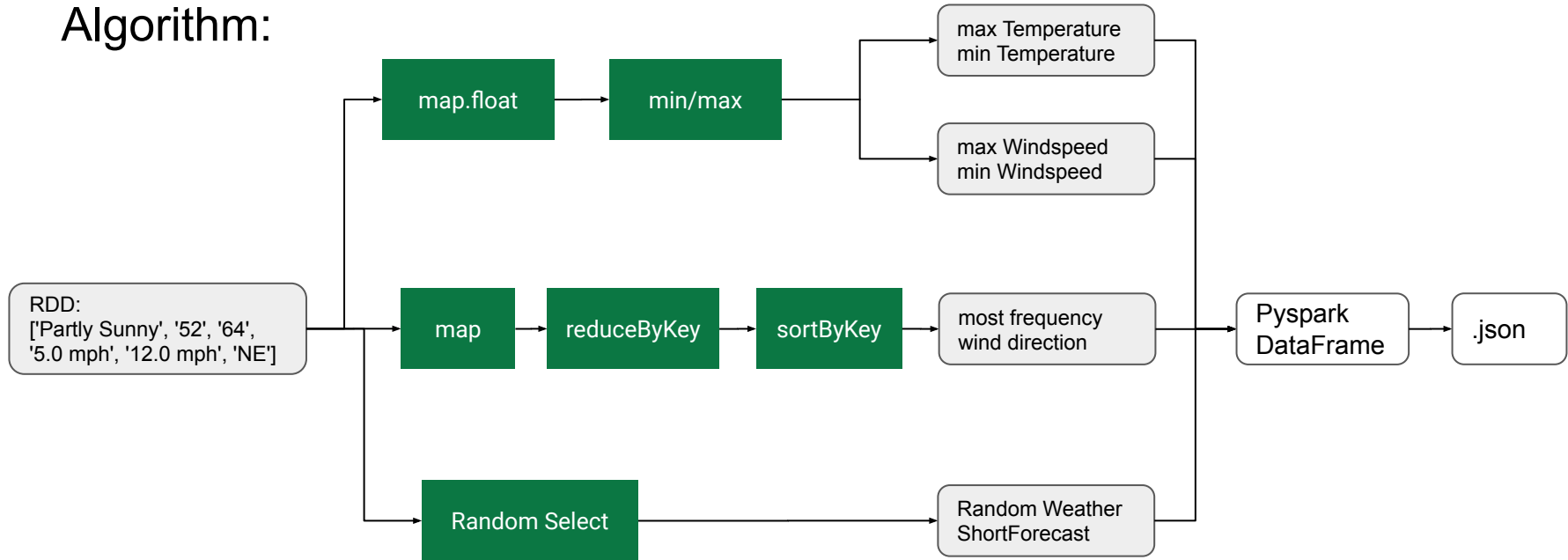
Data Processing

Data Processing:



Data Processing

Algorithm:



Data Processing

‘.json’ file modification:

We have extracted the ‘us-state.js’ and transform it to ‘.json’ format like this:

```
[{"type": "Feature", "id": "01", "properties": {"name": "Alabama", "density": 94.65}, "geometry": {"type": "Polygon",  
{"type": "Feature", "id": "04", "properties": {"name": "Arizona", "density": 57.05}, "geometry": {"type": "Polygon",  
{"type": "Feature", "id": "05", "properties": {"name": "Arkansas", "density": 56.43}, "geometry": {"type": "Polygon",  
{"type": "Feature", "id": "06", "properties": {"name": "California", "density": 241.7}, "geometry": {"type": "Polygo  
{"type": "Feature", "id": "08", "properties": {"name": "Colorado", "density": 49.33}, "geometry": {"type": "Polygon",  
{"type": "Feature", "id": "09", "properties": {"name": "Connecticut", "density": 739.1}, "geometry": {"type": "Polyg  
{"type": "Feature", "id": "10", "properties": {"name": "Delaware", "density": 464.3}, "geometry": {"type": "Polygon",  
{"type": "Feature", "id": "11", "properties": {"name": "District of Columbia", "density": 10065}, "geometry": {"typ  
{"type": "Feature", "id": "12", "properties": {"name": "Florida", "density": 353.4}, "geometry": {"type": "Polygon",  
{"type": "Feature", "id": "13", "properties": {"name": "Georgia", "density": 169.5}, "geometry": {"type": "Polygon",  
{"type": "Feature", "id": "15", "properties": {"name": "Hawaii", "density": 214.1}, "geometry": {"type": "MultiPolyg  
{"type": "Feature", "id": "16", "properties": {"name": "Idaho", "density": 19.15}, "geometry": {"type": "Polygon", "c
```

Data Processing

‘.json’ file modification:

Add the weather condition in a {key:value} format:

```
[{"Weather": "Mostly Clear", "MinTemperature": 67.0, "MaxTemperature": 89.0, "MinWindSpeed": 0.0},  
{"Weather": "Sunny", "MinTemperature": 66.0, "MaxTemperature": 92.0, "MinWindSpeed": 0.0, "MaxWin"},  
{"Weather": "Partly Sunny", "MinTemperature": 64.0, "MaxTemperature": 73.0, "MinWindSpeed": 5.0},  
{"Weather": "Sunny", "MinTemperature": 52.0, "MaxTemperature": 88.0, "MinWindSpeed": 3.0, "MaxWin"},  
{"Weather": "Partly Sunny", "MinTemperature": 39.0, "MaxTemperature": 48.0, "MinWindSpeed": 5.0},  
{"Weather": "Chance Rain Showers", "MinTemperature": 49.0, "MaxTemperature": 63.0, "MinWindSpee"},  
{"Weather": "Slight Chance Rain Showers", "MinTemperature": 55.0, "MaxTemperature": 71.0, "Mink"},  
{"Weather": "Scattered Showers And Thunderstorms", "MinTemperature": 56.0, "MaxTemperature": 7},  
{"Weather": "Areas Of Fog", "MinTemperature": 67.0, "MaxTemperature": 89.0, "MinWindSpeed": 0.0},  
{"Weather": "Mostly Sunny", "MinTemperature": 64.0, "MaxTemperature": 84.0, "MinWindSpeed": 0.0},
```

Data Processing

‘.json’ file modification:

Update json function:

```
#!/usr/bin/env python
# coding: utf-8
import json
import re

with open("weather_condition.json", "r", encoding="utf-8") as f:
    load_dict_wea = json.load(f)

states_list = ['Alabama', 'Arizona', 'Arkansas', 'California', 'Colorado', 'Co
weather_list = load_dict_wea
#weather_list.append(load_dict_wea)
print(weather_list)

with open("us-states.json", "r", encoding="utf-8") as f:
    load_dict = json.load(f)
    print(load_dict)

n = len(load_dict)
print(range(0,n))

for i in range(0,n):
    if states_list[i]==load_dict[i]['properties']['name']:
        load_dict[i]['properties'].update(weather_list[i])

with open("updated_weather.json", "w", encoding="utf-8") as f:
    json.dump(load_dict, f)
```

Transform ‘.json’ to ‘.js’:

```
with open('updated_weather.json', 'r') as file:
    res = []
    res = file.readlines()
with open('updated_weather.js', "w") as resfile:
    resfile.write('var testData2 = {"type":"FeatureCollection","features":')
    for r in res:
        resfile.write('\n' + r + ', '+ '\n')
    resfile.write('\n};')
```



- I. Problem Definition
- II. Data Processing
- III. Visualization**
- IV. Results Demo
- V. Future Work



Frontend Part



We use Django to build our web server and use Bootstrap&CSS to decorate our web

Frontend Part - django



urls.py

```
urlpatterns = [  
    path('', views.results),  
    path('<str:state_name>weather', views.state_weather)  
]
```

views.py

```
def state_weather(request, state_name):  
    context = {}  
    context['state_weather'] = {  
        "name": str(state_name)  
    }  
  
    return render(request, "weather.html", context)
```

./templates/index.html

```
location.href = `${t_s_name}weather`;
```


Frontend Part - Bootstrap

Headings

Example heading New

Example heading New

Example heading New

Example heading New

Example heading New

Example heading New

```
<h1>Example heading <span class="badge bg-secondary">New</span></h1>
<h2>Example heading <span class="badge bg-secondary">New</span></h2>
<h3>Example heading <span class="badge bg-secondary">New</span></h3>
<h4>Example heading <span class="badge bg-secondary">New</span></h4>
<h5>Example heading <span class="badge bg-secondary">New</span></h5>
<h6>Example heading <span class="badge bg-secondary">New</span></h6>
```

Copy



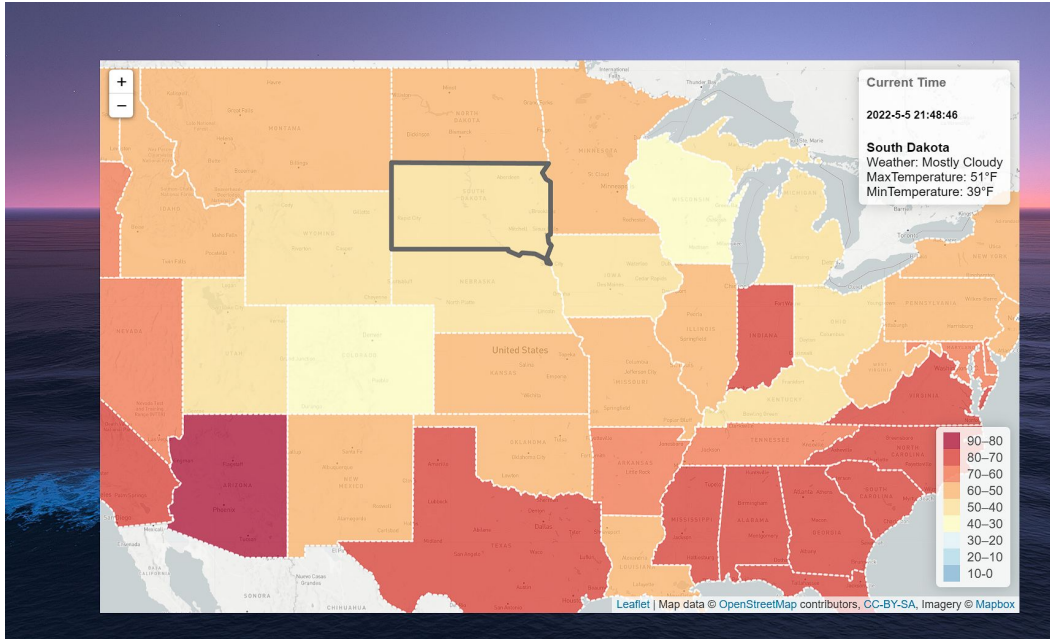
Frontend Part



an open-source JavaScript library
for mobile-friendly interactive maps

Leaflet is the leading open-source JavaScript library for mobile-friendly interactive maps. Weighing just about 39 KB of JS, it has all the mapping features most developers ever need. Using mapbox's api.

Frontend Part



GeoJSON data

```
{  
  "type": "Feature",  
  "properties": {  
    "name": "Alabama",  
    "density": 94.65  
  },  
  "geometry": ...  
  ...  
}
```

Frontend Part



Detail information will be showed if user click on the map.

These information include max&min temperature, max&min windspeed and wind direction.

Frontend Part

auto refresh data

```
auto_refresh = function() {  
    initialization();  
    layer_flag = 1;  
    setTimeout(auto_refresh, refresh_seconds * 200);  
    reload_js("{% static 'updated_weather.js' %}");  
}
```

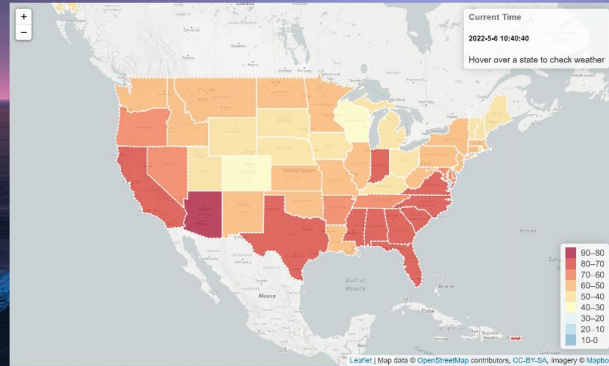
- I. Problem Definition
- II. Data Processing
- III. Visualization
- IV. Results Demo**
- V. Future Work



demo

Real Time Weather Condition Analysis of United States

Yuhang Wang, Kangrui Li, Linyang Han, Xiaohang He



- I. Problem Definition
- II. Data Processing
- III. Visualization
- IV. Results Demo
- V. Future Work**



Future Work

- Expand the dataset to cover weather data in every city
- Apply streaming optimization to improve the streaming process and process data more efficiently
- Try to find a better weather data source to implement this system

Reference

<https://pypi.org/project/noaa-sdk/>

<https://leafletjs.com/examples/choropleth/>

<https://spark.apache.org/docs/latest/rdd-programming-guide.html#rdd-operation>

S

Thank you for your listening!

