EXERCISE 7

Cartographic Visualization

Colab Link

https://colab.research.google.com/drive/1WKXOKlk68nhwKmUNNFWdLYDnH8Xb8F\_-?usp=sharing

Aim:

To perform cartographic visualization for multiple datasets involving various countries of the world

# INSTALLATION OF GEOPANDAS

#install geopython libraries

!apt install gdal-bin python-gdal python3-gdal

#install python3-rtree - Geopandas requirement

!apt install python3-rtree

#install geopandas

!pip install git+git://github.com/geopandas/geopandas.git

#install descartes - Geopandas requirement

!pip install descartes

# MOUNTING MAP

from google.colab import drive

drive.mount('/content/drive')

import os

import sys

import os

sys.path.append('/usr/local/lib/python3.8/site-packages')

os.environ["GMT\_LIBRARY\_PATH"]="/usr/local/lib"

!wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86\_64.sh

!chmod +x Miniconda3-latest-Linux-x86\_64.sh

!bash ./Miniconda3-latest-Linux-x86\_64.sh -bfp /usr/local

!conda update conda -y -q

!conda config --prepend channels conda-forge

!conda install -q -y --prefix /usr/local python=3.8 pygmt

import sys

import os

sys.path.append('/usr/local/lib/python3.8/site-packages')

os.environ["GMT\_LIBRARY\_PATH"]="/usr/local/lib"

!pip install geopandas

# Install packages to colab environment

!sudo apt-get update && apt-get install -y libspatialindex-dev

!pip install rtree

!pip install geopandas

import pandas as pd

import altair as alt

from vega\_datasets import data

world = data.world\_110m.url

world

world\_topo = data.world\_110m()

world\_topo.keys()

world\_topo['objects'].keys()



OBSERVATION:

Wold map is executed.

alt.Chart(alt.topo\_feature(world, 'countries')).mark\_geoshape(

    fill='#2a1d0c', stroke='#706545', strokeWidth=1

).project(

    type='mercator'

)



OBSERVATION:

Topology of countries is shown.

map = alt.layer(

    # use the sphere of the Earth as the base layer

    alt.Chart({'sphere': True}).mark\_geoshape(

        fill='#e6f3ff'

    ),

    # add a graticule for geographic reference lines

    alt.Chart({'graticule': True}).mark\_geoshape(

        stroke='#ffffff', strokeWidth=1

    ),

    # and then the countries of the world

    alt.Chart(alt.topo\_feature(world, 'countries')).mark\_geoshape(

        fill='#2a1d0c', stroke='#706545', strokeWidth=0.5

    )

).properties(

    width=600,

    height=400

)

map = alt.layer(

    # use the sphere of the Earth as the base layer

    alt.Chart({'sphere': True}).mark\_geoshape(

        fill='#e6f3ff'

    ),

    # add a graticule for geographic reference lines

    alt.Chart({'graticule': True}).mark\_geoshape(

        stroke='#ffffff', strokeWidth=1

    ),

    # and then the countries of the world

    alt.Chart(alt.topo\_feature(world, 'countries')).mark\_geoshape(

        fill='#2a1d0c', stroke='#706545', strokeWidth=0.5

    )

).properties(

    width=600,

    height=400

)

zipcodes = data.zipcodes.url

zipcodes

alt.Chart(zipcodes).mark\_square(

    size=1, opacity=1

).encode(

    longitude='longitude:Q', # apply the field named 'longitude' to the longitude channel

    latitude='latitude:Q'    # apply the field named 'latitude' to the latitude channel

).project(

    type='albersUsa'

).properties(

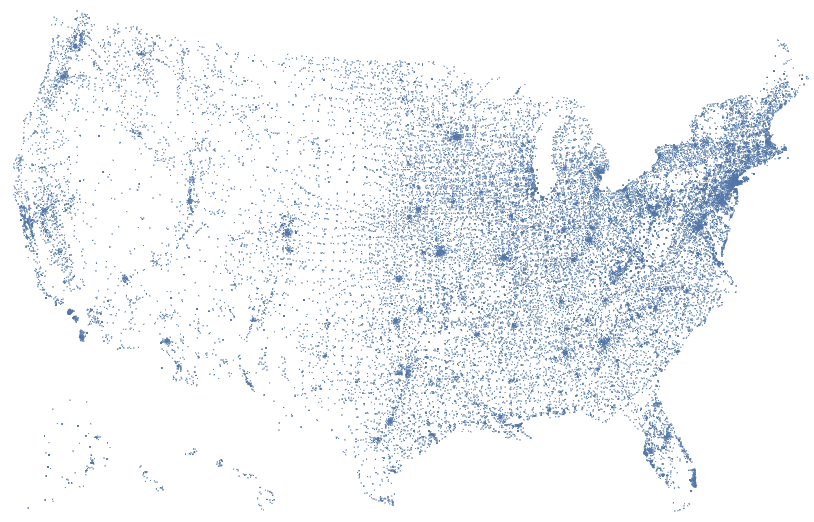
    width=900,

    height=500

).configure\_view(

    stroke=None

)



OBSERVATION:

Strokes of map is given using Altair.

alt.Chart(zipcodes).transform\_calculate(

    digit='datum.zip\_code[0]'

).mark\_square(

    size=2, opacity=1

).encode(

    longitude='longitude:Q',

    latitude='latitude:Q',

    color='digit:N'

).project(

    type='albersUsa'

).properties(

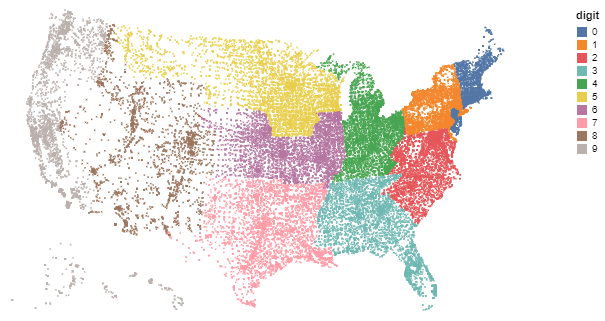
    width=600,

    height=300

).configure\_view(

    stroke=None

)



OBSERVATION:

Map is encoded using colours.

alt.Chart(zipcodes).transform\_filter(

    '-150 < datum.longitude && 22 < datum.latitude && datum.latitude < 55'

).transform\_calculate(

    digit='datum.zip\_code[0]'

).mark\_line(

    strokeWidth=0.5

).encode(

    longitude='longitude:Q',

    latitude='latitude:Q',

    color='digit:N',

    order='zip\_code:O'

).project(

    type='albersUsa'

).properties(

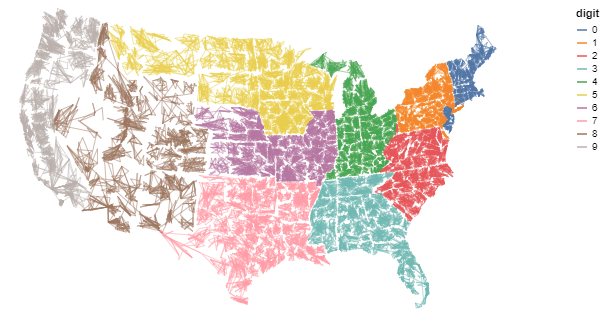
    width=600,

    height=300

).configure\_view(

    stroke=None

)



OBSERVATION:

Map is encoded using line marks.

usa = data.us\_10m.url

usa

airports = data.airports.url

airports

flights = data.flights\_airport.url

flights

alt.layer(

    alt.Chart(alt.topo\_feature(usa, 'states')).mark\_geoshape(

        fill='#ddd', stroke='#fff', strokeWidth=1

    ),

    alt.Chart(airports).mark\_circle(size=9).encode(

        latitude='latitude:Q',

        longitude='longitude:Q',

        tooltip='iata:N'

    )

).project(

    type='albersUsa'

).properties(

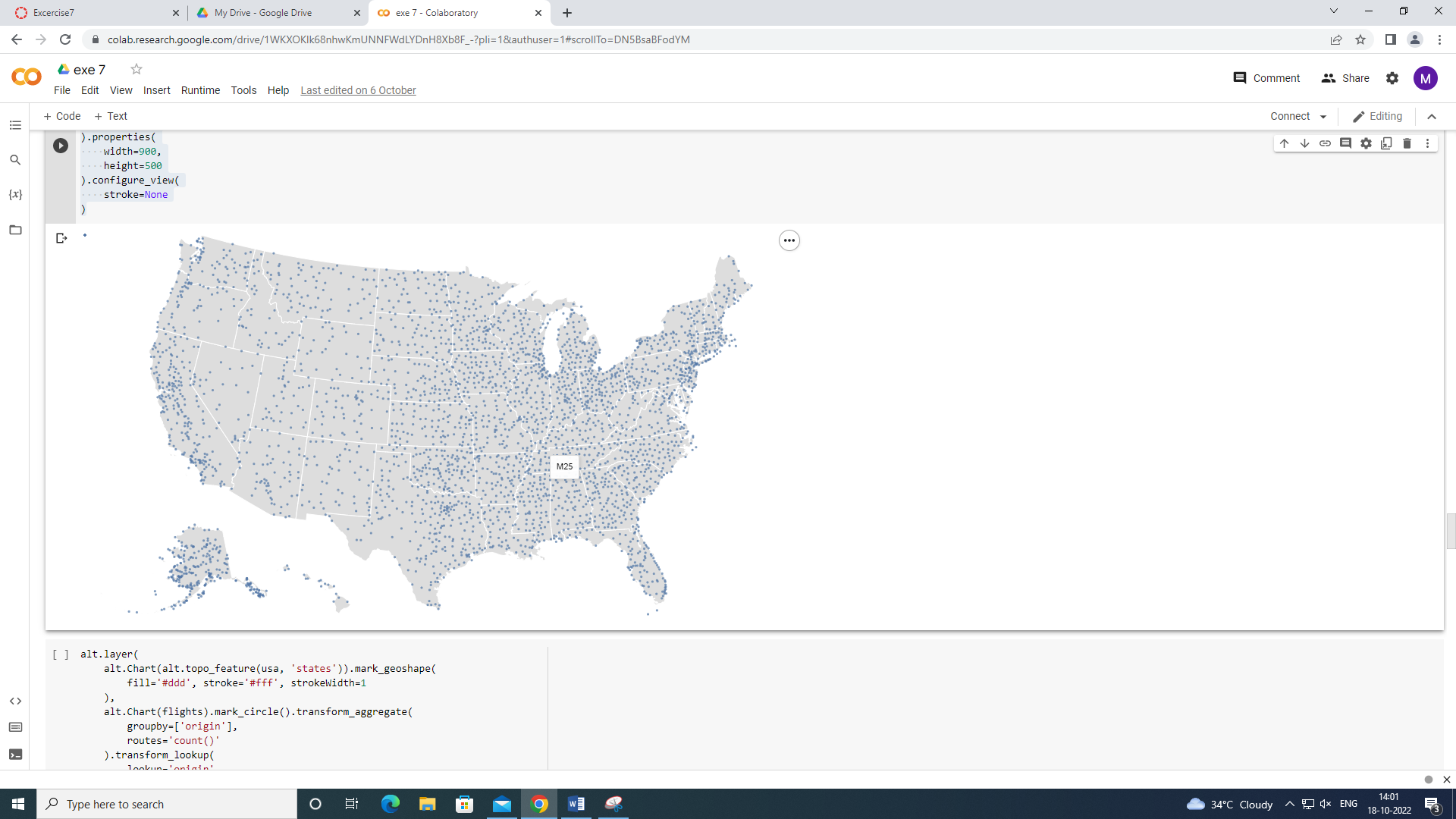
    width=900,

    height=500

).configure\_view(

    stroke=None

)



OBSERVATION:

Each airport in plotted using circles.

alt.layer(

    alt.Chart(alt.topo\_feature(usa, 'states')).mark\_geoshape(

        fill='#ddd', stroke='#fff', strokeWidth=1

    ),

    alt.Chart(flights).mark\_circle().transform\_aggregate(

        groupby=['origin'],

        routes='count()'

    ).transform\_lookup(

        lookup='origin',

        from\_=alt.LookupData(data=airports, key='iata',

                             fields=['state', 'latitude', 'longitude'])

    ).transform\_filter(

        'datum.state !== "PR" && datum.state !== "VI"'

    ).encode(

        latitude='latitude:Q',

        longitude='longitude:Q',

        tooltip=['origin:N', 'routes:Q'],

        size=alt.Size('routes:Q', scale=alt.Scale(range=[0, 1000]), legend=None),

        order=alt.Order('routes:Q', sort='descending')

    )

).project(

    type='albersUsa'

).properties(

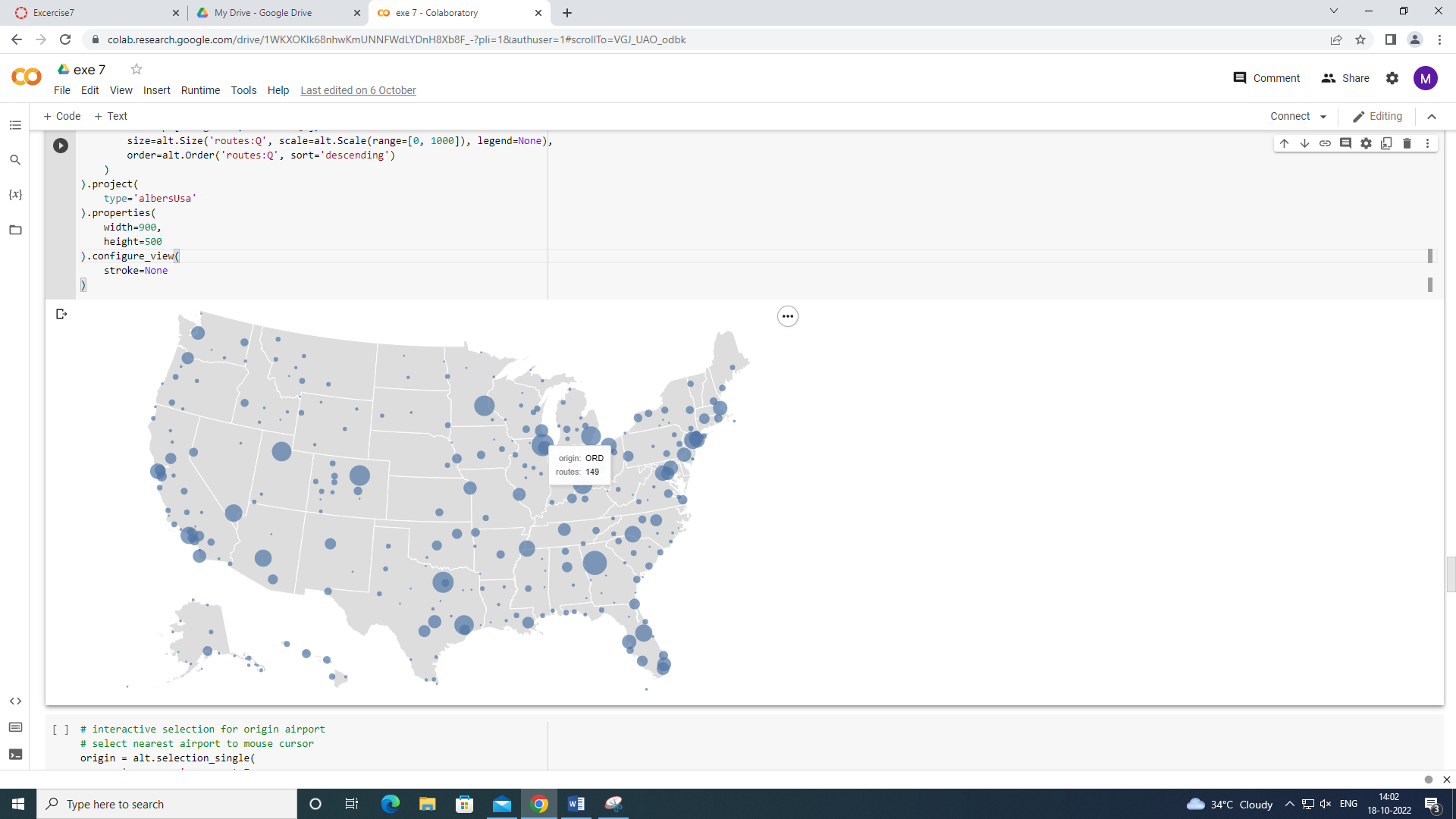
    width=900,

    height=500

).configure\_view(

    stroke=None

)



OBSERVATION:

Each airport’s route is given.

# interactive selection for origin airport

# select nearest airport to mouse cursor

origin = alt.selection\_single(

    on='mouseover', nearest=True,

    fields=['origin'], empty='none'

)

# shared data reference for lookup transforms

foreign = alt.LookupData(data=airports, key='iata',

                         fields=['latitude', 'longitude'])

alt.layer(

    # base map of the United States

    alt.Chart(alt.topo\_feature(usa, 'states')).mark\_geoshape(

        fill='#ddd', stroke='#fff', strokeWidth=1

    ),

    # route lines from selected origin airport to destination airports

    alt.Chart(flights).mark\_rule(

        color='#000', opacity=0.35

    ).transform\_filter(

        origin # filter to selected origin only

    ).transform\_lookup(

        lookup='origin', from\_=foreign # origin lat/lon

    ).transform\_lookup(

        lookup='destination', from\_=foreign, as\_=['lat2', 'lon2'] # dest lat/lon

    ).encode(

        latitude='latitude:Q',

        longitude='longitude:Q',

        latitude2='lat2',

        longitude2='lon2',

    ),

    # size airports by number of outgoing routes

    # 1. aggregate flights-airport data set

    # 2. lookup location data from airports data set

    # 3. remove Puerto Rico (PR) and Virgin Islands (VI)

    alt.Chart(flights).mark\_circle().transform\_aggregate(

        groupby=['origin'],

        routes='count()'

    ).transform\_lookup(

        lookup='origin',

        from\_=alt.LookupData(data=airports, key='iata',

                             fields=['state', 'latitude', 'longitude'])

    ).transform\_filter(

        'datum.state !== "PR" && datum.state !== "VI"'

    ).add\_selection(

        origin

    ).encode(

        latitude='latitude:Q',

        longitude='longitude:Q',

        tooltip=['origin:N', 'routes:Q'],

        size=alt.Size('routes:Q', scale=alt.Scale(range=[0, 1000]), legend=None),

        order=alt.Order('routes:Q', sort='descending') # place smaller circles on top

    )

).project(

    type='albersUsa'

).properties(

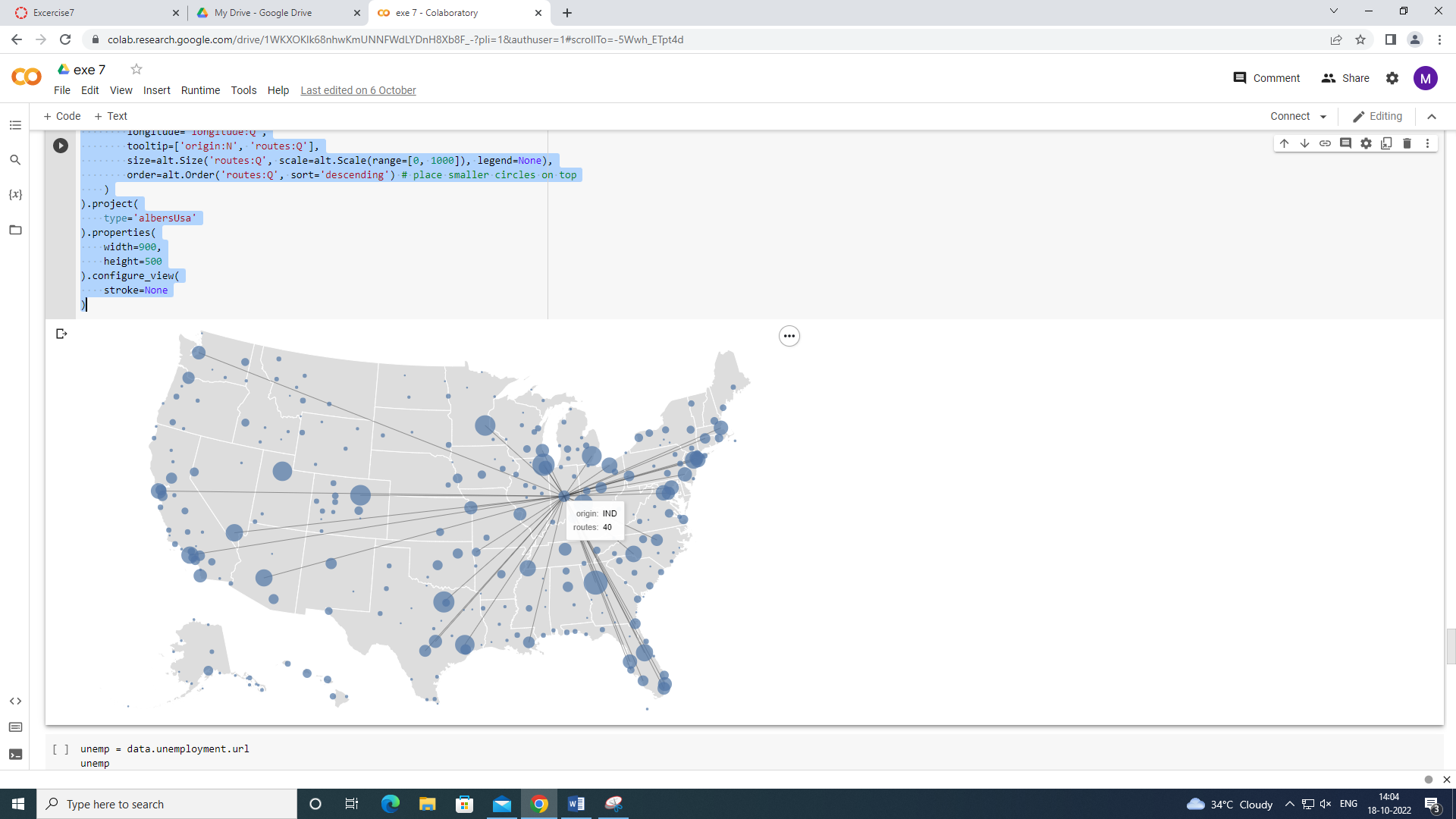
    width=900,

    height=500

).configure\_view(

    stroke=None

)



OBSERVATION:

Rule map is used to find the path from origin to destination.

unemp = data.unemployment.url

unemp

alt.Chart(alt.topo\_feature(usa, 'counties')).mark\_geoshape(

    stroke='#aaa', strokeWidth=0.25

).transform\_lookup(

    lookup='id', from\_=alt.LookupData(data=unemp, key='id', fields=['rate'])

).encode(

    alt.Color('rate:Q',

              scale=alt.Scale(domain=[0, 0.3], clamp=True),

              legend=alt.Legend(format='%')),

    alt.Tooltip('rate:Q', format='.0%')

).project(

    type='albersUsa'

).properties(

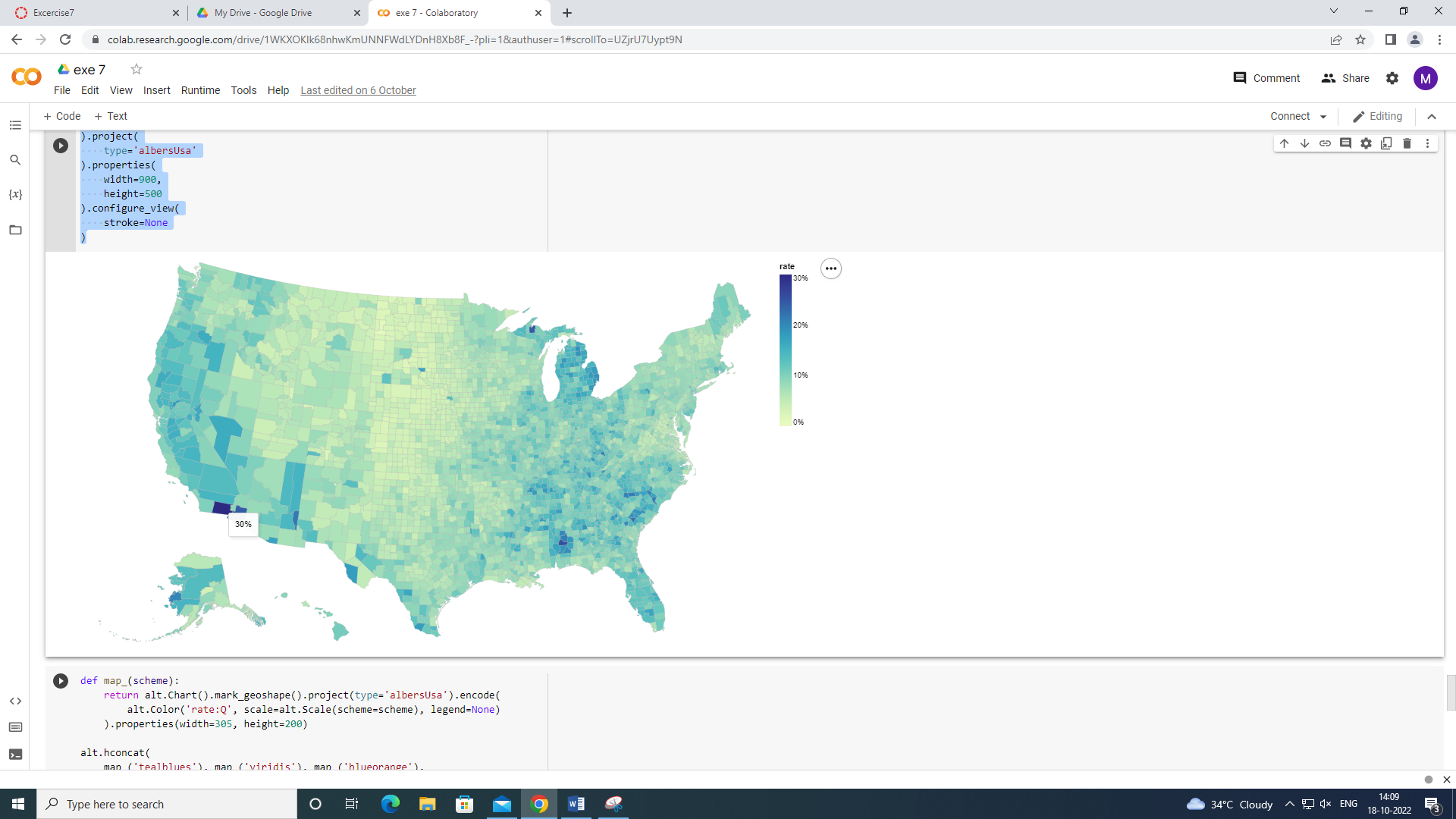
    width=900,

    height=500

).configure\_view(

    stroke=None

)



OBSERVATION:

This gives us the unemployment rate in regions.

def map\_(scheme):

    return alt.Chart().mark\_geoshape().project(type='albersUsa').encode(

        alt.Color('rate:Q', scale=alt.Scale(scheme=scheme), legend=None)

    ).properties(width=305, height=200)

alt.hconcat(

    map\_('tealblues'), map\_('viridis'), map\_('blueorange'),

    data=alt.topo\_feature(usa, 'counties')

).transform\_lookup(

    lookup='id', from\_=alt.LookupData(data=unemp, key='id', fields=['rate'])

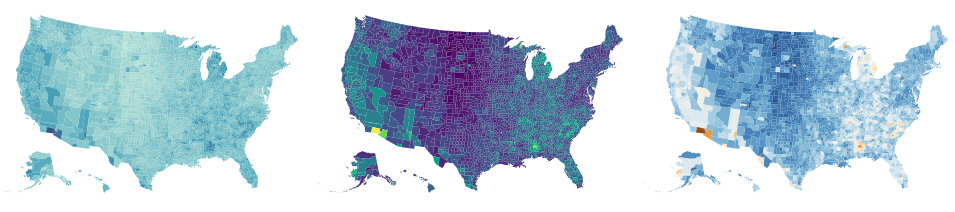
).configure\_view(

    stroke=None

).resolve\_scale(

    color='independent'

)



OBSERVATION:

Choropleth map is represented which is a choice of colours.

Result:

Thus performing of cartographic visualization for multiple datasets involving various countries of the world was successfully done.