Transformations on Flat File

Kaylynn Mosier

15 April 2024

In [1]:
```
# Importing required packages
import pandas as pd
import numpy as np
```

In [23]:
```
# Load and print data
temperature_data = pd.read_csv("C:/Users/kayly/OneDrive/Desktop/MSDS/DSC540/Tem Project/TemperatureByCity.csv")
temperature_data
```

Out[23]:

	Region	Country	State	City	Month	Day	Year	AvgTemperature
0	Africa	Algeria	NaN	Algiers	1	1	1995	64.2
1	Africa	Algeria	NaN	Algiers	1	2	1995	49.4
2	Africa	Algeria	NaN	Algiers	1	3	1995	48.8
3	Africa	Algeria	NaN	Algiers	1	4	1995	46.4
4	Africa	Algeria	NaN	Algiers	1	5	1995	47.9
...	...	...	...	...	...	...	...	...
1048570	Middle East	Oman	NaN	Muscat	2	7	1999	-99.0
1048571	Middle East	Oman	NaN	Muscat	2	8	1999	-99.0
1048572	Middle East	Oman	NaN	Muscat	2	9	1999	-99.0
1048573	Middle East	Oman	NaN	Muscat	2	10	1999	-99.0
1048574	Middle East	Oman	NaN	Muscat	2	11	1999	-99.0

1048575 rows × 8 columns

Transformation 1: Setting index

In [24]:
```
temperature_data.set_index('City', inplace=True)
temperature_data
```

Out[24]:

City	Region	Country	State	Month	Day	Year	AvgTemperature
Algiers	Africa	Algeria	NaN	1	1	1995	64.2
Algiers	Africa	Algeria	NaN	1	2	1995	49.4
Algiers	Africa	Algeria	NaN	1	3	1995	48.8
Algiers	Africa	Algeria	NaN	1	4	1995	46.4
Algiers	Africa	Algeria	NaN	1	5	1995	47.9
...	...	...	...	...	...	...	...
Muscat	Middle East	Oman	NaN	2	7	1999	-99.0
Muscat	Middle East	Oman	NaN	2	8	1999	-99.0
Muscat	Middle East	Oman	NaN	2	9	1999	-99.0
Muscat	Middle East	Oman	NaN	2	10	1999	-99.0
Muscat	Middle East	Oman	NaN	2	11	1999	-99.0

1048575 rows × 7 columns

Transformation 2: Dropping State column

In [25]:
```
# Accessing State column and counting the number of null values
temperature_data['State'].isna().sum()
```

Out[25]:
```
1048575
```

This entire column is empty, so it adds no value to my research.

In [26]:
```
# Dropping State column from dataframe and updatating the dataframe to reflect this change
temperature_data = temperature_data.drop(columns = 'State')
temperature_data
```

Out[26]:

City	Region	Country	Month	Day	Year	AvgTemperature
Algiers	Africa	Algeria	1	1	1995	64.2
Algiers	Africa	Algeria	1	2	1995	49.4
Algiers	Africa	Algeria	1	3	1995	48.8
Algiers	Africa	Algeria	1	4	1995	46.4
Algiers	Africa	Algeria	1	5	1995	47.9
...	...	...	...	...	...	...
Muscat	Middle East	Oman	2	7	1999	-99.0
Muscat	Middle East	Oman	2	8	1999	-99.0
Muscat	Middle East	Oman	2	9	1999	-99.0
Muscat	Middle East	Oman	2	10	1999	-99.0
Muscat	Middle East	Oman	2	11	1999	-99.0

1048575 rows × 6 columns

Transformation 3: Removing values from AvgTemperature column

When examining the data, I realized the AvgTemperature column contains values for -99. These seem to be filling Na values.

In [28]:
```
# Removing data with AvgTemperature values greater than -99
temperature_data = temperature_data[temperature_data['AvgTemperature'] > -99]
temperature_data
```

Out[28]:

City	Region	Country	Month	Day	Year	AvgTemperature
Algiers	Africa	Algeria	1	1	1995	64.2
Algiers	Africa	Algeria	1	2	1995	49.4
Algiers	Africa	Algeria	1	3	1995	48.8
Algiers	Africa	Algeria	1	4	1995	46.4
Algiers	Africa	Algeria	1	5	1995	47.9
...	...	...	...	...	...	...
Beirut	Middle East	Lebanon	5	13	2020	67.8
Muscat	Middle East	Oman	4	19	1995	82.8
Muscat	Middle East	Oman	9	24	1995	94.5
Muscat	Middle East	Oman	9	25	1995	92.6
Muscat	Middle East	Oman	10	1	1995	93.1

992090 rows × 6 columns

Transformation 4: Change column title

I plan to add a columns that calculates average temperatures by month and by year for each city. The current AvgTemperature column shows the average daily temperature. This column title will become confusing when I add the new column.

In [29]:
```
# Renaming AvgTemperature column using remane method
temperature_data = temperature_data.rename(columns={"AvgTemperature":"AvgDailyTemp"})
temperature_data
```

Out[29]:

City	Region	Country	Month	Day	Year	AvgDailyTemp
Algiers	Africa	Algeria	1	1	1995	64.2
Algiers	Africa	Algeria	1	2	1995	49.4
Algiers	Africa	Algeria	1	3	1995	48.8
Algiers	Africa	Algeria	1	4	1995	46.4
Algiers	Africa	Algeria	1	5	1995	47.9
...	...	...	...	...	...	...
Beirut	Middle East	Lebanon	5	13	2020	67.8
Muscat	Middle East	Oman	4	19	1995	82.8
Muscat	Middle East	Oman	9	24	1995	94.5
Muscat	Middle East	Oman	9	25	1995	92.6
Muscat	Middle East	Oman	10	1	1995	93.1

992090 rows × 6 columns

Transformation 5: Removing years that are not possible

In trying to recode data to datetime format, I found an errors in the year column. This dataset is supposed to cover years from 1995-2020. A few entires contain values of 200 and 201. Becuase I cannot tell what these years are supposed to be, I will remove all years less than 1995 from the dataset.

In [30]:
```
# Removing data with Year values less than 1995
temperature_data = temperature_data[temperature_data['Year'] > 1995]
temperature_data
```

Out[30]:

City	Region	Country	Month	Day	Year	AvgDailyTemp
Algiers	Africa	Algeria	1	1	1996	67.4
Algiers	Africa	Algeria	1	2	1996	60.0
Algiers	Africa	Algeria	1	3	1996	54.4
Algiers	Africa	Algeria	1	4	1996	57.7
Algiers	Africa	Algeria	1	5	1996	57.6
...	...	...	...	...	...	...
Beirut	Middle East	Lebanon	5	9	2020	70.4
Beirut	Middle East	Lebanon	5	10	2020	68.5
Beirut	Middle East	Lebanon	5	11	2020	68.7
Beirut	Middle East	Lebanon	5	12	2020	71.5
Beirut	Middle East	Lebanon	5	13	2020	67.8

952987 rows × 6 columns

Transformation 6: Adding datetime column

In [31]:
```
# Adds date column by combining Month, Day, and Year columns using to_datetime method
temperature_data['Date'] = pd.to_datetime(temperature_data[['Month', 'Day', 'Year']])
temperature_data
```

```
C:\Users\kayly\AppData\Local\Temp\ipykernel_15608\437200125.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  temperature_data['Date'] = pd.to_datetime(temperature_data[['Month', 'Day', 'Year']])
```

Out[31]:

City	Region	Country	Month	Day	Year	AvgDailyTemp	Date
Algiers	Africa	Algeria	1	1	1996	67.4	1996-01-01
Algiers	Africa	Algeria	1	2	1996	60.0	1996-01-02
Algiers	Africa	Algeria	1	3	1996	54.4	1996-01-03
Algiers	Africa	Algeria	1	4	1996	57.7	1996-01-04
Algiers	Africa	Algeria	1	5	1996	57.6	1996-01-05
...	...	...	...	...	...	...	...
Beirut	Middle East	Lebanon	5	9	2020	70.4	2020-05-09
Beirut	Middle East	Lebanon	5	10	2020	68.5	2020-05-10
Beirut	Middle East	Lebanon	5	11	2020	68.7	2020-05-11
Beirut	Middle East	Lebanon	5	12	2020	71.5	2020-05-12
Beirut	Middle East	Lebanon	5	13	2020	67.8	2020-05-13

952987 rows × 7 columns

Transformation 7: Calculate average monthly temperature for each city

For later analysis, it may be useful to see the average monthly temperature of each city.

In [32]:
```
# Group values by City, Month, and Year
temperature_subset_month = temperature_data.groupby(['City','Month', 'Year'])
```

In [33]:
```
# Find the average temperature in each City for each Month
avg_monthly_temp = temperature_subset_month.mean(['AvgDailyTemp'])
avg_monthly_temp
```

Out[33]:

City	Month	Year	Day	AvgDailyTemp
Abidjan	**1**	**1996**	16.000000	80.754839
		1997	16.000000	80.483871
		1998	16.000000	81.145161
		1999	16.200000	81.286667
		2000	15.733333	80.456667
...	**...**	**...**	...	...
Zurich	**12**	**2015**	15.500000	40.020000
		2016	16.000000	33.254839
		2017	16.000000	34.364516
		2018	16.000000	38.219355
		2019	16.000000	38.848387

32404 rows × 2 columns

In [34]:
```
# Change column title from AvgDailyTemp to AvgMonthlyTemp
avg_monthly_temp = avg_monthly_temp.rename(columns={"AvgDailyTemp":"AvgMonthlyTemp"})
avg_monthly_temp
```

Out[34]:

City	Month	Year	Day	AvgMonthlyTemp
Abidjan	**1**	**1996**	16.000000	80.754839
		1997	16.000000	80.483871
		1998	16.000000	81.145161
		1999	16.200000	81.286667
		2000	15.733333	80.456667
...	**...**	**...**	...	...
Zurich	**12**	**2015**	15.500000	40.020000
		2016	16.000000	33.254839
		2017	16.000000	34.364516
		2018	16.000000	38.219355
		2019	16.000000	38.848387

32404 rows × 2 columns

In [35]:
```
# Drop Day column
avg_monthly_temp = avg_monthly_temp.drop(columns='Day')
avg_monthly_temp
```

Out[35]:

City	Month	Year	AvgMonthlyTemp
Abidjan	1	1996	80.754839
		1997	80.483871
		1998	81.145161
		1999	81.286667
		2000	80.456667
...	...	...	...
Zurich	12	2015	40.020000
		2016	33.254839
		2017	34.364516
		2018	38.219355
		2019	38.848387

32404 rows × 1 columns

Transformation 8: Calculate average yearly temperature for each city

For later analysis, it may be useful to see the average yearly temperature of each city.

In [36]:
```
temperature_sebset_year = temperature_data.groupby(['City', 'Year'])
AvgYearlyTemp = temperature_sebset_year.mean(['AvgDailyTemp'])
AvgYearlyTemp
```

Out[36]:

City	Year	Month	Day	AvgDailyTemp
Abidjan	1996	6.498623	15.757576	80.513774
	1997	6.575419	15.768156	80.284637
	1998	6.465374	15.590028	81.113573
	1999	6.150307	15.503067	80.573006
	2000	6.541547	15.974212	79.888825
...	...	...	...	...
Zurich	2016	6.508287	15.801105	49.890884
	2017	6.526027	15.720548	50.350685
	2018	6.497222	15.752778	52.093889
	2019	6.565460	15.657382	51.086908
	2020	2.738806	14.798507	45.263433

2797 rows × 3 columns

In [37]:
```
# Change column title from AvgDailyTemp to AvgYearlyTemp
AvgYearlyTemp = AvgYearlyTemp.rename(columns={"AvgDailyTemp":"AvgYearlyTemp"})
AvgYearlyTemp
```

Out[37]:

City	Year	Month	Day	AvgYearlyTemp
Abidjan	1996	6.498623	15.757576	80.513774
	1997	6.575419	15.768156	80.284637
	1998	6.465374	15.590028	81.113573
	1999	6.150307	15.503067	80.573006
	2000	6.541547	15.974212	79.888825
...	...	...	...	...
Zurich	2016	6.508287	15.801105	49.890884
	2017	6.526027	15.720548	50.350685
	2018	6.497222	15.752778	52.093889
	2019	6.565460	15.657382	51.086908
	2020	2.738806	14.798507	45.263433

2797 rows × 3 columns

In [38]:
```
# Drop Month and Day columns
AvgYearlyTemp = AvgYearlyTemp.drop(columns=['Day','Month'])
AvgYearlyTemp
```

Out[38]:

City	Year	AvgYearlyTemp
Abidjan	1996	80.513774
	1997	80.284637
	1998	81.113573
	1999	80.573006
	2000	79.888825
...	...	...
Zurich	2016	49.890884
	2017	50.350685
	2018	52.093889
	2019	51.086908
	2020	45.263433

2797 rows × 1 columns

I was able to calculate averge monthly temperature and average yearly temperature by city, but I couldn't figure out how to join it back to the original dataset. I need to continue working on this to have a fully finalized dataset. However, my first 6 transformations are shown in the final dataset.

Final Dataset

In [39]:
```
# Prints a human-readable version of my final dataset
temperature_data
```

Out[39]:

City	Region	Country	Month	Day	Year	AvgDailyTemp	Date
Algiers	Africa	Algeria	1	1	1996	67.4	1996-01-01
Algiers	Africa	Algeria	1	2	1996	60.0	1996-01-02
Algiers	Africa	Algeria	1	3	1996	54.4	1996-01-03
Algiers	Africa	Algeria	1	4	1996	57.7	1996-01-04
Algiers	Africa	Algeria	1	5	1996	57.6	1996-01-05
...	...	...	...	...	...	...	...
Beirut	Middle East	Lebanon	5	9	2020	70.4	2020-05-09
Beirut	Middle East	Lebanon	5	10	2020	68.5	2020-05-10
Beirut	Middle East	Lebanon	5	11	2020	68.7	2020-05-11
Beirut	Middle East	Lebanon	5	12	2020	71.5	2020-05-12
Beirut	Middle East	Lebanon	5	13	2020	67.8	2020-05-13

952987 rows × 7 columns

In [25]:
```
temperature_data['City'].unique()
```

Out[25]:
```
array(['Algiers', 'Bujumbura', 'Cotonou', 'Bangui', 'Brazzaville',
       'Cairo', 'Addis Ababa', 'Libreville', 'Banjul', 'Conakry',
       'Bissau', 'Abidjan', 'Nairobi', 'Rabat', 'Antananarivo',
       'Nouakchott', 'Lilongwe', 'Maputo', 'Windhoek', 'Niamey', 'Lagos',
       'Dakar', 'Freetown', 'Capetown', 'Lome', 'Tunis', 'Dar Es Salaam',
       'Kampala', 'Lusaka', 'Dhaka', 'Beijing', 'Chengdu', 'Guangzhou',
       'Shanghai', 'Shenyang', 'Hong Kong', 'Bombay (Mumbai)', 'Calcutta',
       'Chennai (Madras)', 'Delhi', 'Jakarta', 'Osaka', 'Sapporo',
       'Tokyo', 'Almaty', 'Bishkek', 'Vientiane', 'Kuala Lumpur',
       'Ulan-bator', 'Rangoon', 'Katmandu', 'Pyongyang', 'Islamabad',
       'Karachi', 'Manila', 'Singapore', 'Seoul', 'Colombo', 'Taipei',
       'Dusanbe', 'Bangkok', 'Ashabad', 'Tashkent', 'Hanoi', 'Brisbane',
       'Canberra', 'Melbourne', 'Perth', 'Sydney', 'Auckland', 'Tirana',
       'Vienna', 'Minsk', 'Brussels', 'Sofia', 'Zagreb', 'Nicosia',
       'Prague', 'Copenhagen', 'Helsinki', 'Paris', 'Bordeaux', 'Bonn',
       'Frankfurt', 'Hamburg', 'Munich', 'Tbilisi', 'Athens', 'Budapest',
       'Reykjavik', 'Dublin', 'Milan', 'Rome', 'Riga', 'Skopje',
       'Amsterdam', 'Oslo', 'Warsaw', 'Lisbon', 'Bucharest', 'Moscow',
       'Yerevan', 'Pristina', 'Bratislava', 'Barcelona', 'Bilbao',
       'Madrid', 'Stockholm', 'Bern', 'Geneva', 'Zurich', 'Kiev',
       'Belfast', 'London', 'Belgrade', 'Manama', 'Tel Aviv', 'Amman',
       'Kuwait', 'Beirut'], dtype=object)
```

Joining Datasets

In [18]:
```
AvgYearlyTemp
```

Out[18]:

City	Year	AvgYearlyTemp
Abidjan	**1996**	80.513774
	1997	80.284637
	1998	81.113573
	1999	80.573006
	2000	79.888825
...	**...**	...
Zurich	**2016**	49.890884
	2017	50.350685
	2018	52.093889
	2019	51.086908
	2020	45.263433

2797 rows × 1 columns

In [40]:
```
temperature_data
```

Out[40]:

City	Region	Country	Month	Day	Year	AvgDailyTemp	Date
Algiers	Africa	Algeria	1	1	1996	67.4	1996-01-01
Algiers	Africa	Algeria	1	2	1996	60.0	1996-01-02
Algiers	Africa	Algeria	1	3	1996	54.4	1996-01-03
Algiers	Africa	Algeria	1	4	1996	57.7	1996-01-04
Algiers	Africa	Algeria	1	5	1996	57.6	1996-01-05
...	...	...	...	...	...	...	...
Beirut	Middle East	Lebanon	5	9	2020	70.4	2020-05-09
Beirut	Middle East	Lebanon	5	10	2020	68.5	2020-05-10
Beirut	Middle East	Lebanon	5	11	2020	68.7	2020-05-11
Beirut	Middle East	Lebanon	5	12	2020	71.5	2020-05-12
Beirut	Middle East	Lebanon	5	13	2020	67.8	2020-05-13

952987 rows × 7 columns

In [45]:
```
# Resetting index of dataframe
temperature_data = temperature_data.reset_index()
```

```
temperature_data
```

Out[45]:

	City	Region	Country	Month	Day	Year	AvgDailyTemp	Date
0	Algiers	Africa	Algeria	1	1	1996	67.4	1996-01-01
1	Algiers	Africa	Algeria	1	2	1996	60.0	1996-01-02
2	Algiers	Africa	Algeria	1	3	1996	54.4	1996-01-03
3	Algiers	Africa	Algeria	1	4	1996	57.7	1996-01-04
4	Algiers	Africa	Algeria	1	5	1996	57.6	1996-01-05
...	...	...	...	...	...	...	...	...
952982	Beirut	Middle East	Lebanon	5	9	2020	70.4	2020-05-09
952983	Beirut	Middle East	Lebanon	5	10	2020	68.5	2020-05-10
952984	Beirut	Middle East	Lebanon	5	11	2020	68.7	2020-05-11
952985	Beirut	Middle East	Lebanon	5	12	2020	71.5	2020-05-12
952986	Beirut	Middle East	Lebanon	5	13	2020	67.8	2020-05-13

952987 rows × 8 columns

In [47]:
```
# Setting index as City and
temperature_data.set_index(['City', 'Year'], inplace=True)
temperature_data
```

In [52]:
```
# Join temperature_data and AvgYearlyTemp dataframes on City and Year
temperature_data_join1 = temperature_data.join(AvgYearlyTemp)
temperature_data_join1
```

Out[52]:

		Region	Country	Month	Day	AvgDailyTemp	Date	AvgYearlyTemp
City	**Year**							
Abidjan	**1996**	Africa	Ivory Coast	1	1	79.6	1996-01-01	80.513774
	1996	Africa	Ivory Coast	1	2	81.2	1996-01-02	80.513774
	1996	Africa	Ivory Coast	1	3	82.2	1996-01-03	80.513774
	1996	Africa	Ivory Coast	1	4	83.0	1996-01-04	80.513774
	1996	Africa	Ivory Coast	1	5	82.1	1996-01-05	80.513774
...	**...**	...	...	...	...	...	...	...
Zurich	**2020**	Europe	Switzerland	5	9	67.1	2020-05-09	45.263433
	2020	Europe	Switzerland	5	10	64.7	2020-05-10	45.263433
	2020	Europe	Switzerland	5	11	52.0	2020-05-11	45.263433
	2020	Europe	Switzerland	5	12	43.5	2020-05-12	45.263433
	2020	Europe	Switzerland	5	13	44.6	2020-05-13	45.263433

952987 rows × 7 columns

In [53]:
```
# Resetting index
temperature_data_join1 = temperature_data_join1.reset_index()
temperature_data_join1
```

Out[53]:

	City	Year	Region	Country	Month	Day	AvgDailyTemp	Date	AvgYearlyTemp
0	Abidjan	1996	Africa	Ivory Coast	1	1	79.6	1996-01-01	80.513774
1	Abidjan	1996	Africa	Ivory Coast	1	2	81.2	1996-01-02	80.513774
2	Abidjan	1996	Africa	Ivory Coast	1	3	82.2	1996-01-03	80.513774
3	Abidjan	1996	Africa	Ivory Coast	1	4	83.0	1996-01-04	80.513774
4	Abidjan	1996	Africa	Ivory Coast	1	5	82.1	1996-01-05	80.513774
...	...	...	...	...	...	...	...	...	...
952982	Zurich	2020	Europe	Switzerland	5	9	67.1	2020-05-09	45.263433
952983	Zurich	2020	Europe	Switzerland	5	10	64.7	2020-05-10	45.263433
952984	Zurich	2020	Europe	Switzerland	5	11	52.0	2020-05-11	45.263433
952985	Zurich	2020	Europe	Switzerland	5	12	43.5	2020-05-12	45.263433
952986	Zurich	2020	Europe	Switzerland	5	13	44.6	2020-05-13	45.263433

952987 rows × 9 columns

In [54]:
```
temperature_data_join1.set_index(['City', 'Month', 'Year'], inplace=True)
temperature_data_join1
```

Out[54]:

			Region	Country	Day	AvgDailyTemp	Date	AvgYearlyTemp
City	**Month**	**Year**						
Abidjan	**1**	**1996**	Africa	Ivory Coast	1	79.6	1996-01-01	80.513774
		1996	Africa	Ivory Coast	2	81.2	1996-01-02	80.513774
		1996	Africa	Ivory Coast	3	82.2	1996-01-03	80.513774
		1996	Africa	Ivory Coast	4	83.0	1996-01-04	80.513774
		1996	Africa	Ivory Coast	5	82.1	1996-01-05	80.513774
...	**...**	**...**	...	...	...	...	...	...
Zurich	**5**	**2020**	Europe	Switzerland	9	67.1	2020-05-09	45.263433
		2020	Europe	Switzerland	10	64.7	2020-05-10	45.263433
		2020	Europe	Switzerland	11	52.0	2020-05-11	45.263433
		2020	Europe	Switzerland	12	43.5	2020-05-12	45.263433
		2020	Europe	Switzerland	13	44.6	2020-05-13	45.263433

952987 rows × 6 columns

In [55]:
```
temperature_data_join2 = temperature_data_join1.join(avg_monthly_temp)
temperature_data_join2
```

Out[55]:

City	Month	Year	Region	Country	Day	AvgDailyTemp	Date	AvgYearlyTemp	AvgMonthlyTemp
Abidjan	**1**	**1996**	Africa	Ivory Coast	1	79.6	1996-01-01	80.513774	80.754839
		1996	Africa	Ivory Coast	2	81.2	1996-01-02	80.513774	80.754839
		1996	Africa	Ivory Coast	3	82.2	1996-01-03	80.513774	80.754839
		1996	Africa	Ivory Coast	4	83.0	1996-01-04	80.513774	80.754839
		1996	Africa	Ivory Coast	5	82.1	1996-01-05	80.513774	80.754839
...	**...**	**...**	...	...	...	...	...	...	...
Zurich	**12**	**2019**	Europe	Switzerland	27	40.8	2019-12-27	51.086908	38.848387
		2019	Europe	Switzerland	28	35.5	2019-12-28	51.086908	38.848387
		2019	Europe	Switzerland	29	30.4	2019-12-29	51.086908	38.848387
		2019	Europe	Switzerland	30	29.9	2019-12-30	51.086908	38.848387
		2019	Europe	Switzerland	31	31.4	2019-12-31	51.086908	38.848387

952987 rows × 7 columns

In [57]:
```
temperature_data_join2.rename(columns={'Date':'Date of Observation'}, inplace=True)
temperature_data_join2
```

Out[57]:

City	Month	Year	Region	Country	Day	AvgDailyTemp	Date of Observation	AvgYearlyTemp	AvgMonthlyTemp
Abidjan	**1**	**1996**	Africa	Ivory Coast	1	79.6	1996-01-01	80.513774	80.754839
		1996	Africa	Ivory Coast	2	81.2	1996-01-02	80.513774	80.754839
		1996	Africa	Ivory Coast	3	82.2	1996-01-03	80.513774	80.754839
		1996	Africa	Ivory Coast	4	83.0	1996-01-04	80.513774	80.754839
		1996	Africa	Ivory Coast	5	82.1	1996-01-05	80.513774	80.754839
...	**...**	**...**	...	...	...	...	...	...	...
Zurich	**12**	**2019**	Europe	Switzerland	27	40.8	2019-12-27	51.086908	38.848387
		2019	Europe	Switzerland	28	35.5	2019-12-28	51.086908	38.848387
		2019	Europe	Switzerland	29	30.4	2019-12-29	51.086908	38.848387
		2019	Europe	Switzerland	30	29.9	2019-12-30	51.086908	38.848387
		2019	Europe	Switzerland	31	31.4	2019-12-31	51.086908	38.848387

952987 rows × 7 columns

Final Dataset

In [60]:
```
temperature_data_join2 = temperature_data_join2.reset_index()
temperature_data_join2.set_index('City', inplace=True)
temperature_data_join2
```

Out[60]:

City	index	Country	Month	Year	Region	Day	AvgDailyTemp	Date of Observation	AvgYearlyTemp	AvgMonthlyTemp
Abidjan	0	Ivory Coast	1	1996	Africa	1	79.6	1996-01-01	80.513774	80.754839
Abidjan	1	Ivory Coast	1	1996	Africa	2	81.2	1996-01-02	80.513774	80.754839
Abidjan	2	Ivory Coast	1	1996	Africa	3	82.2	1996-01-03	80.513774	80.754839
Abidjan	3	Ivory Coast	1	1996	Africa	4	83.0	1996-01-04	80.513774	80.754839
Abidjan	4	Ivory Coast	1	1996	Africa	5	82.1	1996-01-05	80.513774	80.754839
...	...	...	...	...	...	...	...	...	...	...
Zurich	952982	Switzerland	12	2019	Europe	27	40.8	2019-12-27	51.086908	38.848387
Zurich	952983	Switzerland	12	2019	Europe	28	35.5	2019-12-28	51.086908	38.848387
Zurich	952984	Switzerland	12	2019	Europe	29	30.4	2019-12-29	51.086908	38.848387
Zurich	952985	Switzerland	12	2019	Europe	30	29.9	2019-12-30	51.086908	38.848387
Zurich	952986	Switzerland	12	2019	Europe	31	31.4	2019-12-31	51.086908	38.848387

952987 rows × 10 columns

In [63]:
```
temperature_data_join2 = temperature_data_join2.drop(columns='index')
temperature_data_join2
```

Out[63]:

City	Country	Month	Year	Region	Day	AvgDailyTemp	Date of Observation	AvgYearlyTemp	AvgMonthlyTemp
Abidjan	Ivory Coast	1	1996	Africa	1	79.6	1996-01-01	80.513774	80.754839
Abidjan	Ivory Coast	1	1996	Africa	2	81.2	1996-01-02	80.513774	80.754839
Abidjan	Ivory Coast	1	1996	Africa	3	82.2	1996-01-03	80.513774	80.754839
Abidjan	Ivory Coast	1	1996	Africa	4	83.0	1996-01-04	80.513774	80.754839
Abidjan	Ivory Coast	1	1996	Africa	5	82.1	1996-01-05	80.513774	80.754839
...	...	...	...	...	...	...	...	...	...
Zurich	Switzerland	12	2019	Europe	27	40.8	2019-12-27	51.086908	38.848387
Zurich	Switzerland	12	2019	Europe	28	35.5	2019-12-28	51.086908	38.848387
Zurich	Switzerland	12	2019	Europe	29	30.4	2019-12-29	51.086908	38.848387
Zurich	Switzerland	12	2019	Europe	30	29.9	2019-12-30	51.086908	38.848387
Zurich	Switzerland	12	2019	Europe	31	31.4	2019-12-31	51.086908	38.848387

952987 rows × 9 columns

In [68]:
```
temperature_data_join2 = temperature_data_join2.rename(columns={'Date of Observation':'DateOfObservation'})
temperature_data_join2
```

Out[68]:

City	Country	Month	Year	Region	Day	AvgDailyTemp	DateOfObservation	AvgYearlyTemp	AvgMonthlyTemp
Abidjan	Ivory Coast	1	1996	Africa	1	79.6	1996-01-01	80.513774	80.754839
Abidjan	Ivory Coast	1	1996	Africa	2	81.2	1996-01-02	80.513774	80.754839
Abidjan	Ivory Coast	1	1996	Africa	3	82.2	1996-01-03	80.513774	80.754839
Abidjan	Ivory Coast	1	1996	Africa	4	83.0	1996-01-04	80.513774	80.754839
Abidjan	Ivory Coast	1	1996	Africa	5	82.1	1996-01-05	80.513774	80.754839
...	...	...	...	...	...	...	...	...	...
Zurich	Switzerland	12	2019	Europe	27	40.8	2019-12-27	51.086908	38.848387
Zurich	Switzerland	12	2019	Europe	28	35.5	2019-12-28	51.086908	38.848387
Zurich	Switzerland	12	2019	Europe	29	30.4	2019-12-29	51.086908	38.848387
Zurich	Switzerland	12	2019	Europe	30	29.9	2019-12-30	51.086908	38.848387
Zurich	Switzerland	12	2019	Europe	31	31.4	2019-12-31	51.086908	38.848387

952987 rows × 9 columns

In [ ]:

Writing to CSV file

In [69]:
```
# Writing dataframe to a csv file
temperature_data_join2.to_csv('TemperatureData', sep=',', encoding='utf-8', index=True)
```

In [70]:
```
# Checking that writing to file worked correctly
csvFile = pd.read_csv("C:/Users/kayly/OneDrive/Desktop/MSDS/DSC540/Tem Project/TemperatureData")
csvFile
```

Out[70]:

	City	Country	Month	Year	Region	Day	AvgDailyTemp	DateOfObservation	AvgYearlyTemp	AvgMonthlyTemp
0	Abidjan	Ivory Coast	1	1996	Africa	1	79.6	1996-01-01	80.513774	80.754839
1	Abidjan	Ivory Coast	1	1996	Africa	2	81.2	1996-01-02	80.513774	80.754839
2	Abidjan	Ivory Coast	1	1996	Africa	3	82.2	1996-01-03	80.513774	80.754839
3	Abidjan	Ivory Coast	1	1996	Africa	4	83.0	1996-01-04	80.513774	80.754839
4	Abidjan	Ivory Coast	1	1996	Africa	5	82.1	1996-01-05	80.513774	80.754839
...	...	...	...	...	...	...	...	...	...	...
952982	Zurich	Switzerland	12	2019	Europe	27	40.8	2019-12-27	51.086908	38.848387
952983	Zurich	Switzerland	12	2019	Europe	28	35.5	2019-12-28	51.086908	38.848387
952984	Zurich	Switzerland	12	2019	Europe	29	30.4	2019-12-29	51.086908	38.848387
952985	Zurich	Switzerland	12	2019	Europe	30	29.9	2019-12-30	51.086908	38.848387
952986	Zurich	Switzerland	12	2019	Europe	31	31.4	2019-12-31	51.086908	38.848387

952987 rows × 10 columns

Ethical Implications

I made numerous changes to the data. To begin I set the index to City. I then dropped the State column because it contained only Na values. I also removed rows that had -99 for the value of AvgTemp. I believe these are meant to code for Na values. I also removed a few rows that contain implausabe years. These were likely entered incorrectly. I added a column containg date-time information to make serching easier. Finally, I caclulated both the average montly temperature and average yearly temperature by city.

There is no legal or regulatory guidelines for my data or project topic. In this dataset, I am simply looking for changes in average temperature of a city over time.

I do not see many risks in the transformations I made. I only excluded data that was obviously incorrect. I did not assume I knew what years were meant to be when years were coded as 200 and 201. Instead, I removed these rows from the dataset. I was pretty conservative in my transformations because I wanted to maintain the integrity of the original dataset while fixing a few imperfections.

I sourced the data from Kaggle where it was posted and collected by the University of Dayton. I do not have any background on how the data was collected, or the original sources of the data. Without any further background information, I just have to trust that the values provided are accurate. Ethically, this is shakey ground. I cannot be sure that any conclusions I make are 100% sound without knowing how the original data was sourced and collected. I must be careful not to draw conclusions from this dataset alone. In combination with other datasets that are more easily verified and more credible, I may confirm assumptions drawn from this dataset.