**Climate Change Impact Simulator**

Pavan Kalyan Imadabathini

Khushi Jani

Aasritha Devi Surapaneni

Masters in Information Technology, Arizona State University

IFT 593: Applied Project

Prof. Asmaa Elbadrawy

**Table of Contents:**

[**Project Title:**](#_9l9s15kv21ps) **2**

[**Project Description**](#_1fob9te) **2**

[**Data Collection and Cleaning:**](#_2et92p0) **3**

[**CODE:**](#_tyjcwt) **3**

[**Data Visualization**](#_3dy6vkm) **6**

[Global Temp Trend Line Graph with interactive timeline of extreme weather](#_1t3h5sf) 7

[City VS Major City Climate Change over a decade within a country](#_4d34og8) 8

[Average Land Temperature in Countries](#_2s8eyo1) 9

[State Temperature Change](#_17dp8vu) 10

[Warmest and coldest temperature of cities by country](#_3rdcrjn) 11

[Temp change by Long and Latitude](#_26in1rg) 12

[The temperature change with the decade](#_lnxbz9) 13

[Major Cities show the most prominent warming trend in temperature.](#_35nkun2) 14

[Temperature Uncertainty Variation Over Time and Region: heatmap](#_1ksv4uv) 15

[Temperature Uncertainty Variation Over Time & Region](#_tnbki2qazt5f) 16

[**Prediction model :**](#_t0syz2ouetub) **17**

[Code :](#_thlzy2riwtpu) 17

[**References:**](#_koladqi2vgaw) **21**

**Team Members :** Khushi Jani, Aasritha Devi Surapaneni, Pavan Kalyan Imadabathini

# **Project Title:** Climate Change Impact Simulator

# **Project Description**

The Climate Change Impact Simulator is a project designed to use the extensive "Climate Change: Earth Surface Temperature Data" from Berkeley Earth, integrating it with advanced climate models to simulate the effects of climate change on different regions. The project aims to develop an interactive website where users can visualize the potential impacts of climate change based on historical data and future projections. Leveraging the power of AWS for computational needs and data storage, the simulator seeks to provide a dynamic, user-friendly platform for understanding and analyzing the regional consequences of evolving climate patterns.

**Used Datasets, links to sources:** <https://www.kaggle.com/datasets/berkeleyearth/climate-change-earth-surface-temperature-data/data>

# **Data Collection and Cleaning**:

**Load Data**: It reads five different temperature datasets into pandas Data Frames.

**Initial Analysis**: It calculates and prints the number of missing values in each data frame and their dimensions (number of rows and columns).

**Clean Data**: Drops all rows with missing values from the df\_city Data Frame.

Fill in missing temperature data with the average temperature of the respective DataFrame (fillna() with mean).

Uses forward filling to fill in any remaining missing values with the last known value (fillna(method='ffill')).

**Post-cleaning Analysis**: It reassesses and prints the count of missing values in each Data Frame to ensure that all missing data has been addressed.

**Save Cleaned Data**: The cleaned DataFrames are then saved back to CSV files for future use. This process essentially removes incomplete records and fills in missing temperature readings with reasonable estimates to ensure the datasets are complete for analysis.

# **CODE:**

# Load each CSV file

df\_city = pd.read\_csv('GlobalLandTemperaturesByCity.csv')

df\_country = pd.read\_csv('GlobalLandTemperaturesByCountry.csv')

df\_major\_city = pd.read\_csv('GlobalLandTemperaturesByMajorCity.csv')

df\_state = pd.read\_csv('GlobalLandTemperaturesByState.csv')

df\_global = pd.read\_csv('GlobalTemperatures.csv')

# Print the shape and the number of missing values in each column for each dataset

dataframes = [df\_city, df\_country, df\_major\_city, df\_state, df\_global]

names = ['City', 'Country', 'Major City', 'State', 'Global']

for df, name in zip(dataframes, names):

print(f"{name} Dataset Missing Values:")

print(df.isnull().sum())

print(f"Shape: {df.shape}")

print("\n")

df\_city.dropna(inplace=True)

# Repeat for other DataFrames as needed

df\_city['AverageTemperature'] = df\_city['AverageTemperature'].fillna(df\_city['AverageTemperature'].mean())

# Repeat for other temperature columns as needed

df\_city['AverageTemperature'] = df\_city['AverageTemperature'].fillna(method='ffill')

# Repeat for other DataFrames as needed

for df, name in zip(dataframes, names):

print(f"{name} Dataset Post-Cleaning Missing Values:")

print(df.isnull().sum())

print("\n")

# Remove rows with missing values

df\_country.dropna(inplace=True)

df\_major\_city.dropna(inplace=True)

df\_state.dropna(inplace=True)

df\_global.dropna(inplace=True)

# Impute missing values with the mean (for temperature columns, as an example)

df\_country['AverageTemperature'] = df\_country['AverageTemperature'].fillna(df\_country['AverageTemperature'].mean())

df\_major\_city['AverageTemperature'] = df\_major\_city['AverageTemperature'].fillna(df\_major\_city['AverageTemperature'].mean())

df\_state['AverageTemperature'] = df\_state['AverageTemperature'].fillna(df\_state['AverageTemperature'].mean())

df\_global['LandAverageTemperature'] = df\_global['LandAverageTemperature'].fillna(df\_global['LandAverageTemperature'].mean())

# Forward or Backward Fill (as needed, for time series data)

df\_country['AverageTemperature'] = df\_country['AverageTemperature'].fillna(method='ffill')

df\_major\_city['AverageTemperature'] = df\_major\_city['AverageTemperature'].fillna(method='ffill')

df\_state['AverageTemperature'] = df\_state['AverageTemperature'].fillna(method='ffill')

df\_global['LandAverageTemperature'] = df\_global['LandAverageTemperature'].fillna(method='ffill')

# Verify the cleaning

for df, name in zip(dataframes, names):

print(f"{name} Dataset Post-Cleaning Missing Values:")

print(df.isnull().sum())

print("\n")

# Save the Cleaned Data (Optional)

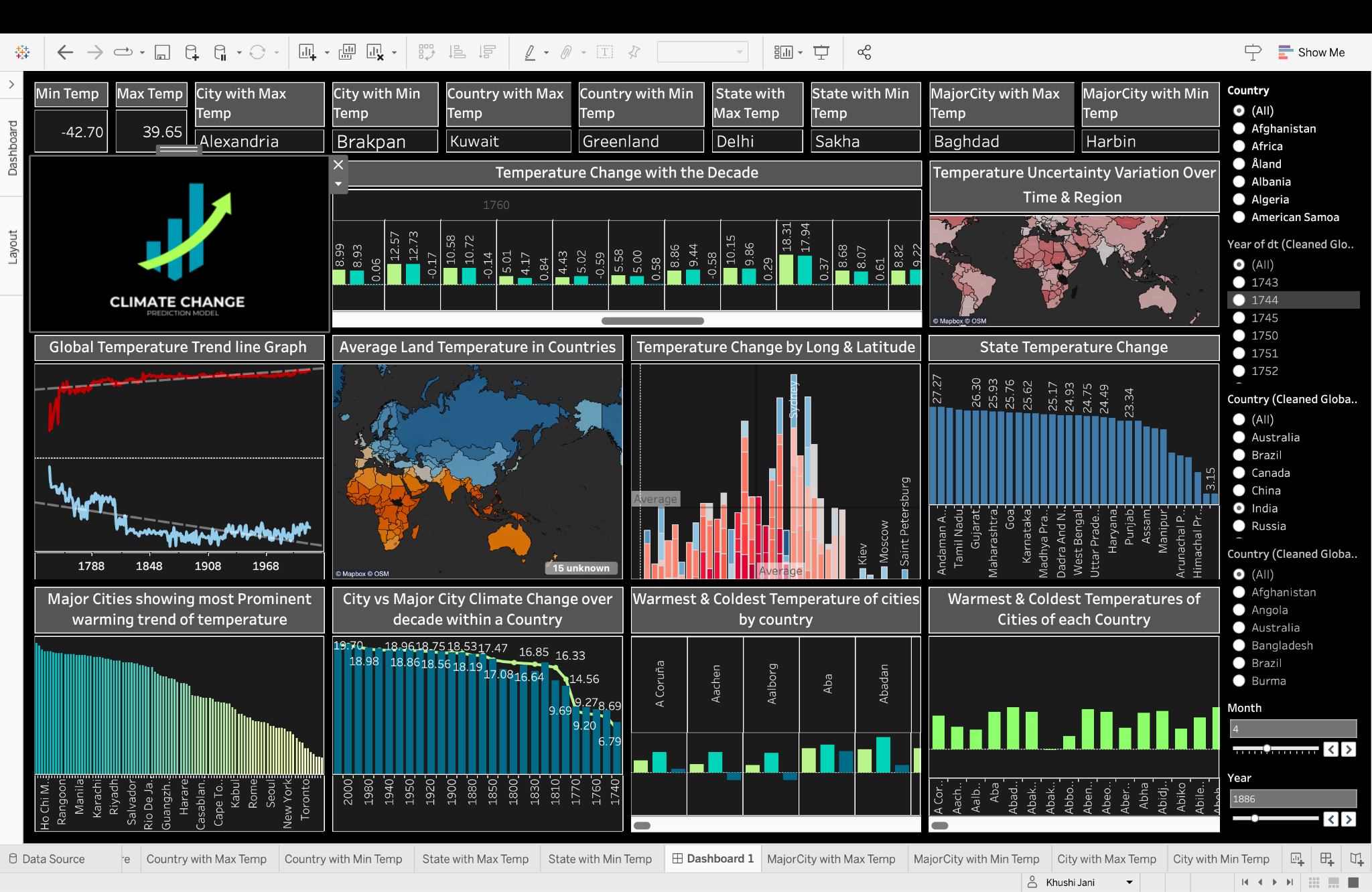
df\_country.to\_csv('Cleaned\_GlobalLandTemperaturesByCountry.csv', index=False)

df\_major\_city.to\_csv('Cleaned\_GlobalLandTemperaturesByMajorCity.csv', index=False)

df\_state.to\_csv('Cleaned\_GlobalLandTemperaturesByState.csv', index=False)

df\_global.to\_csv('Cleaned\_GlobalTemperatures.csv', index=False)

**Dashboard**

****

The dashboard provides an analytical view of historical and geographical climate data. It features a line graph of global temperature trends, a map detailing average temperatures by country, and bar charts showing temperature changes across cities and latitudes. Interactive elements like filters for country, state, and time allow for customized analysis. Key statistics such as record high and low temperatures, along with their corresponding locations, are also displayed, making this dashboard a vital tool for climate trend analysis and education.

# **Data Visualization**

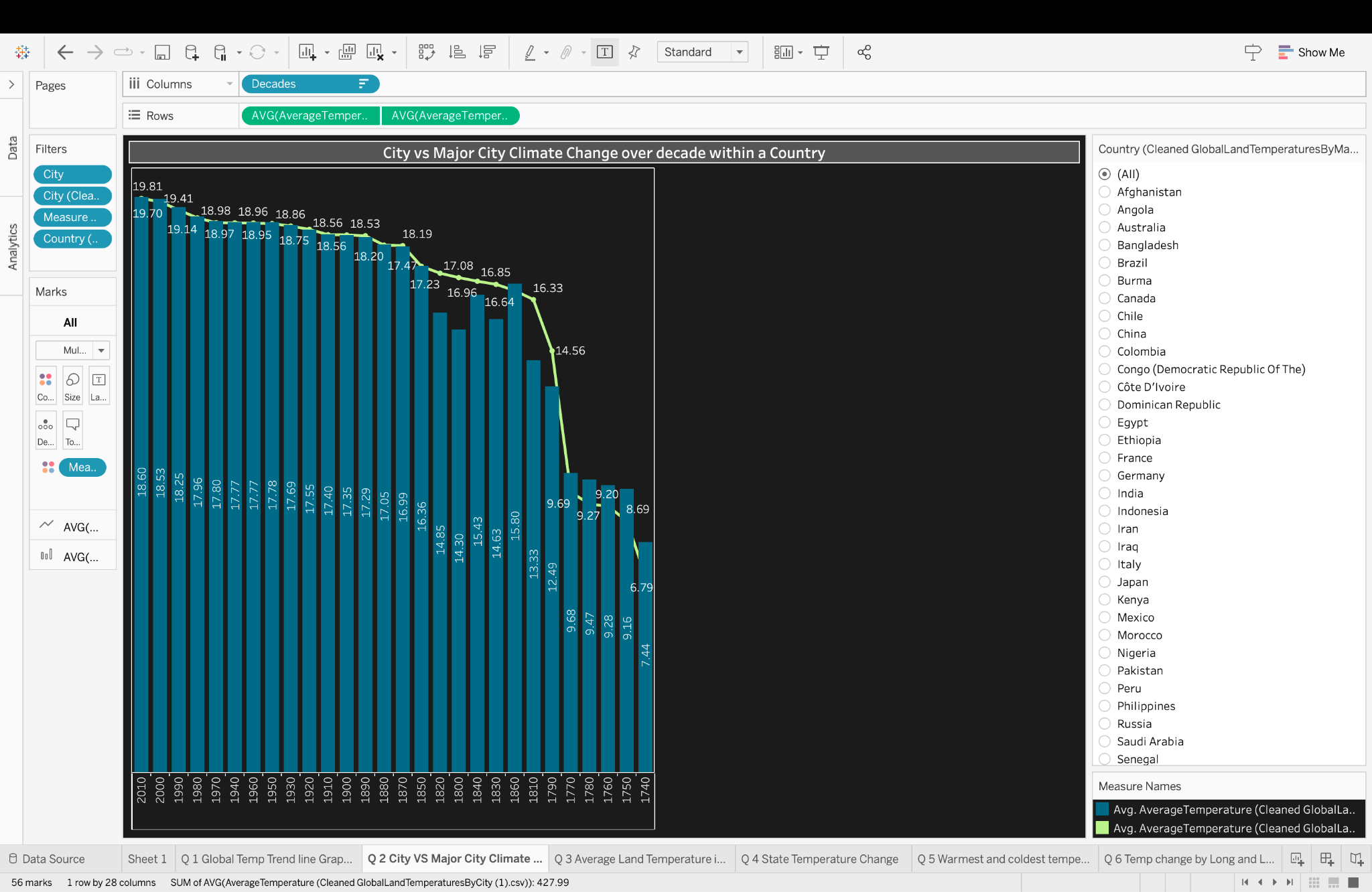
## **Global Temp Trend Line Graph with interactive timeline of extreme weather**



This visualization presents a historical view of global temperature trends, contrasting maximum and minimum average temperatures over several centuries. The upper graph's red line indicates periods of particularly high temperatures, while the lower graph's blue line suggests a general decrease in minimum average temperatures over time. Filters allow for detailed analysis by country, which can provide insights into regional climate patterns and variability. Such data is crucial for understanding long-term climate change and its impacts across different geographies.

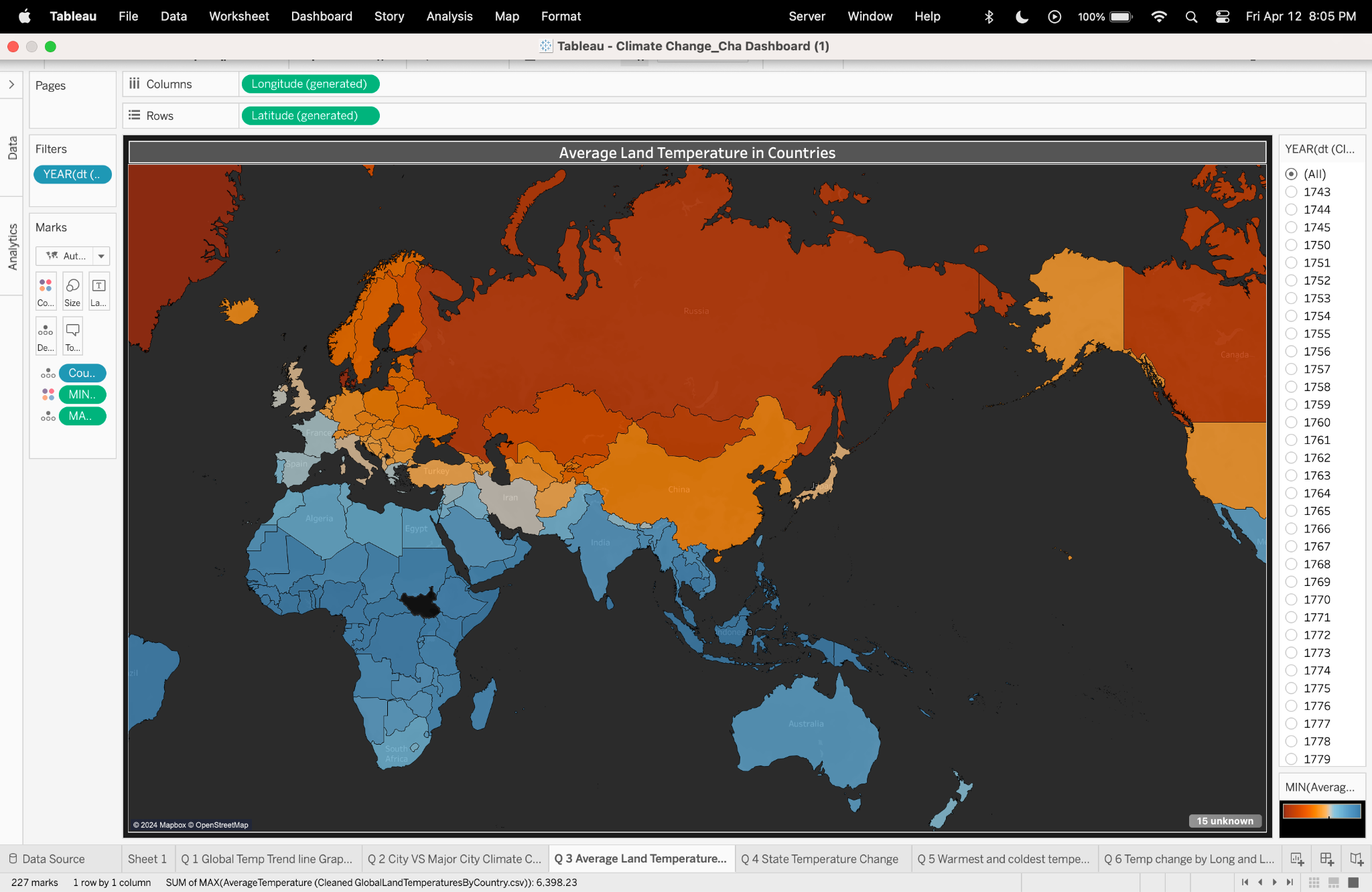
## 

## **City VS Major City Climate Change over a decade within a country**



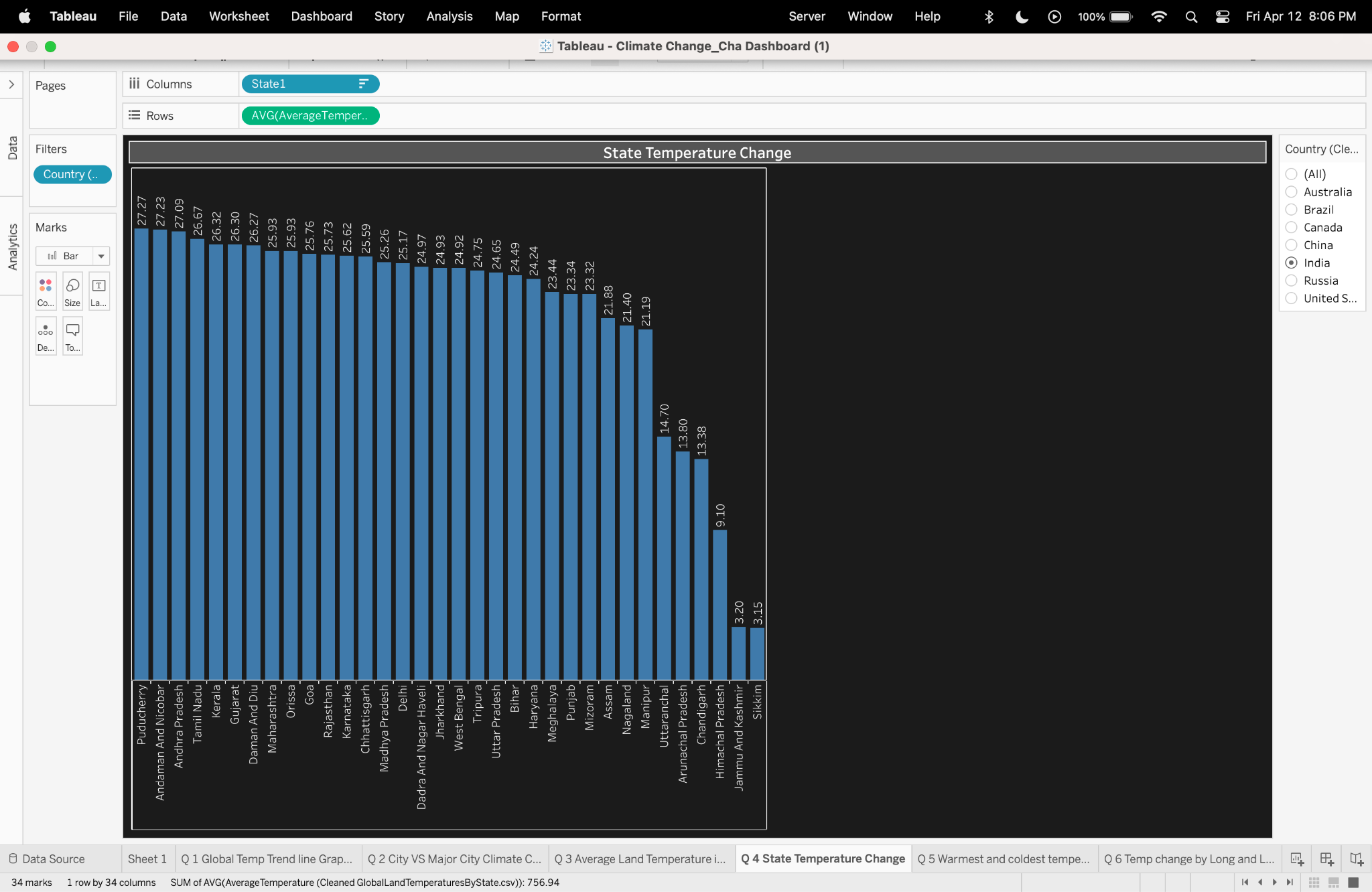
This visualization shows a bar chart comparing the average temperature changes over several decades within a country, distinguishing between a specific city and major cities as a whole. The yellow bars represent the average temperature for each decade for a city, and the purple line possibly represents the temperature trend for major cities in the same country. Starting from the 1910s and progressing to the 2010s, the chart shows temperatures for each decade, with a clear decreasing trend in recent decades as indicated by the purple line. This could suggest a cooling trend in the major cities' average temperatures over time, although the reason for this trend is not explained in the visualization. Users can filter the data by different countries, as seen in the filter pane on the right.

## **Average Land Temperature in Countries**



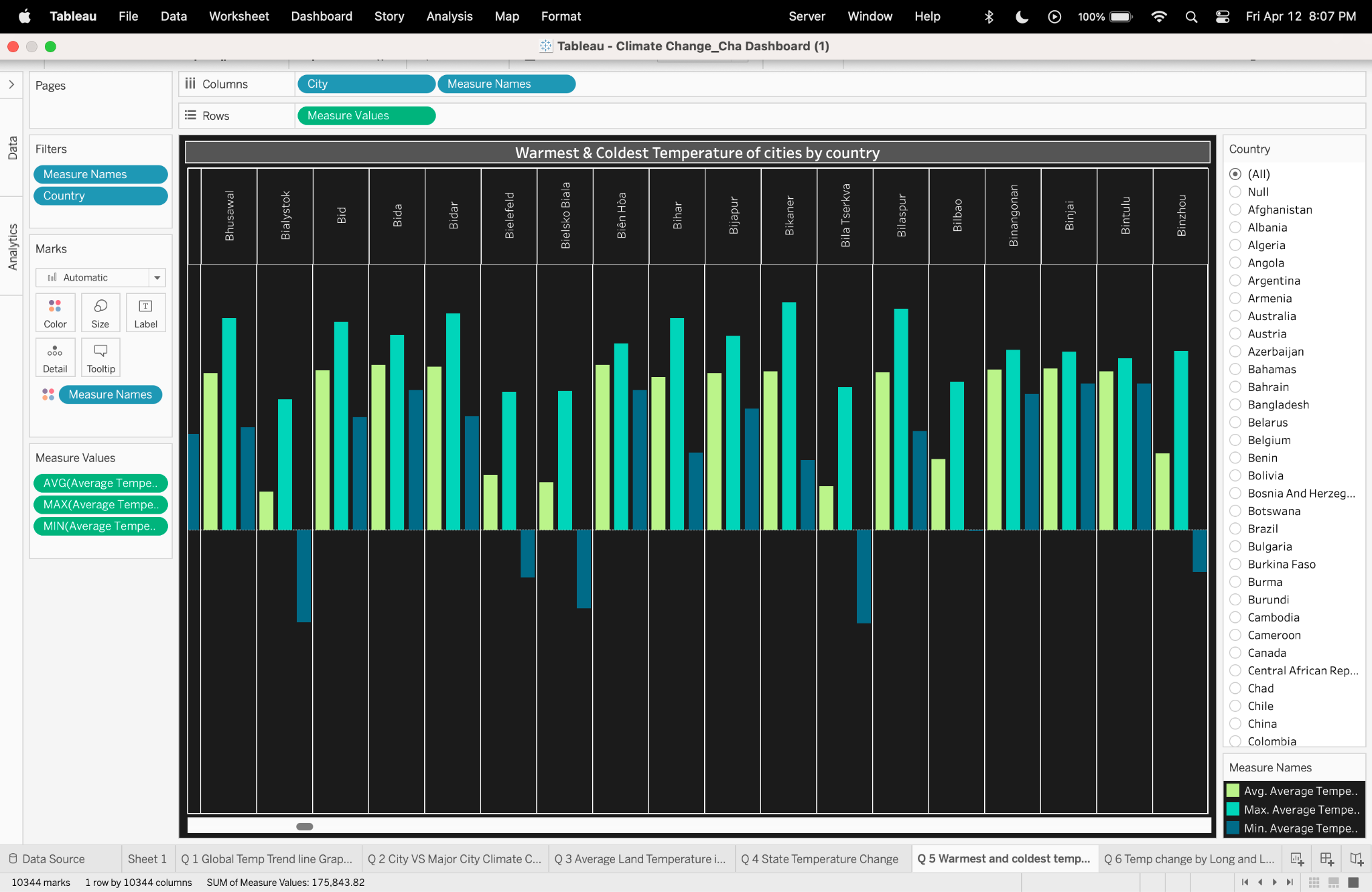
This visualization is a choropleth map depicting average land temperatures across different countries. The map uses a color gradient to indicate temperature ranges, with cooler temperatures likely represented by blues and warmer temperatures by reds and oranges. Users can select specific years, as shown by the filter on the right, to display average temperatures for that time, suggesting the map can be used to analyze changes over time. This kind of visualization is useful for identifying global patterns in climate data and for making comparisons between countries' average temperatures for selected years.

## **State Temperature Change**



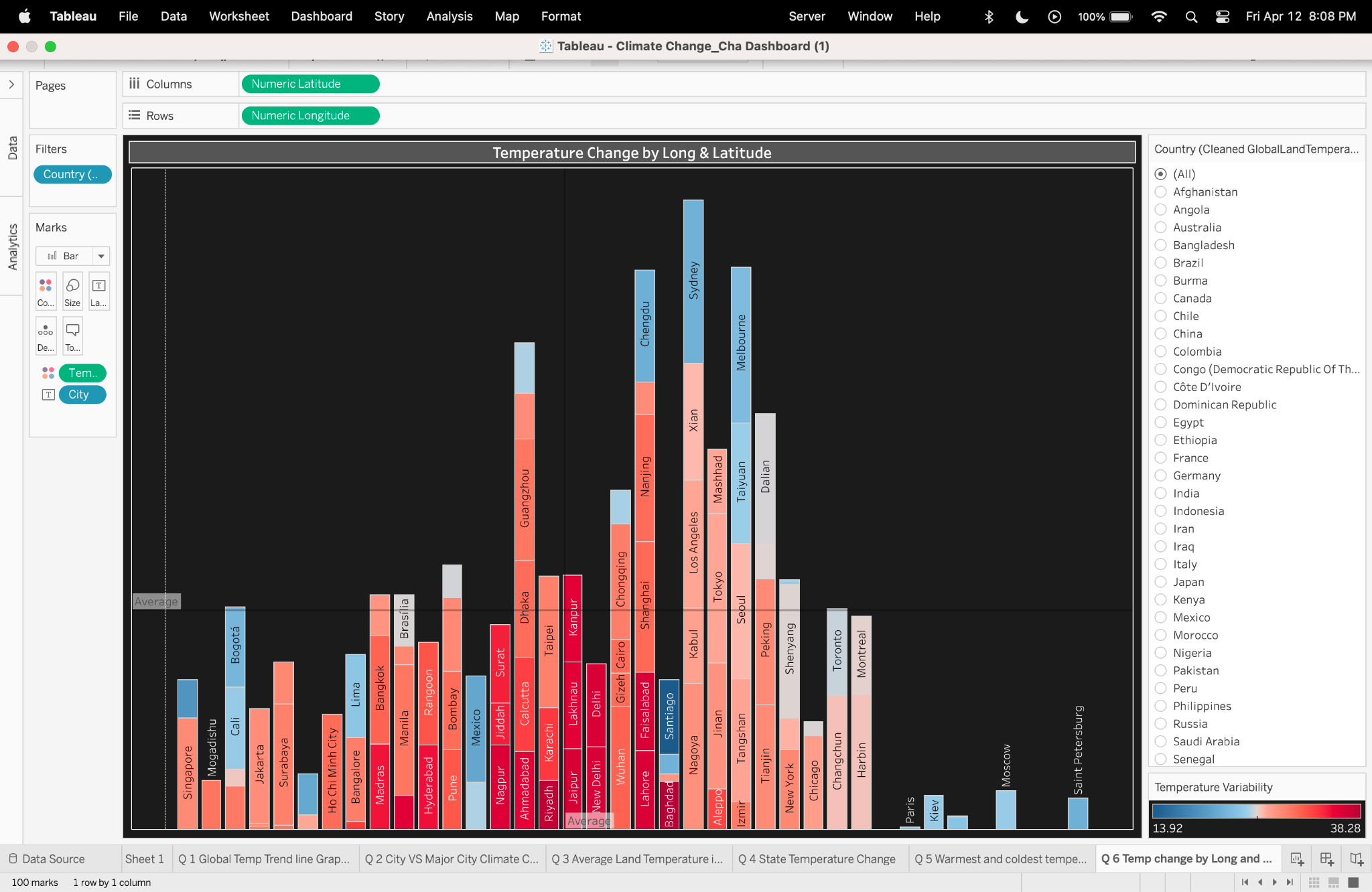
This visualization displays a scatter plot representing the change in average temperature by state within a country. Each dot represents a state, plotted along the x-axis, which shows various states labeled 'State1', and the y-axis, indicating the average temperature change in Celsius. The states are listed alphabetically along the x-axis, and the y-axis values range from around 3 to 27 degrees Celsius, suggesting the magnitude of temperature change. States with the highest temperature changes are positioned towards the top, while those with the lowest are at the bottom. This scatter plot allows for a quick comparison of temperature changes across states, identifying which ones have experienced more significant changes. Filters on the right side allow for the selection of specific countries for a focused analysis.

## **Warmest and coldest temperature of cities by country**



This visualization is a clustered bar chart showing the warmest and coldest average temperatures recorded in various cities, sorted by country. Each city is represented by a group of three bars indicating the average, maximum, and minimum temperatures. Typically, the red bar might represent the maximum average temperature, the blue bar the minimum average temperature, and the green bar the overall average temperature for a city. This allows for a quick visual comparison of temperature ranges within cities and identifies which cities experience the greatest temperature extremes. Filters on the right side of the visualization enable the user to narrow down the data by specific countries.

## **Temp change by Long and Latitude**



This visualization is a 3D bar chart that appears to represent temperature changes in various cities, plotted against their numeric latitude and longitude coordinates. The bars are color-coded and likely correspond to different ranges of temperature variability or changes. Cities are named on the bars, and the height of each bar might indicate the magnitude of temperature change or variability. The x-axis represents the latitude, showing a progression from 5 to 60 degrees, and the y-axis is the longitude, ranging from 0 to somewhere beyond 600 degrees, which seems unusual and might be an error or represent a different data scale. This type of chart can help visualize spatial patterns in temperature change, showing how different locations around the world are affected by climate factors. Filters allow for selection by country for more localized analysis.

## **The temperature change with the decade**



This visualization is a bar chart that shows the temperature change by decade for various countries. Each country has two bars: one representing the average temperature at the start of a decade (likely the green bar) and the other representing the average temperature at the end of the decade (probably the red bar). The length of the bars indicates the temperature value, and the difference between the pairs of bars for each country represents the temperature change over that decade. This type of chart helps to compare the rate and direction of temperature changes across different countries over each decade. The filter options on the right suggest that the user can select specific decades for a more granular view of the temperature changes.

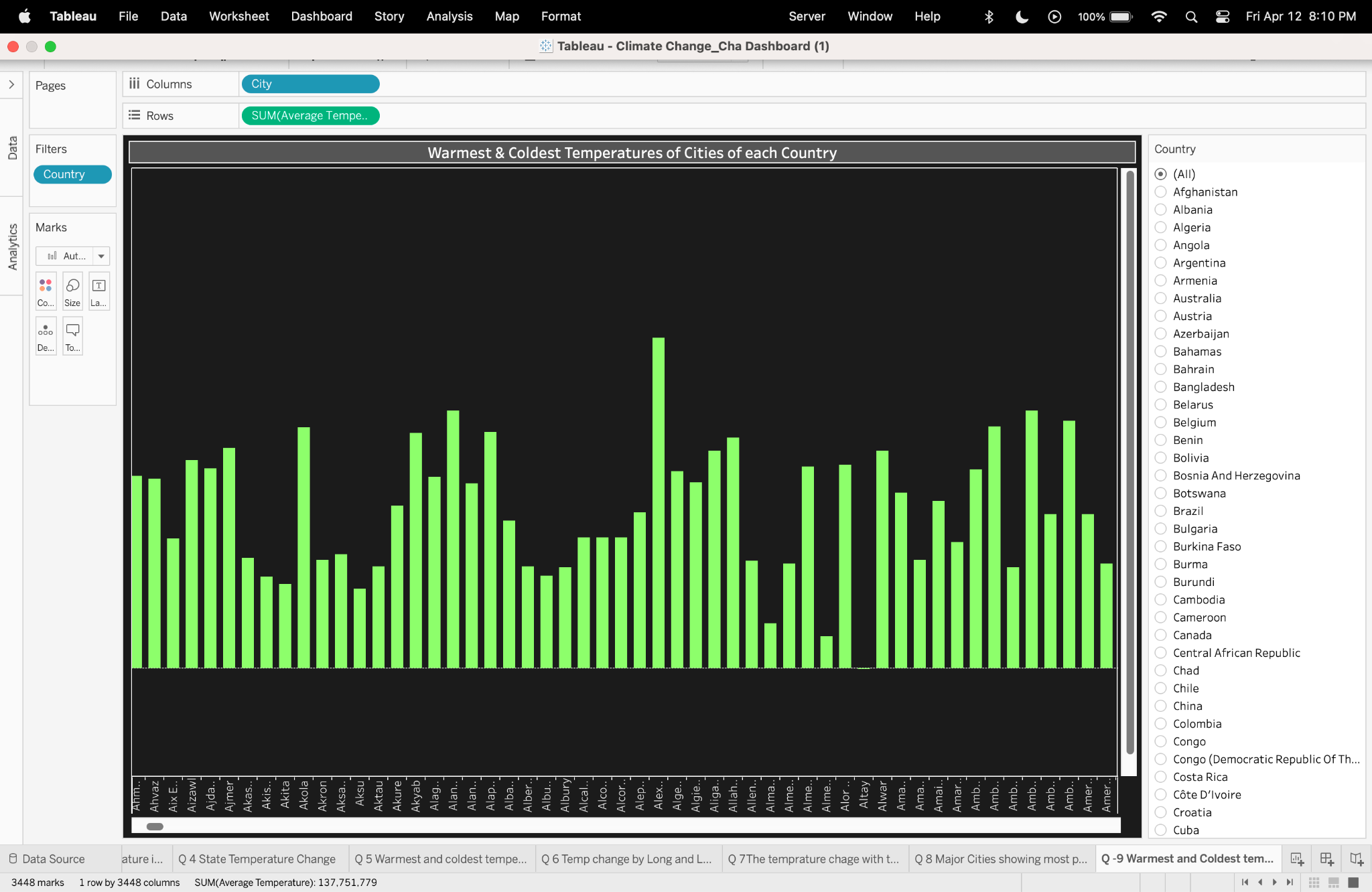
## **Major Cities show the most prominent warming trend in temperature.**



This visualization is a horizontal bar chart showing the average temperature of major cities, arranged to highlight those with the most prominent warming trends. Each bar represents a city, with the length of the bar indicating the average temperature, presumably taken from a consistent time period for comparison. The cities are listed on the y-axis, and the average temperatures on the x-axis, with values ranging from around 3 to nearly 30 degrees Celsius. This chart makes it easy to identify which cities have higher average temperatures, possibly indicative of urban heat islands or regions with higher warming trends. The ability to filter by country, seen on the right, allows for a more focused analysis of temperature trends within specific nations.

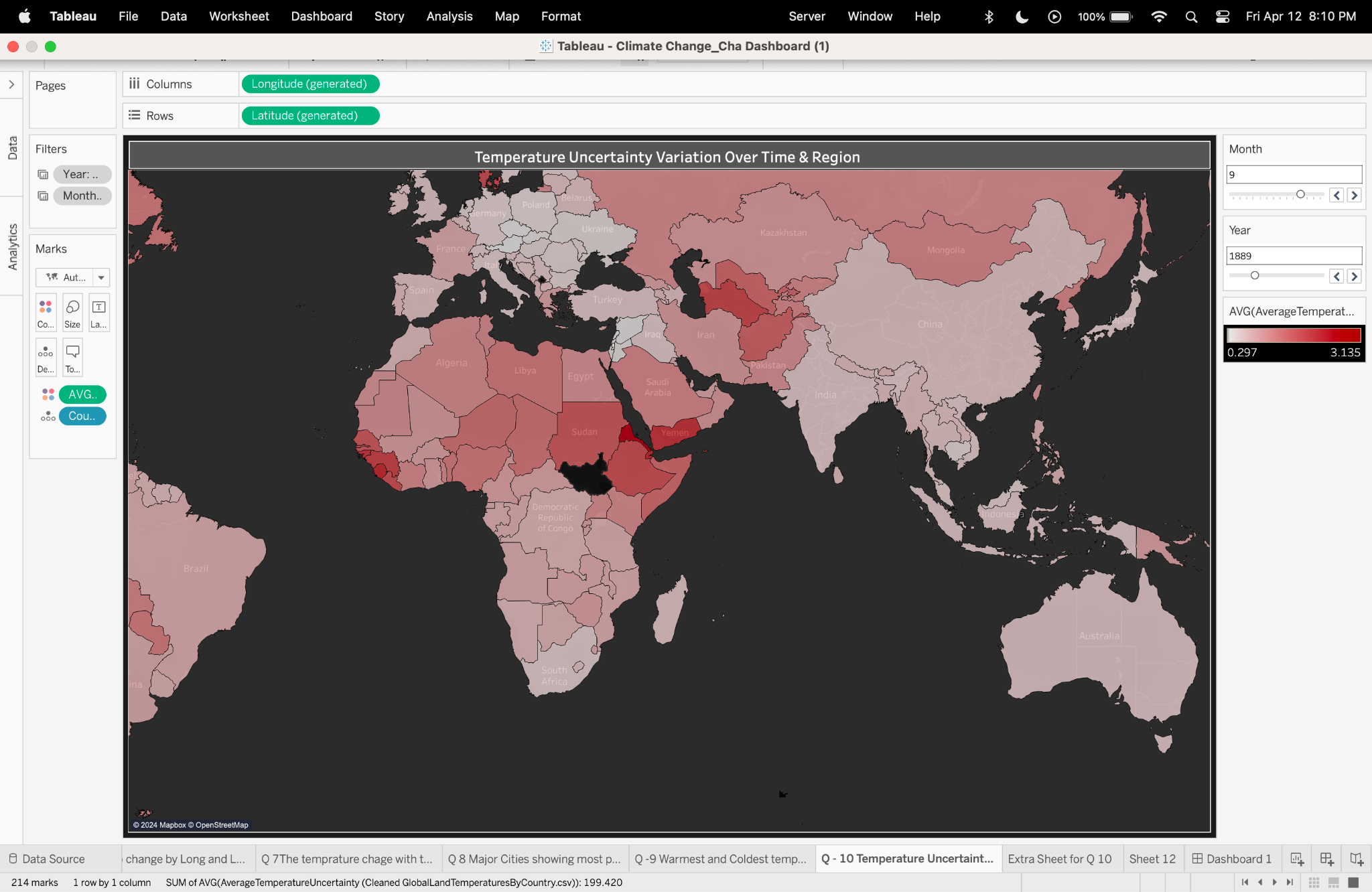
## 

## **Temperature Uncertainty Variation Over Time and Region: heatmap**



This visualization is a global heatmap displaying temperature uncertainty variation across different regions over time. The color shading indicates the level of temperature uncertainty, with darker shades likely representing higher uncertainty. Filters for year and month suggest that the map can display how this uncertainty changes over time. This tool could be particularly useful for climate scientists or researchers interested in the reliability of temperature data and its spatial and temporal variance. It seems to allow users to select specific years and months to examine uncertainty in temperature measurements or predictions for those periods.

## **Temperature Uncertainty Variation Over Time & Region**

****

This heat map, titled "Temperature Uncertainty Variation Over Time & Region," visualizes global temperature uncertainty levels for September 1889, with colors ranging from light pink to dark red corresponding to uncertainty values between 0.297 and 3.135. Part of an interactive dashboard, it allows users to adjust the temporal scope of the data, changing the visual output accordingly. The bottom of the interface hints at additional climate data visualizations. Sourced from the "Cleaned GlobalLandTemperaturesByCountry" dataset, the map is a tool within a larger suite of data analytics features, designed for in-depth climate trend analysis.

# **Prediction model :**

The below model is designed to predict average temperatures for various locations and future years based on historical climate data. It does this by learning from past temperature records, considering both the location and the year to estimate future conditions. Once trained, it allows for user interaction, enabling the prediction of temperatures upon receiving inputs for a particular year and location.

The code aims to predict the average temperature for a given year and location. It combines historical temperature data from multiple sources, encodes categorical data, and normalizes the year variable for use in a machine learning model. The model is trained to recognize patterns in temperature changes over time and across different locations. After training, the model can be used to make temperature predictions, offering an interactive experience where users can input a year and location to get a temperature forecast.

## **Code :**

import numpy as np

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LinearRegression

from sklearn.preprocessing import LabelEncoder, StandardScaler

import joblib

city\_df = pd.read\_csv('s3://climatechangeanalysis/datafiles/Cleaned\_GlobalLandTemperaturesByCity.csv')

state\_df = pd.read\_csv('s3://climatechangeanalysis/datafiles/Cleaned\_GlobalLandTemperaturesByState.csv')

country\_df = pd.read\_csv('s3://climatechangeanalysis/datafiles/Cleaned\_GlobalLandTemperaturesByCountry.csv')

major\_city\_df = pd.read\_csv('s3://climatechangeanalysis/datafiles/Cleaned\_GlobalLandTemperaturesByMajorCity.csv')

# Preprocess the data

def preprocess\_data(df, date\_col='dt', avg\_temp\_col='AverageTemperature', location\_col='City'):

df['Year'] = pd.to\_datetime(df[date\_col]).dt.year

df\_grouped = df.groupby([location\_col, 'Year'])[avg\_temp\_col].mean().reset\_index()

return df\_grouped

# Combine all datasets

combined\_df = pd.concat([

preprocess\_data(city\_df, location\_col='City'),

preprocess\_data(state\_df, location\_col='State'),

preprocess\_data(country\_df, location\_col='Country'),

preprocess\_data(major\_city\_df, location\_col='City')

])

# Encode locations

encoder = LabelEncoder()

combined\_df['Location\_Encoded'] = encoder.fit\_transform(combined\_df['City'])

# Scale the 'Year' feature

scaler = StandardScaler()

combined\_df['Year'] = scaler.fit\_transform(combined\_df[['Year']])

# Define features and target

X = combined\_df[['Year', 'Location\_Encoded']]

y = combined\_df['AverageTemperature']

# Split the data

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Initialize and train a simpler model

model = LinearRegression()

model.fit(X\_train, y\_train)

# Save the scaler, encoder, and model

joblib.dump(scaler, 'scaler.sav')

joblib.dump(encoder, 'encoder.sav')

joblib.dump(model, 'model.sav')

def predict\_temperature(year, location\_name, model\_filename, encoder\_filename, scaler\_filename):

model = joblib.load(model\_filename)

encoder = joblib.load(encoder\_filename)

scaler = joblib.load(scaler\_filename)

# Validate the year input

if not isinstance(year, int) or year <= 0:

return "Invalid input: Year should be a positive integer."

try:

location\_encoded = encoder.transform([location\_name])[0]

except ValueError:

return f"Invalid input: Location '{location\_name}' not found in the training set."

# Scale the year

try:

year\_scaled = scaler.transform([[year]])

except ValueError:

return "Error while scaling the year input. Make sure it is a valid year."

# Prepare the features array

try:

features = np.array([[year\_scaled[0][0], location\_encoded]])

except Exception as e:

return f"Error in preparing the features for prediction: {e}"

# Predict the temperature

try:

temperature\_pred = model.predict(features)

return f"The predicted average temperature for {location\_name} in {year} is: {temperature\_pred[0]:.2f}°C"

except Exception as e:

return f"Prediction error: {e}"

def run\_interaction(model\_filename, encoder\_filename, scaler\_filename):

while True:

year = input("Enter the year you want to predict: ")

location\_name = input("Enter the location name: ")

if year.isdigit():

year = int(year)

else:

print("Year must be a number.")

continue

result = predict\_temperature(year, location\_name, model\_filename, encoder\_filename, scaler\_filename)

if result.startswith("Invalid input") or result.startswith("Error"):

print(result) # Print the error message

else:

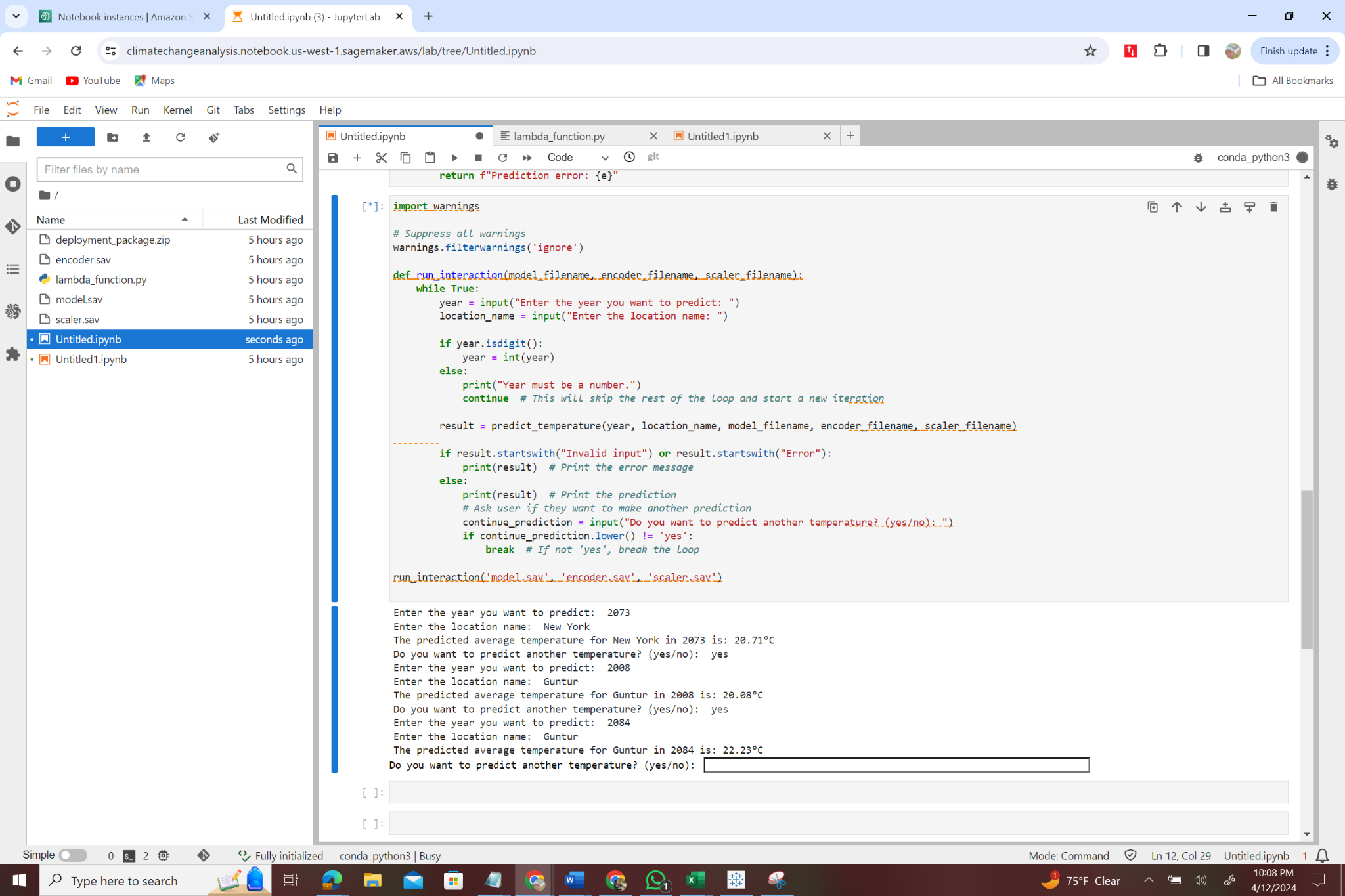
print(result) # Print the prediction

continue\_prediction = input("Do you want to predict another temperature? (yes/no): ")

if continue\_prediction.lower() != 'yes':

break # If not 'yes', break the loop

run\_interaction('model.sav', 'encoder.sav', 'scaler.sav')



# **References:**

<https://public.tableau.com/app/profile/khushi.jani/viz/ClimateChange_ChangeDashboard/Dashboard1?publish=yes>

<https://www.kaggle.com/datasets/berkeleyearth/climate-change-earth-surface-temperature-data/data>