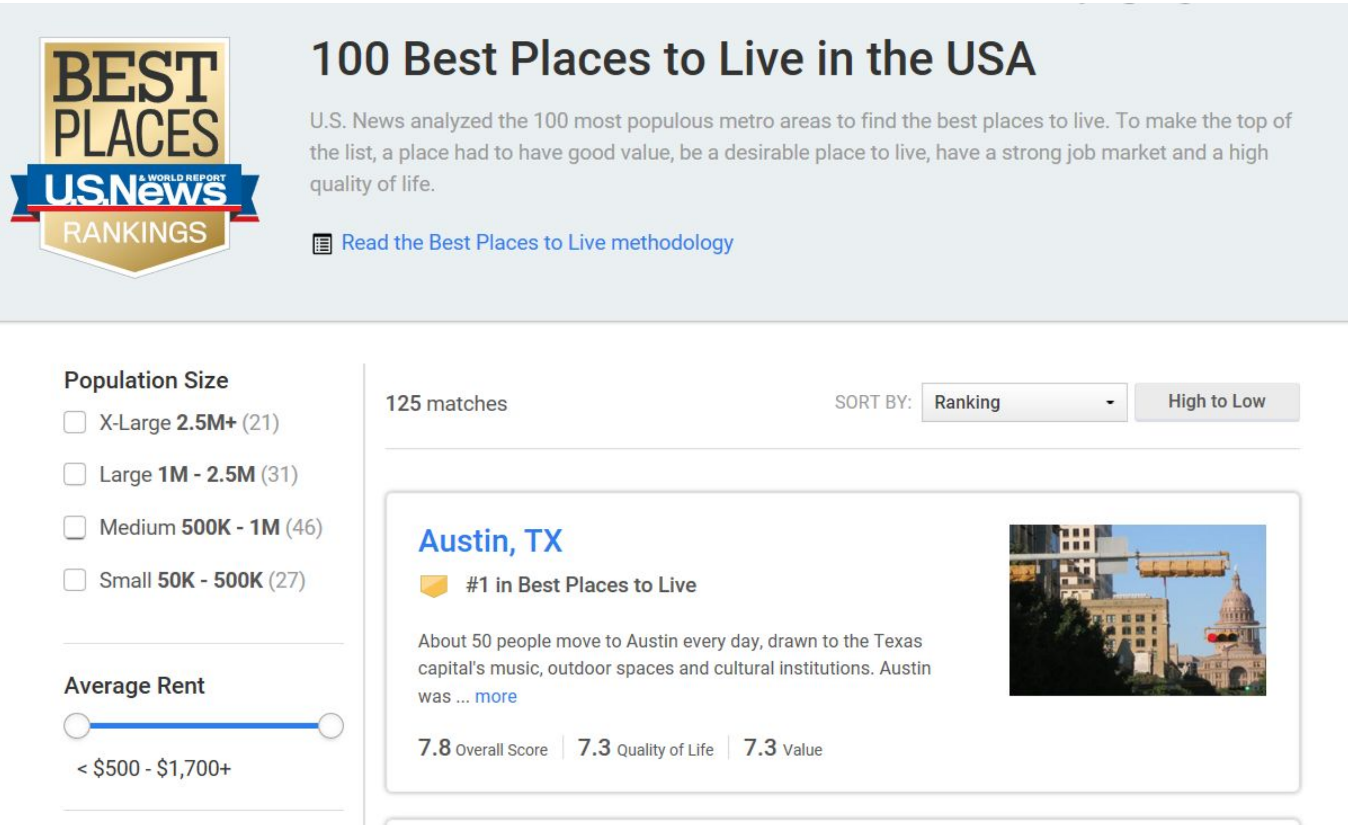
Which city would you like to live?

- City Fγndεγs

Xiangyu Zhang, Wendan Yan, Yiran Zhang, Zhuochen Han

Background



BEST PLACES U.S.News & WORLD REPORT RANKINGS

100 Best Places to Live in the USA

U.S. News analyzed the 100 most populous metro areas to find the best places to live. To make the top of the list, a place had to have good value, be a desirable place to live, have a strong job market and a high quality of life.

Read the Best Places to Live methodology

Population Size

- □ X-Large **2.5M+** (21)
- □ Large **1M - 2.5M** (31)
- □ Medium **500K - 1M** (46)
- □ Small **50K - 500K** (27)

Average Rent

< $500 - $1,700+

125 matches

SORT BY: Ranking High to Low

Austin, TX

#1 in Best Places to Live

About 50 people move to Austin every day, drawn to the Texas capital's music, outdoor spaces and cultural institutions. Austin was ... more

7.8 Overall Score | 7.3 Quality of Life | 7.3 Value

Denver, CO

#2 in Best Places to Live

ounded in the mid-1800s as a mining hub during the gold rush, enver has come a long way since its Wild West days. Over time, ts ... more

Now, let’s DIY

4 types of selected criteria

- Environment
- Human related
- Economy
- Entertainment

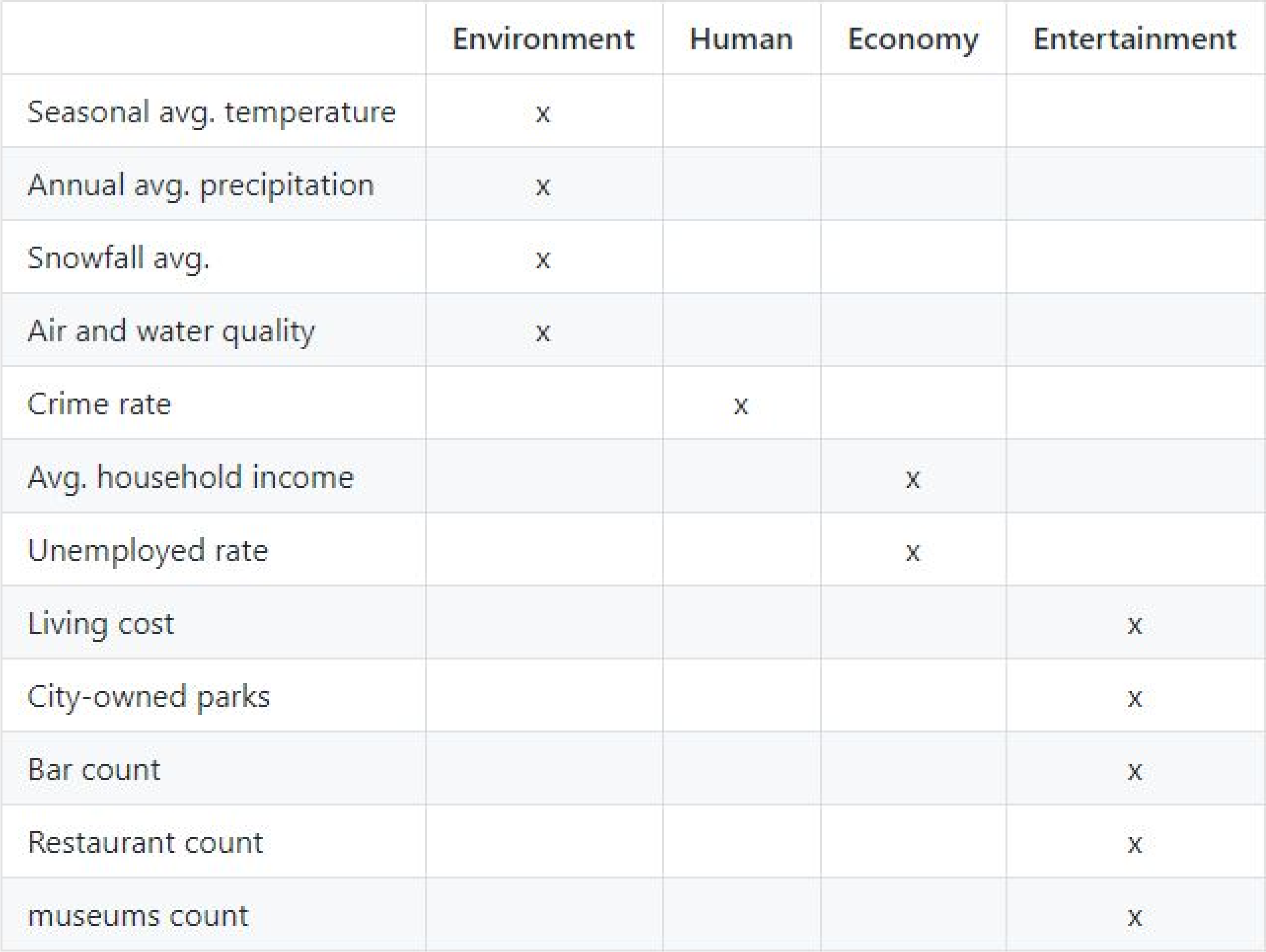
Data resource

1. Nature related data:
   - Multi-year climate: Monthly avg temperature, annual avg precipitation, snowfall avg https://www.infoplease.com/science-health/weather/climate-100-selected-us-cities
   - Pollution since 2000 https://www.kaggle.com/sogun3/uspollution/data
2. Human related data:
   - Crime rate: https://en.wikipedia.org/wiki/List_of_United_States_cities_by_crime_rate
   - Unemployment Rates for Metropolitan Areas: https://www.bls.gov/web/metro/laummtrk.htm#laummtrk.f.p
   - School education quality (avg SAT scores) https://www.privateschoolreview.com/average-sat-score-stats/national-data/towns
3. Economy Related:
   - Tax rates: https://taxfoundation.org/sales-tax-rates-major-cities-midyear-2017
4. Tertiary Industry
   - Restaurant price: https://www.numbeo.com/cost-of-living/country_result.jsp?country=United+States

Data used

4 types of criteria

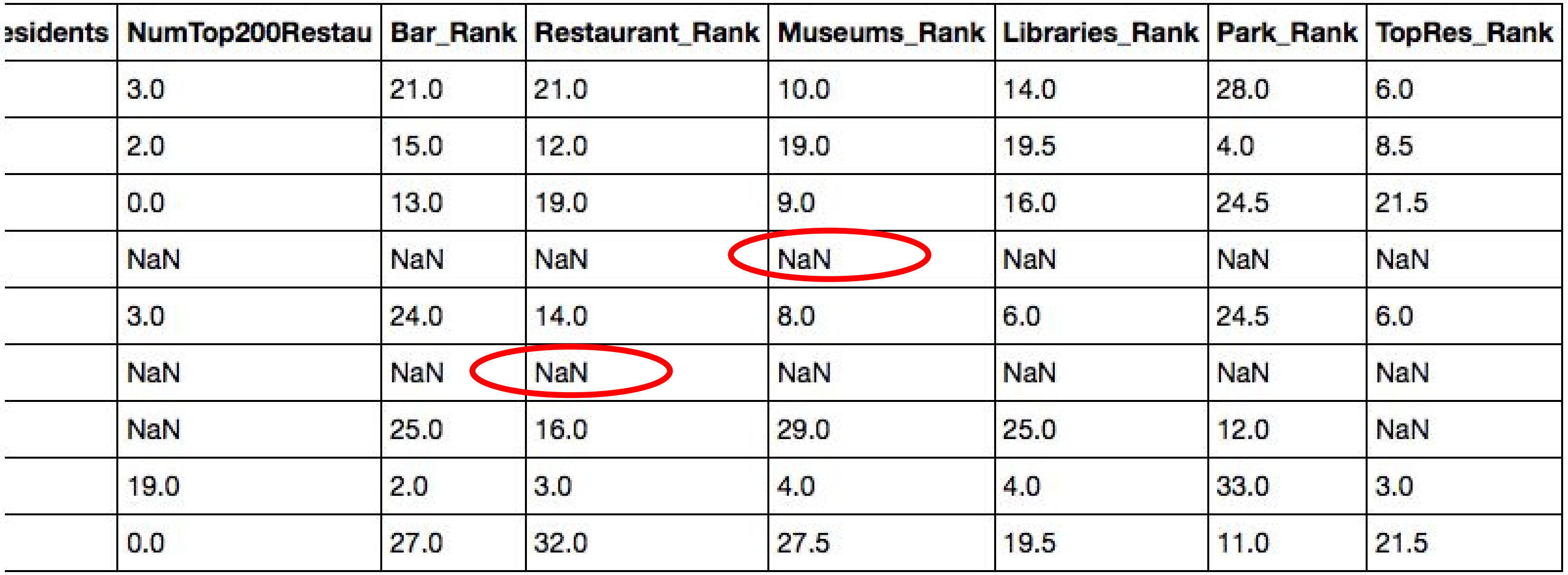
- Environment
- Human related
- Economy
- Entertainment



	Environment	Human	Economy	Entertainment
Seasonal avg. temperature	x			
Annual avg. precipitation	x			
Snowfall avg.	x			
Air and water quality	x			
Crime rate		x		
Avg. household income			x	
Unemployed rate			x	
Living cost				x
City-owned parks				x
Bar count				x
Restaurant count				x
museums count				x

Data limitation

- Data obtained fails to satisfy every city ('NaN exists')
- Data found from different year
- Limited data obtained based on the chosen category



esidents	NumTop200Restau	Bar_Rank	Restaurant_Rank	Museums_Rank	Libraries_Rank	Park_Rank	TopRes_Rank
	3.0	21.0	21.0	10.0	14.0	28.0	6.0
	2.0	15.0	12.0	19.0	19.5	4.0	8.5
	0.0	13.0	19.0	9.0	16.0	24.5	21.5
	NaN	NaN	NaN	NaN	NaN	NaN	NaN
	3.0	24.0	14.0	8.0	6.0	24.5	6.0
	NaN	NaN	NaN	NaN	NaN	NaN	NaN
	NaN	25.0	16.0	29.0	25.0	12.0	NaN
	19.0	2.0	3.0	4.0	4.0	33.0	3.0
	0.0	27.0	32.0	27.5	19.5	11.0	21.5

Use cases

Welcome Page with default research result

- Run setup.py file to prompt up the Welcome Page
- Present the general research results of the top cities of the four categories
- Present some important basic data with scatters

Users choose ranking criteria and assign weight to each criterion

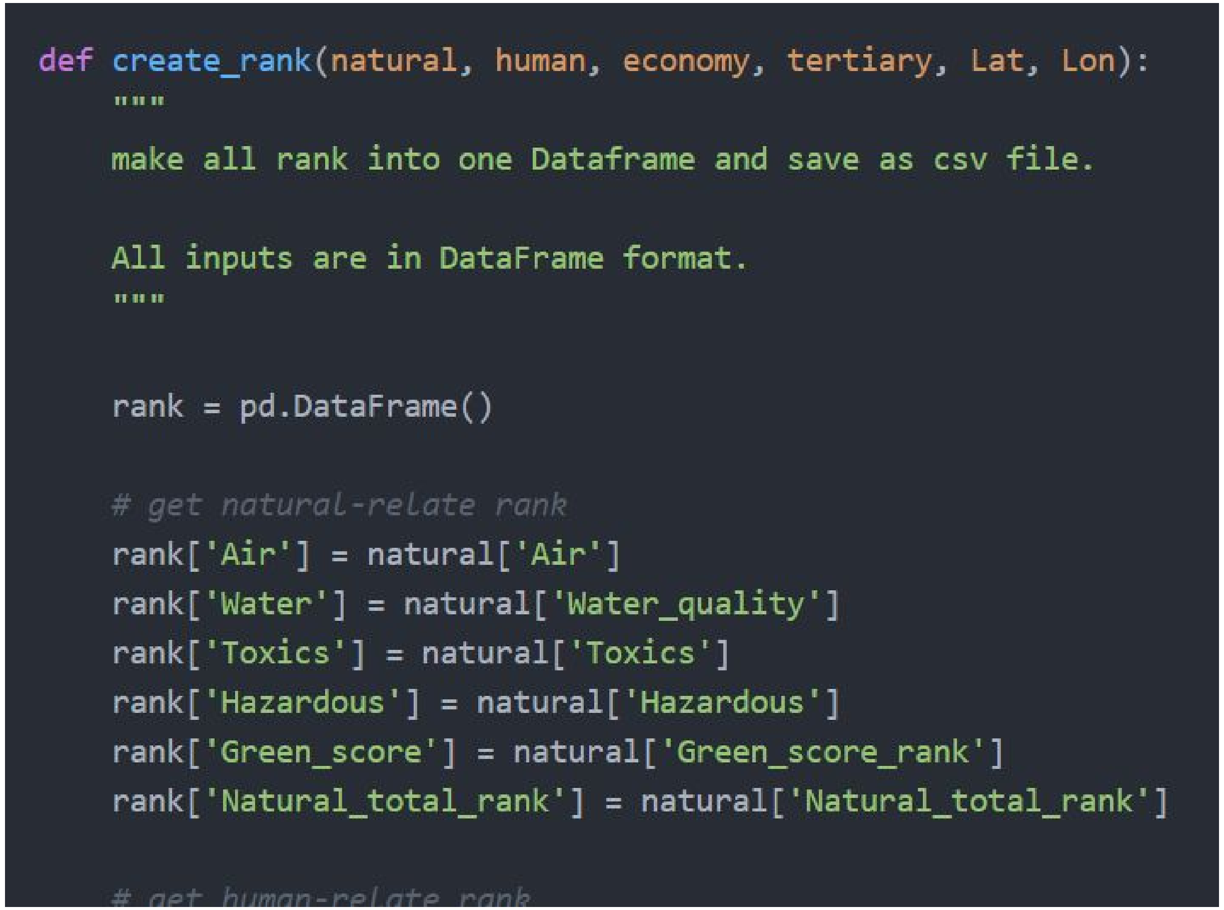
- From all the factors, users pick 5 top criteria they care about and rank them from 1 to 5
- Calculate the s
- Core of each city (*Score1* 5 +Score2 * 4 + Score3 * 3 + Score4 * 2 + Score5 * 1*)

Visualization ranking results on the US map

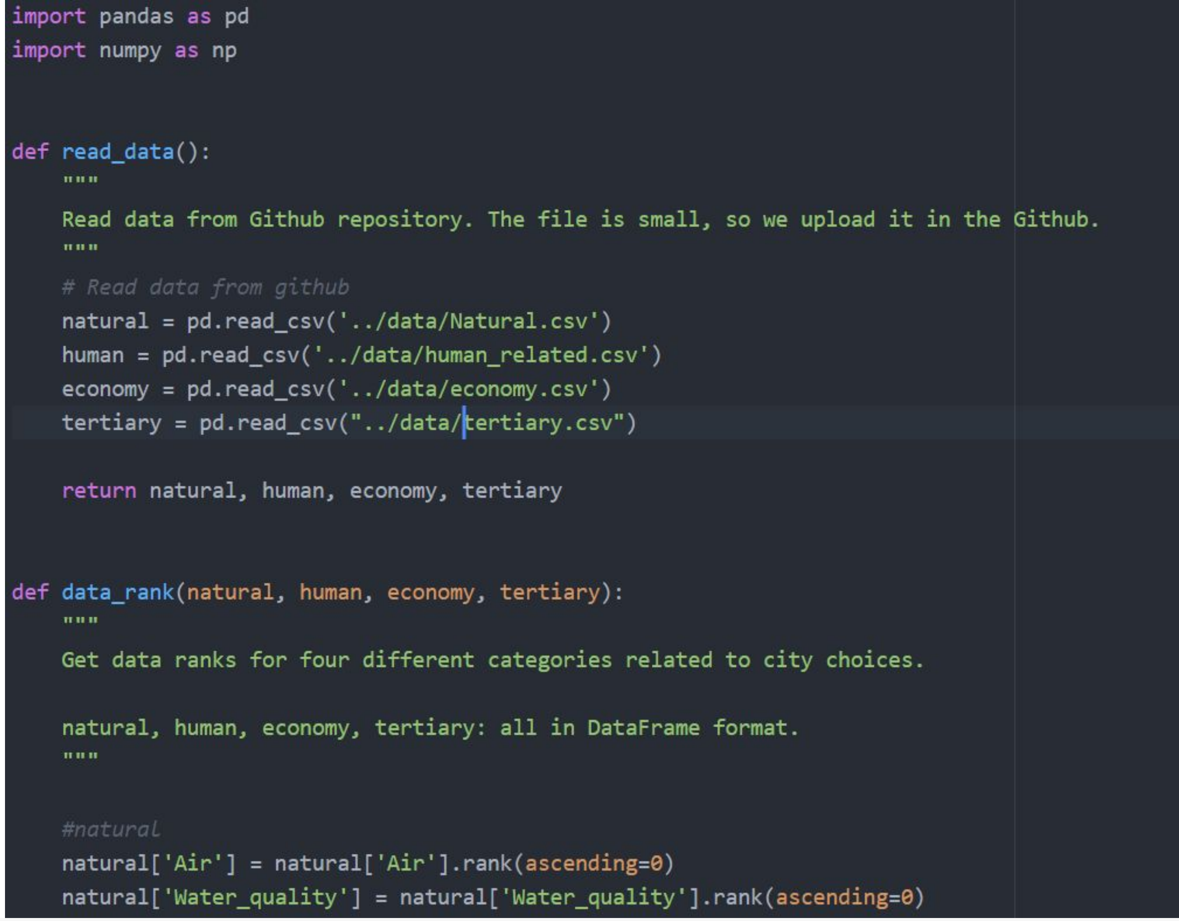
- Import a certain US map package showing all the city information
- Click on a city to show the rankings of factors user care about

Design

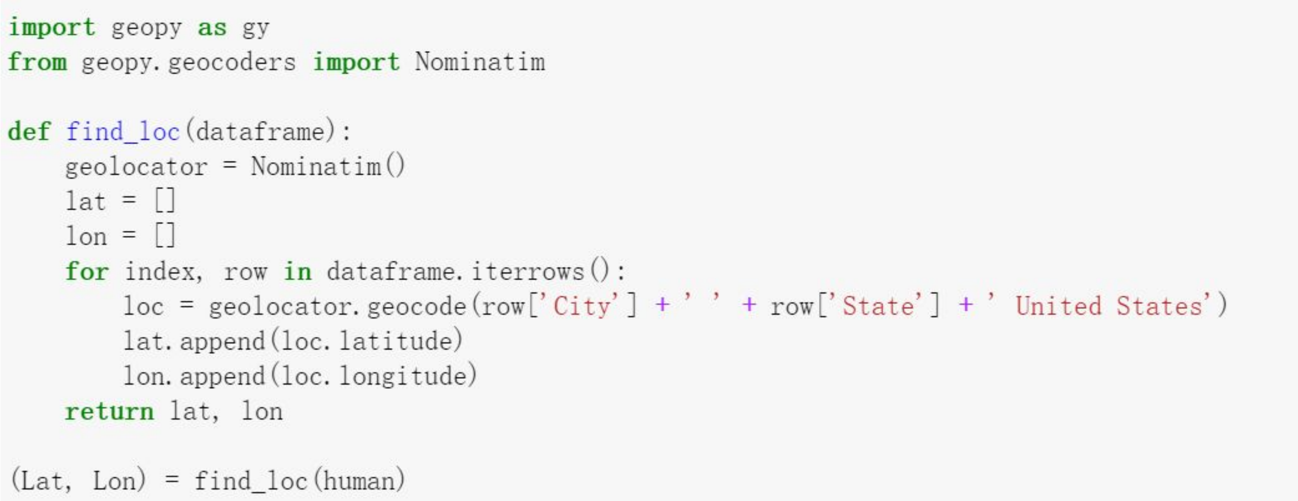
First work: Data processing



```python
def create_rank(natural, human, economy, tertiary, Lat, Lon):
    """
    make all rank into one Dataframe and save as csv file.

    All inputs are in DataFrame format.
    """

    rank = pd.DataFrame()

    # get natural-relate rank
    rank['Air'] = natural['Air']
    rank['Water'] = natural['Water_quality']
    rank['Toxics'] = natural['Toxics']
    rank['Hazardous'] = natural['Hazardous']
    rank['Green_score'] = natural['Green_score_rank']
    rank['Natural_total_rank'] = natural['Natural_total_rank']

    # get human-relate rank
```

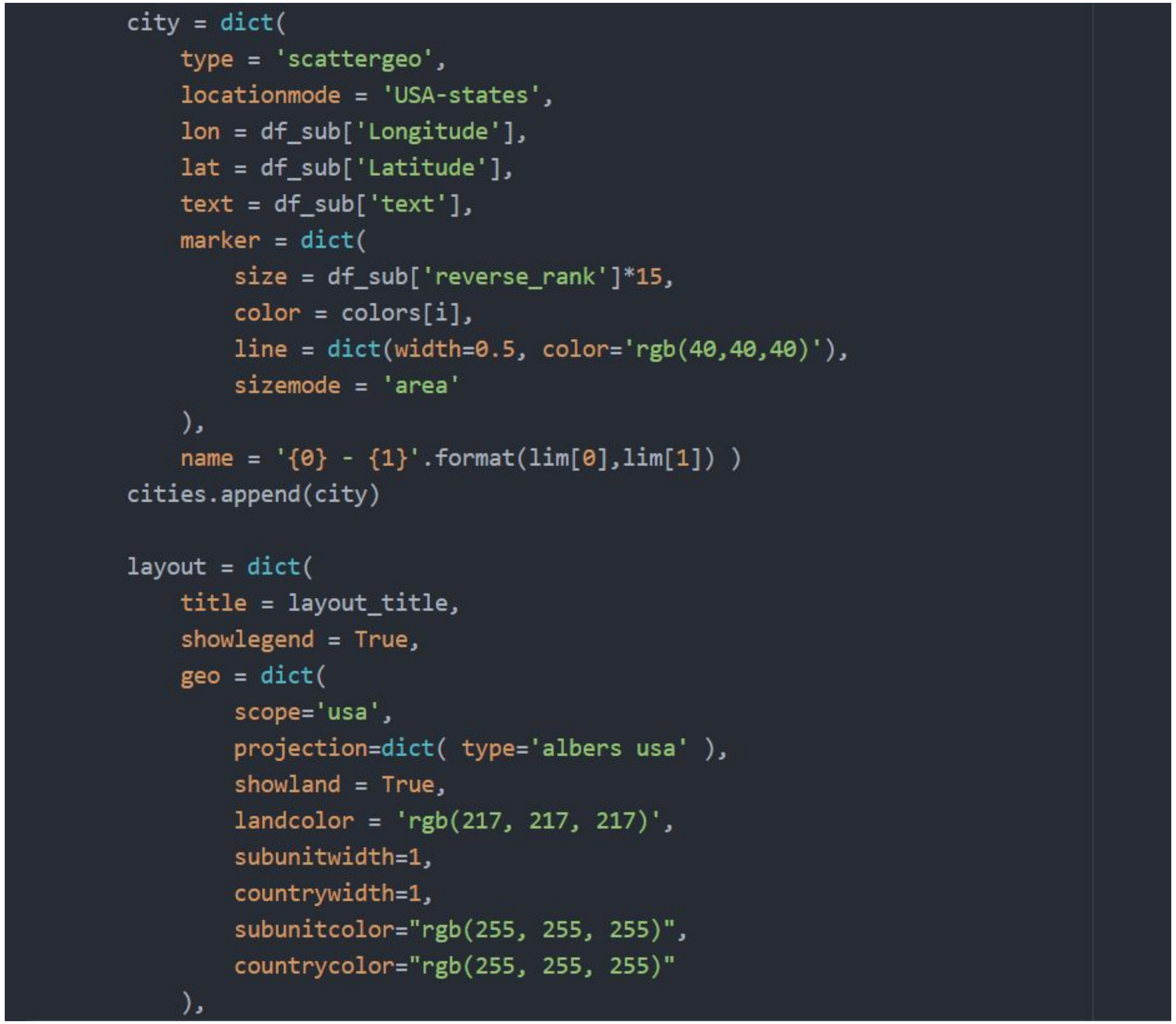


```python
import pandas as pd
import numpy as np


def read_data():
    """
    Read data from Github repository. The file is small, so we upload it in the Github.
    """
    # Read data from github
    natural = pd.read_csv('../data/Natural.csv')
    human = pd.read_csv('../data/human_related.csv')
    economy = pd.read_csv('../data/economy.csv')
    tertiary = pd.read_csv("../data/tertiary.csv")

    return natural, human, economy, tertiary


def data_rank(natural, human, economy, tertiary):
    """
    Get data ranks for four different categories related to city choices.

    natural, human, economy, tertiary: all in DataFrame format.
    """

    #natural
    natural['Air'] = natural['Air'].rank(ascending=0)
    natural['Water_quality'] = natural['Water_quality'].rank(ascending=0)
```

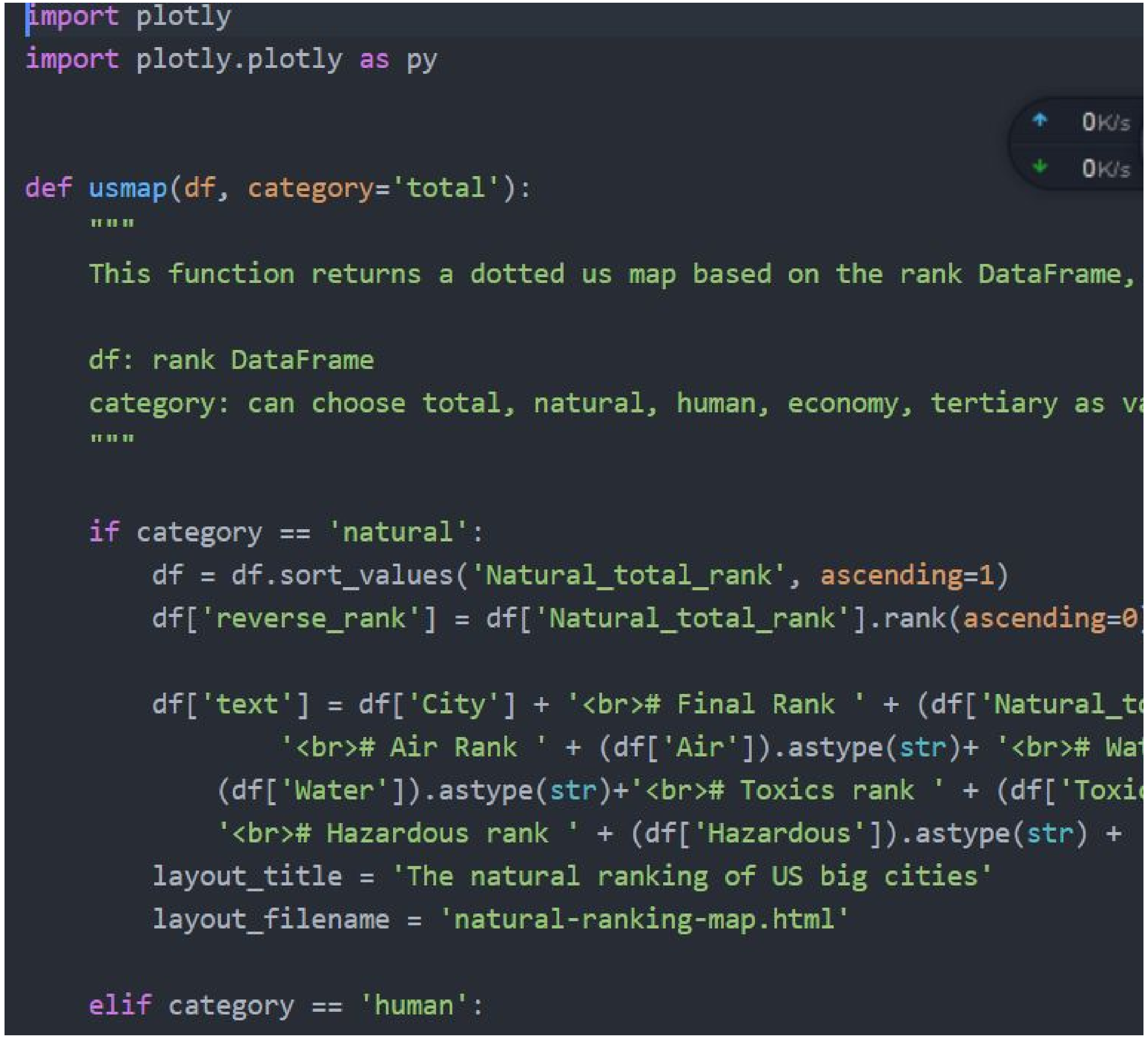


```python
import geopy as gy
from geopy.geocoders import Nominatim

def find_loc(dataframe):
    geolocator = Nominatim()
    lat = []
    lon = []
    for index, row in dataframe.iterrows():
        loc = geolocator.geocode(row['City'] + ' ' + row['State'] + ' United States')
        lat.append(loc.latitude)
        lon.append(loc.longitude)
    return lat, lon

(Lat, Lon) = find_loc(human)
```

Next: Default plot

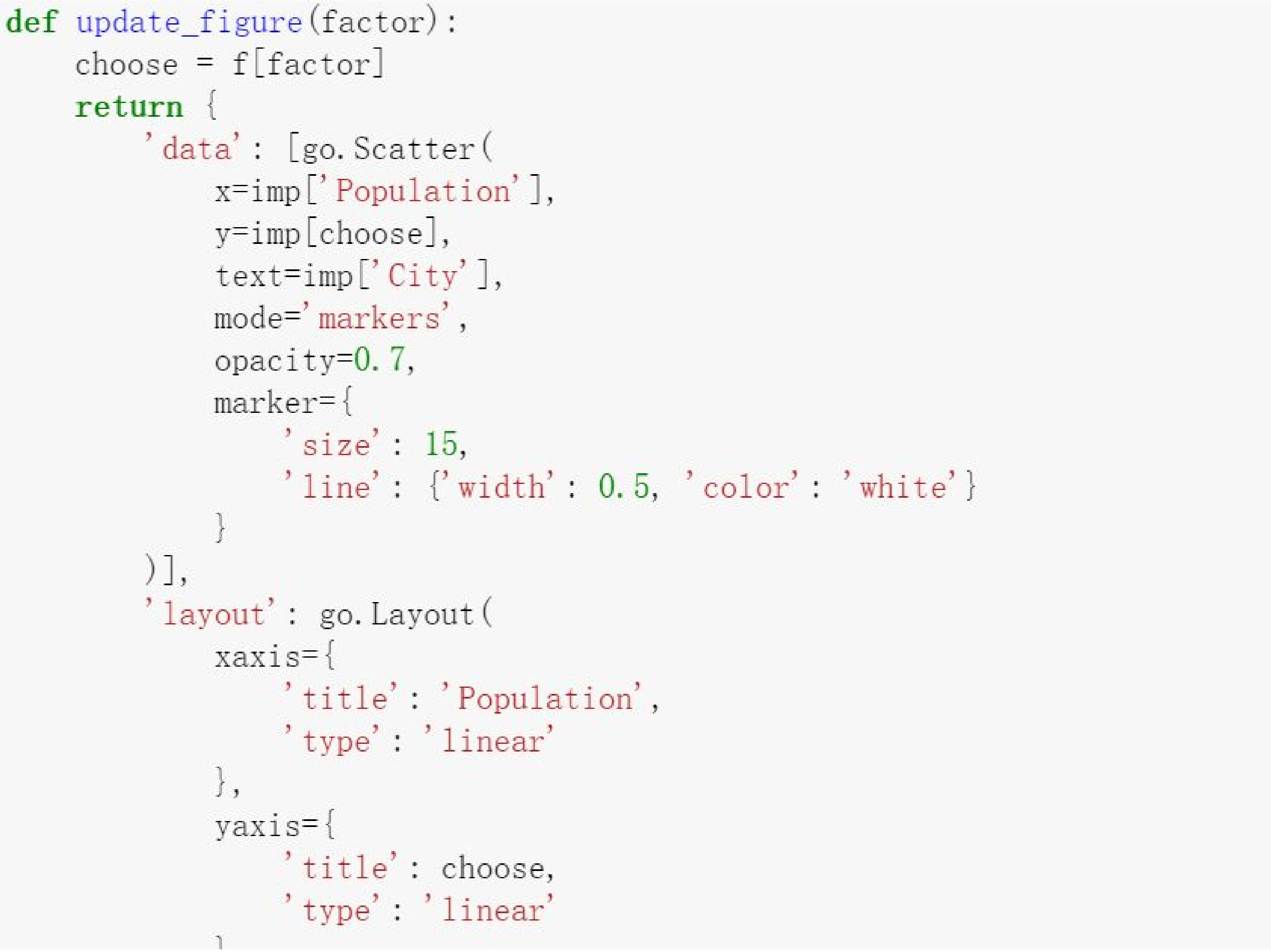


```
city = dict(
    type = 'scattergeo',
    locationmode = 'USA-states',
    lon = df_sub['Longitude'],
    lat = df_sub['Latitude'],
    text = df_sub['text'],
    marker = dict(
        size = df_sub['reverse_rank']*15,
        color = colors[i],
        line = dict(width=0.5, color='rgb(40,40,40)'),
        sizemode = 'area'
    ),
    name = '{0} - {1}'.format(lim[0],lim[1]) )
cities.append(city)

layout = dict(
    title = layout_title,
    showlegend = True,
    geo = dict(
        scope='usa',
        projection=dict( type='albers usa' ),
        showland = True,
        landcolor = 'rgb(217, 217, 217)',
        subunitwidth=1,
        countrywidth=1,
        subunitcolor="rgb(255, 255, 255)",
        countrycolor="rgb(255, 255, 255)"
    ),
```



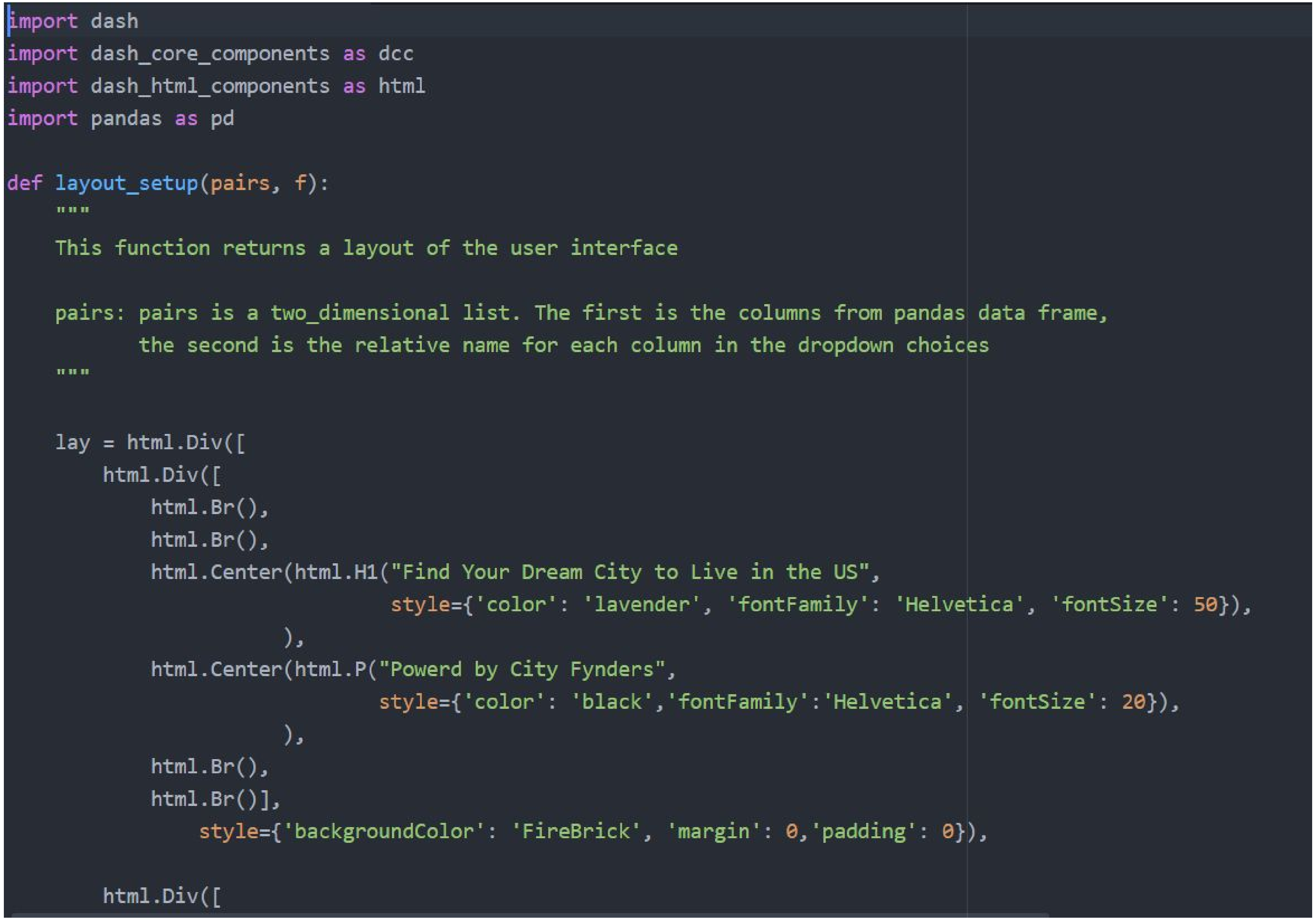
```
import plotly
import plotly.plotly as py


def usmap(df, category='total'):
    """
    This function returns a dotted us map based on the rank DataFrame,

    df: rank DataFrame
    category: can choose total, natural, human, economy, tertiary as va
    """

    if category == 'natural':
        df = df.sort_values('Natural_total_rank', ascending=1)
        df['reverse_rank'] = df['Natural_total_rank'].rank(ascending=0

        df['text'] = df['City'] + '<br># Final Rank ' + (df['Natural_to
                '<br># Air Rank ' + (df['Air']).astype(str)+ '<br># Wat
            (df['Water']).astype(str)+'<br># Toxics rank ' + (df['Toxic
            '<br># Hazardous rank ' + (df['Hazardous']).astype(str) +
        layout_title = 'The natural ranking of US big cities'
        layout_filename = 'natural-ranking-map.html'

    elif category == 'human':
```

Important basic information on scatter graph



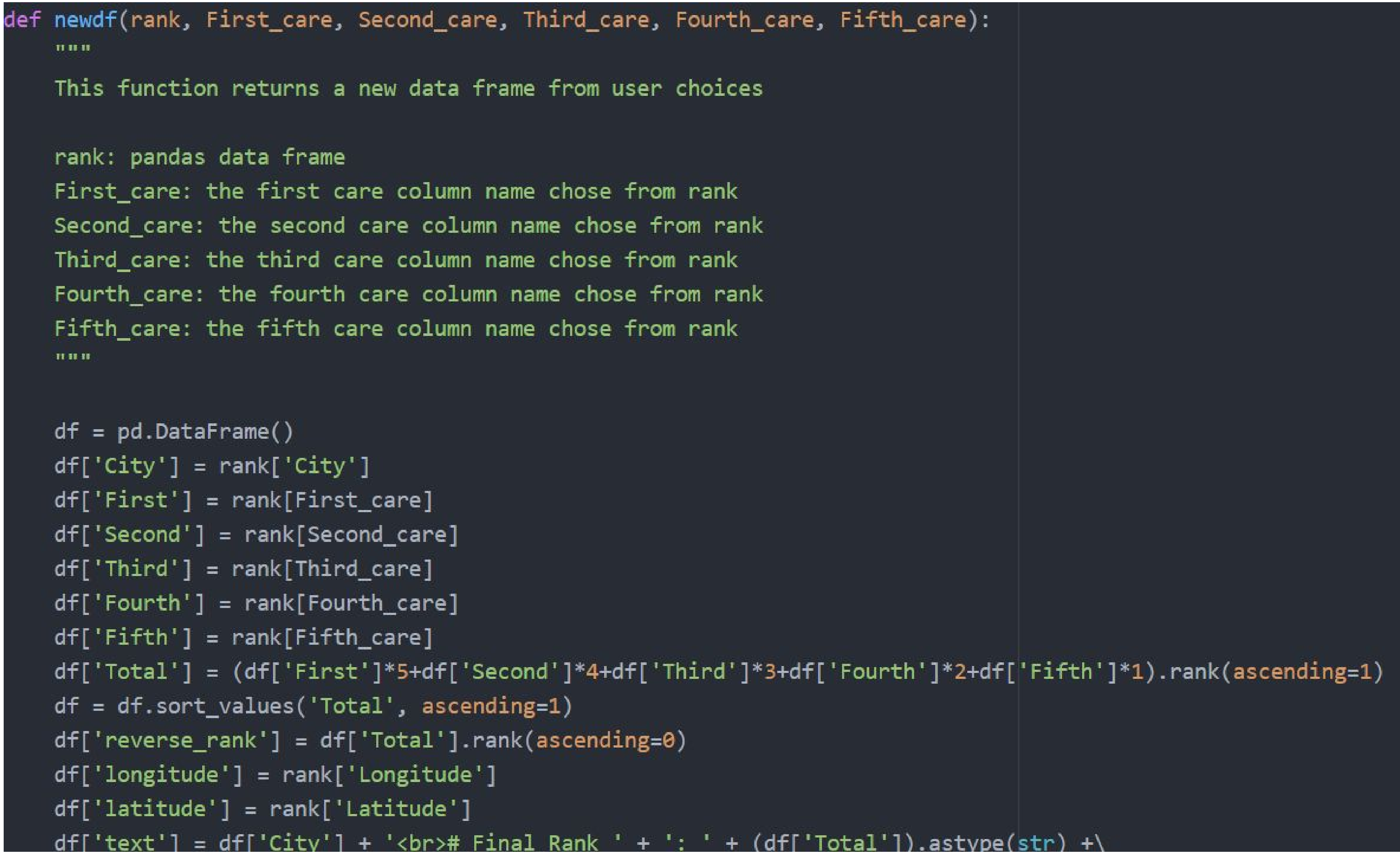
```
def update_figure(factor):
    choose = f[factor]
    return {
        'data': [go.Scatter(
            x=imp['Population'],
            y=imp[choose],
            text=imp['City'],
            mode='markers',
            opacity=0.7,
            marker={
                'size': 15,
                'line': {'width': 0.5, 'color': 'white'}
            }
        )],
        'layout': go.Layout(
            xaxis={
                'title': 'Population',
                'type': 'linear'
            },
            yaxis={
                'title': choose,
                'type': 'linear'
            }
```

Then: html format layout set up



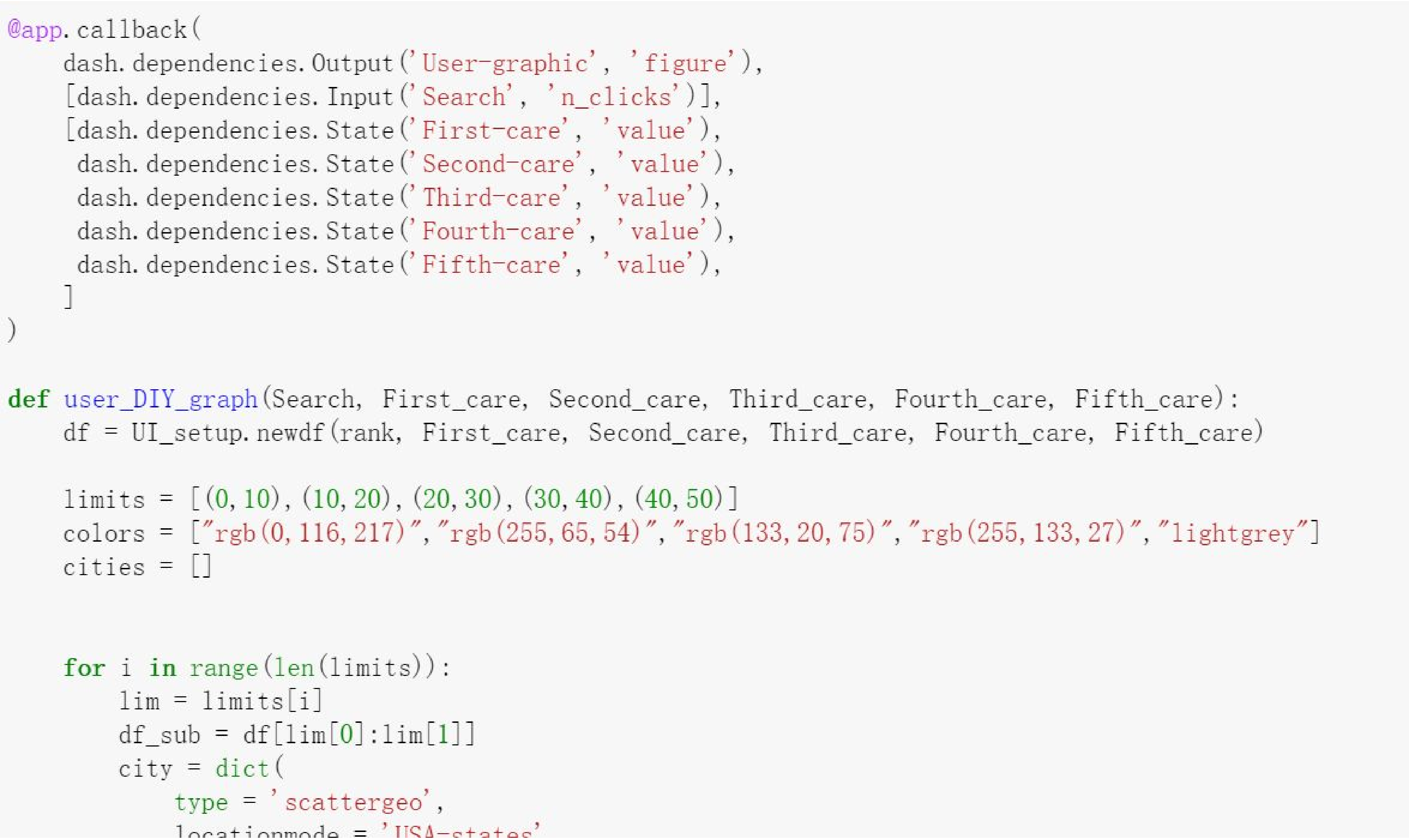
```
import dash
import dash_core_components as dcc
import dash_html_components as html
import pandas as pd

def layout_setup(pairs, f):
    """
    This function returns a layout of the user interface

    pairs: pairs is a two_dimensional list. The first is the columns from pandas data frame,
           the second is the relative name for each column in the dropdown choices
    """

    lay = html.Div([
        html.Div([
            html.Br(),
            html.Br(),
            html.Center(html.H1("Find Your Dream City to Live in the US",
                                style={'color': 'lavender', 'fontFamily': 'Helvetica', 'fontSize': 50}),
                       ),
            html.Center(html.P("Powerd by City Fynders",
                               style={'color': 'black','fontFamily':'Helvetica', 'fontSize': 20}),
                       ),
            html.Br(),
            html.Br()],
                style={'backgroundColor': 'FireBrick', 'margin': 0,'padding': 0}),

        html.Div([
```

Criteria calculation



```python
def newdf(rank, First_care, Second_care, Third_care, Fourth_care, Fifth_care):
    """
    This function returns a new data frame from user choices

    rank: pandas data frame
    First_care: the first care column name chose from rank
    Second_care: the second care column name chose from rank
    Third_care: the third care column name chose from rank
    Fourth_care: the fourth care column name chose from rank
    Fifth_care: the fifth care column name chose from rank
    """

    df = pd.DataFrame()
    df['City'] = rank['City']
    df['First'] = rank[First_care]
    df['Second'] = rank[Second_care]
    df['Third'] = rank[Third_care]
    df['Fourth'] = rank[Fourth_care]
    df['Fifth'] = rank[Fifth_care]
    df['Total'] = (df['First']*5+df['Second']*4+df['Third']*3+df['Fourth']*2+df['Fifth']*1).rank(ascending=1)
    df = df.sort_values('Total', ascending=1)
    df['reverse_rank'] = df['Total'].rank(ascending=0)
    df['longitude'] = rank['Longitude']
    df['latitude'] = rank['Latitude']
    df['text'] = df['City'] + '<br># Final Rank ' + ': ' + (df['Total']).astype(str) +\
```

Callback based on user choices

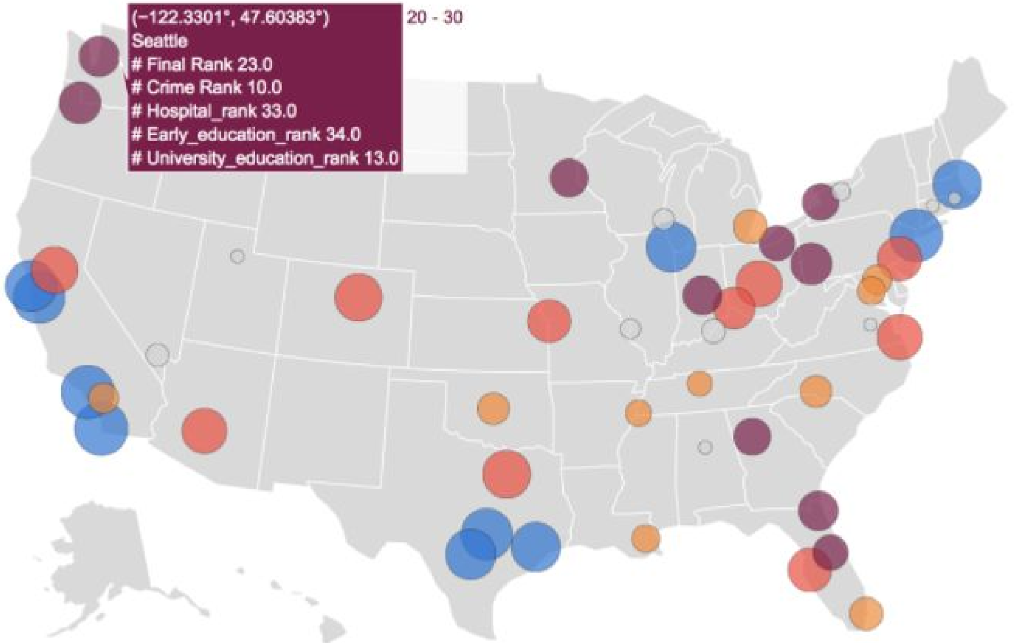


```
@app.callback(
    dash.dependencies.Output('User-graphic', 'figure'),
    [dash.dependencies.Input('Search', 'n_clicks')],
    [dash.dependencies.State('First-care', 'value'),
     dash.dependencies.State('Second-care', 'value'),
     dash.dependencies.State('Third-care', 'value'),
     dash.dependencies.State('Fourth-care', 'value'),
     dash.dependencies.State('Fifth-care', 'value'),
    ]
)

def user_DIY_graph(Search, First_care, Second_care, Third_care, Fourth_care, Fifth_care):
    df = UI_setup.newdf(rank, First_care, Second_care, Third_care, Fourth_care, Fifth_care)

    limits = [(0,10),(10,20),(20,30),(30,40),(40,50)]
    colors = ["rgb(0,116,217)","rgb(255,65,54)","rgb(133,20,75)","rgb(255,133,27)","lightgrey"]
    cities = []


    for i in range(len(limits)):
        lim = limits[i]
        df_sub = df[lim[0]:lim[1]]
        city = dict(
            type = 'scattergeo',
            locationmode = 'USA-states'
```

Find Your Dream City to Live in the US

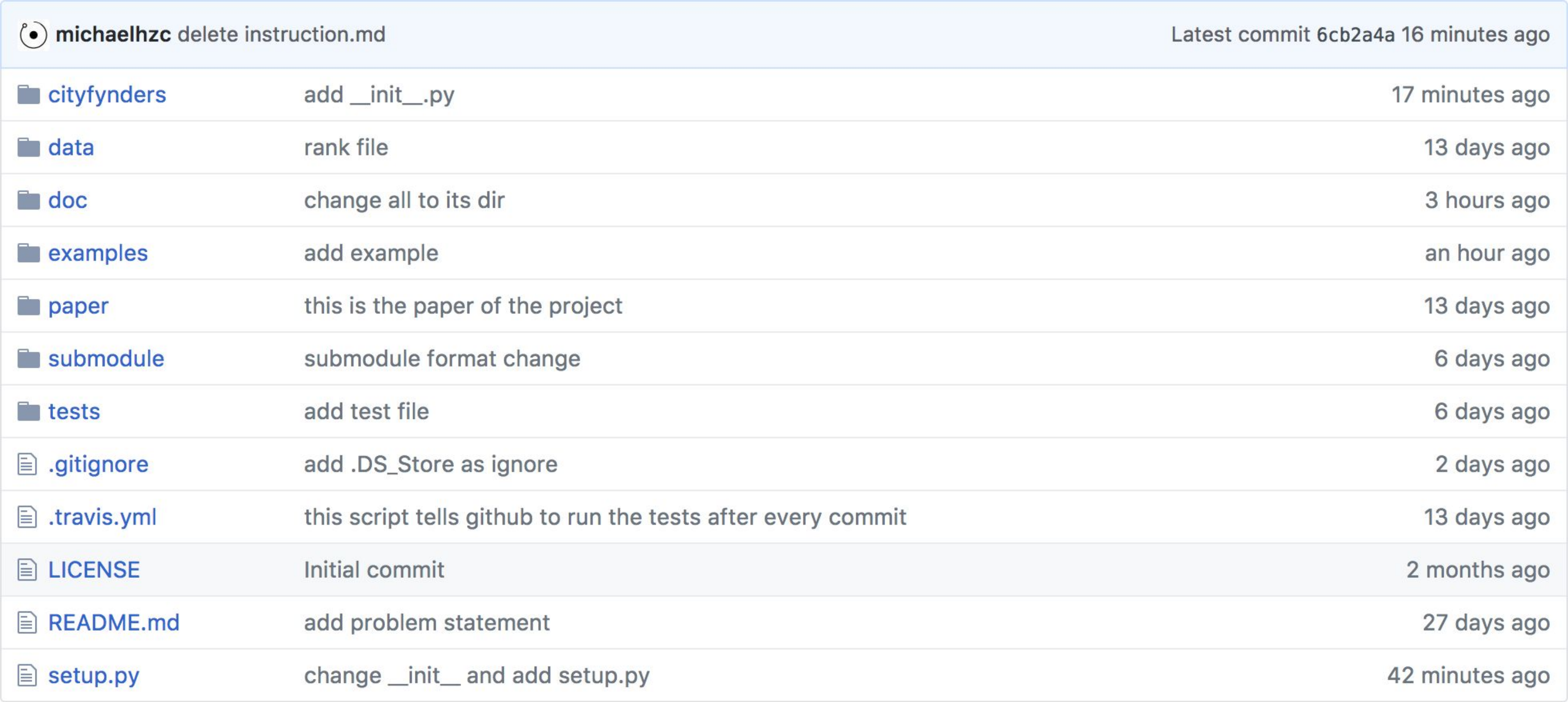
Powerd by City Fynders

The human related ranking of US big cities


(−122.3301°, 47.60383°)
Seattle
Final Rank 23.0
Crime Rank 10.0
Hospital_rank 33.0
Early_education_rank 34.0
University_education_rank 13.0

Project Structure

https://github.com/UWSEDS-aut17/uwseds-group-city-fynders



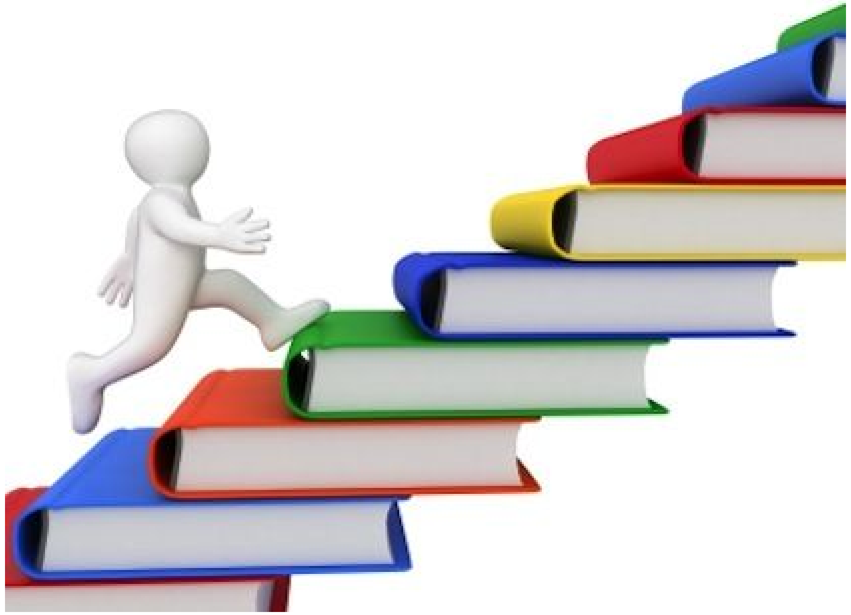
michaelhzc delete instruction.md Latest commit 6cb2a4a 16 minutes ago

cityfynders	add __init__.py	17 minutes ago
data	rank file	13 days ago
doc	change all to its dir	3 hours ago
examples	add example	an hour ago
paper	this is the paper of the project	13 days ago
submodule	submodule format change	6 days ago
tests	add test file	6 days ago
.gitignore	add .DS_Store as ignore	2 days ago
.travis.yml	this script tells github to run the tests after every commit	13 days ago
LICENSE	Initial commit	2 months ago
README.md	add problem statement	27 days ago
setup.py	change __init__ and add setup.py	42 minutes ago

Lessons Learned

What we learned?

- Basic recognition of Python
- GitHub
- Version control
- Programming styles
- Visualization in Python
- Tests
- Software design
- Basic machine learning



Future work

- Set up a server displaying the webpage only
- Beautify and optimize the website page
- Allow users to update the data source
- More data analysis
- Application of ML to predict the changing trend of the data

Thank
YOU

Q&A