Potential Expansion Based on Temperature Changes

Emma Mantel, Michael Airhart, Zabriel Adams, Bryan Oyuela

Business Objective

As a small manufacturer of Air Conditioning units, we are turning to this dataset to explore five domestic and five international cities that could potentially represent a new market given their temperature increases over the last 25 years.

The Dataset

Demographic: 154 US cities and 167 international cities from 124 countries

Years spanned: 1995 - 2020 (partially)

region	country	state	city	month	day	year	avgtemperature
Africa	Algeria	NaN	Algiers	1	1	1995	64.2
Middle East	Turkey	NaN	Ankara	1	14	1995	41.2
...	...	...	...	...	...	...	...
North America	US	Washington	Spokane	1	18	2020	26.9
North America	US	Washington	Spokane	1	19	2020	32.8
2826652 rows × 8 columns							

Data Source: http://academic.udayton.edu/kissock/http/Weather/default.htm

Data Cleaning

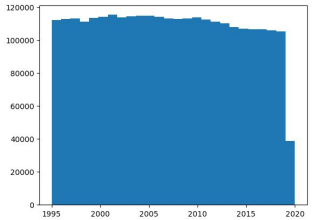
3 main faults in the data:

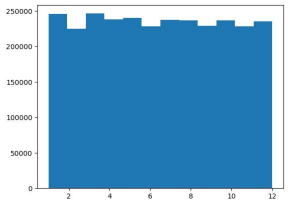
- Some temperatures were set to **“-99”**, which may have accounted for missing data
- Some of those **“-99”** days also had the year incorrectly entered as **“201”**
- Certain days were indicated as day **“0”**

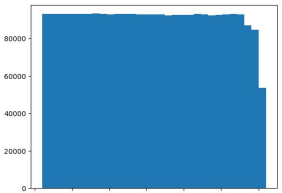
These data were removed, resulting in **79,672** out of **2,906,327** rows removed **(2.74%)**.

Lastly, the data for **2020** is <u>only</u> included up to and including the month of **May**.

Robustness: through histograms of each date category we were able to confirm that no single day, month, or year was grossly over- or underrepresented







Procedure

To get a more robust measure:

- Took the average of the **first 5 years** and the averages of the **last 5 years** in the dataset. Then, we obtained an average temperature increase for the summer months by taking the difference
- Ranked cities by **largest average temperature increase**
- Selected **10 cities** as potential candidates for market expansion (global and domestic)

Results

After ordering the cities **(5 international cities and 5 domestic cities)** by biggest temperature increase, we narrowed our search to the following 10 candidate cities in no particular order:

1. Grand Junction, Colorado
2. Milan, Italy
3. Pueblo, Salt Lake City
4. San Angelo, Texas
5. Denver, Colorado
6. Madrid, Spain
7. Zagreb, Croatia
8. Salt Lake City, Utah
9. Belgrade, Serbia
10. Bishkek, Kyrgyzstan

➢ **Dubai, United Arab Emirates (Was in top 10, but excluded due to extreme heat)**

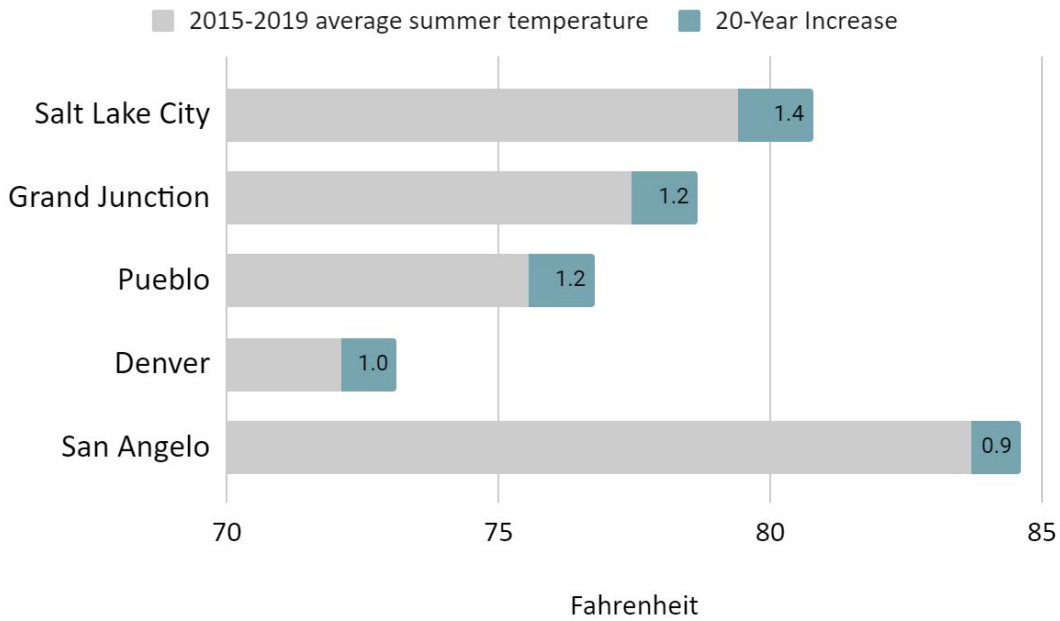
Color Legend
International
Domestic (US)

Temperature Filtering

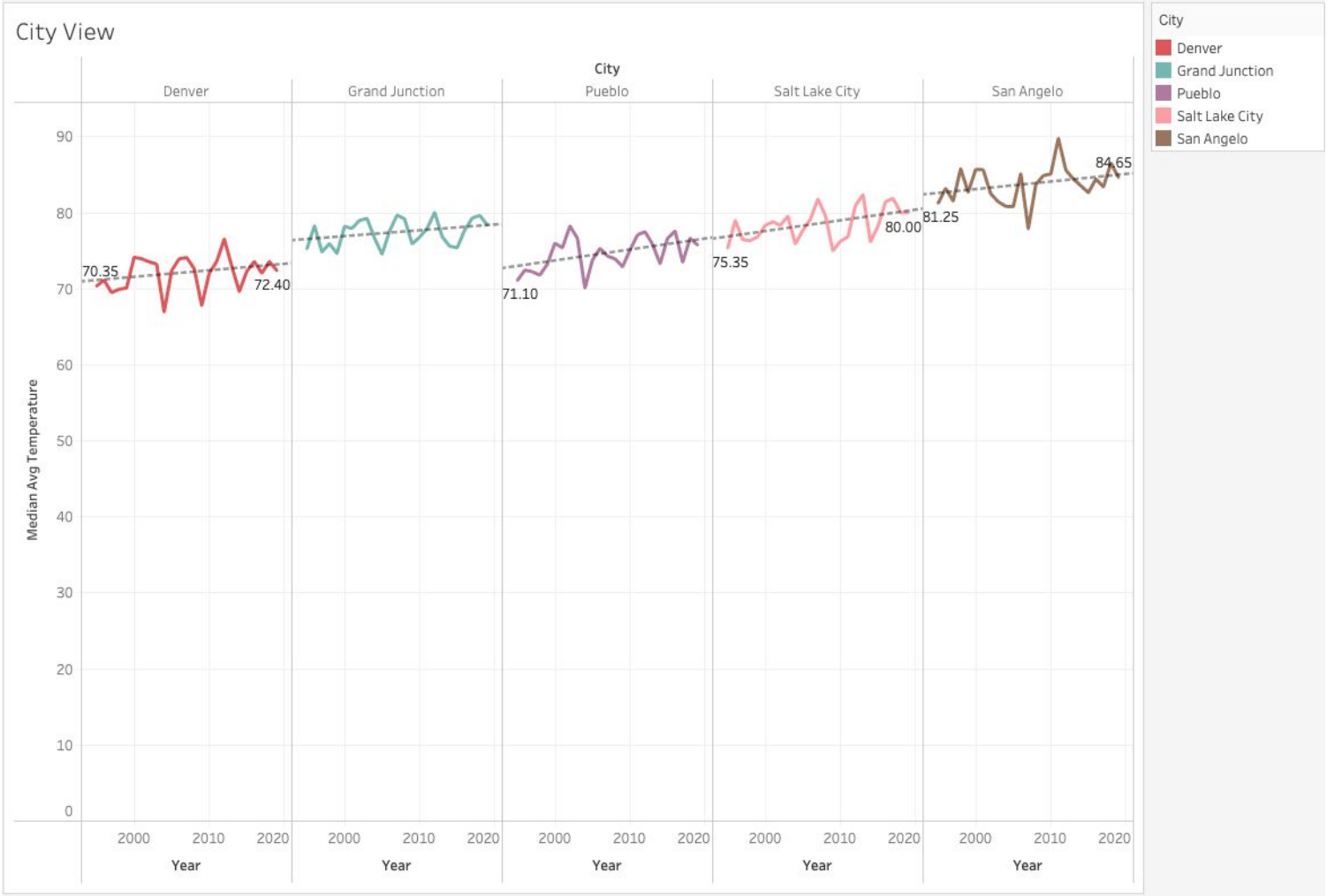
After looking over the list we created, we decided to refine our search further. This includes the following parameters:

- Remove from list if city is relatively cool (Average temperature < 75 degrees)
  - A city that could still be too cool for AC may not help us maximize our sales.
  - **This excludes Denver, Colorado.**
- Remove from list if city is too hot (Average temperature > 85 degrees)
  - A city that is already too hot is more likely to already have a saturated AC market
  - **This excludes Dubai, United Arab Emirates**

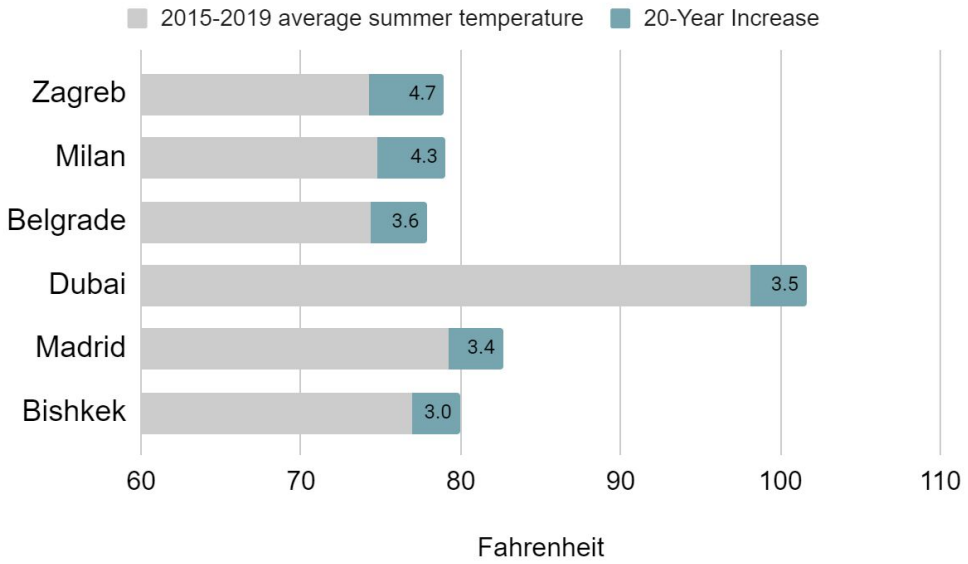
Candidate Domestic Cities



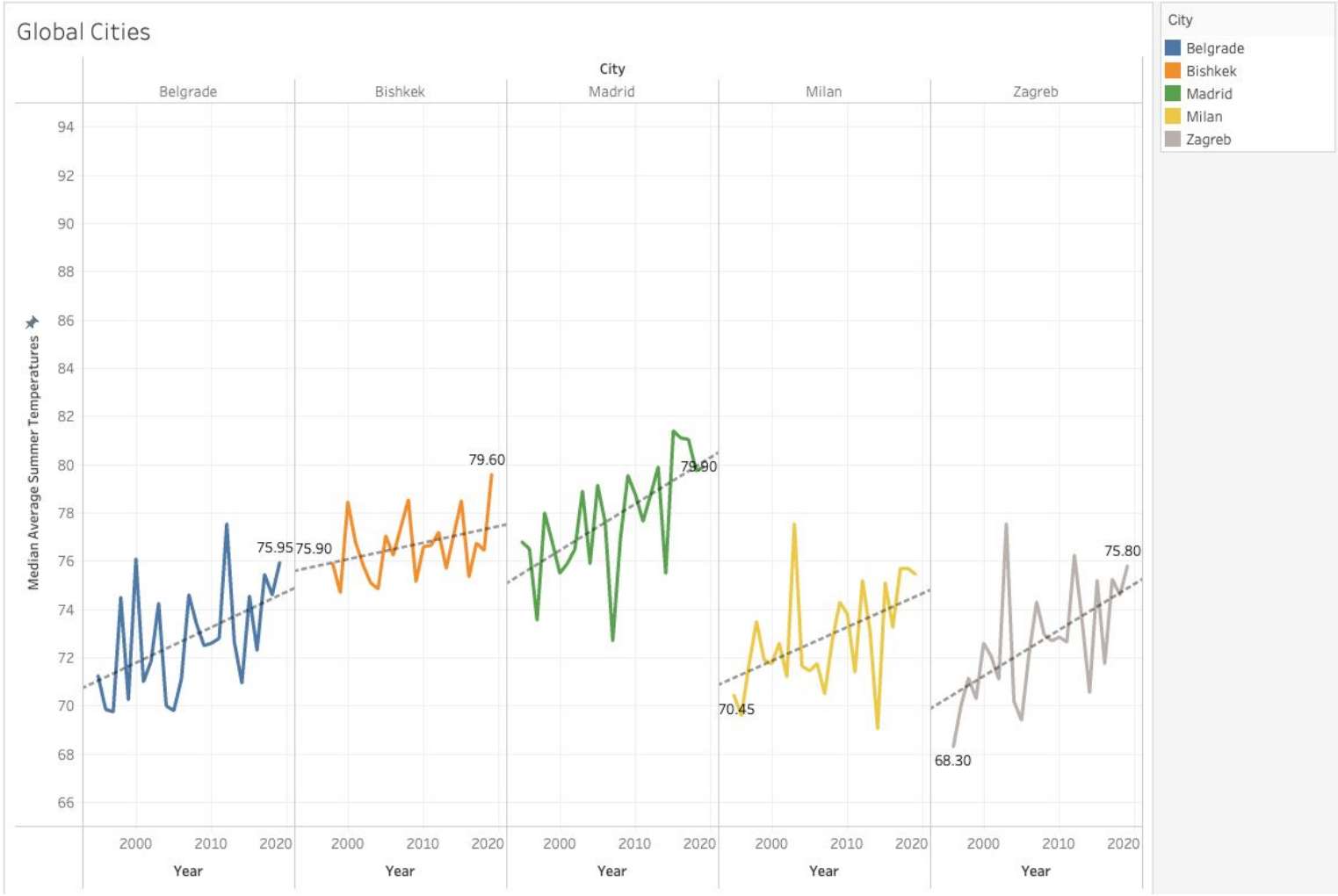
* Denver does not quite meet our temperature requirement


City View
City
Denver
Grand Junction
Pueblo
Salt Lake City
San Angelo
Median Avg Temperature
Year
70.35
72.40
71.10
75.35
80.00
81.25
84.65

Candidate International Cities

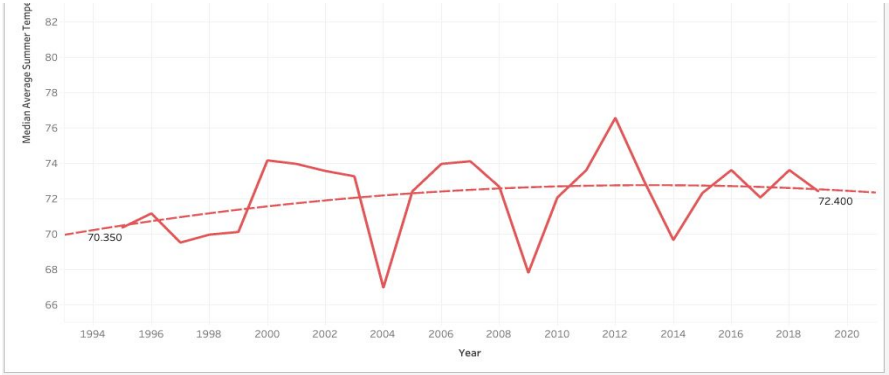


* Dubai was excluded


Global Cities
City
Belgrade
Bishkek
Madrid
Milan
Zagreb
Median Average Summer Temperatures
94
92
90
88
86
84
82
80
78
76
74
72
70
68
66
75.95
75.90
79.60
79.90
70.45
68.30
75.80
2000
2010
2020
Year

Could Denver Fit Our Criteria Eventually?

Temperature Prediction: Linear Regression



Denver is unlikely to reach our threshold in the future.

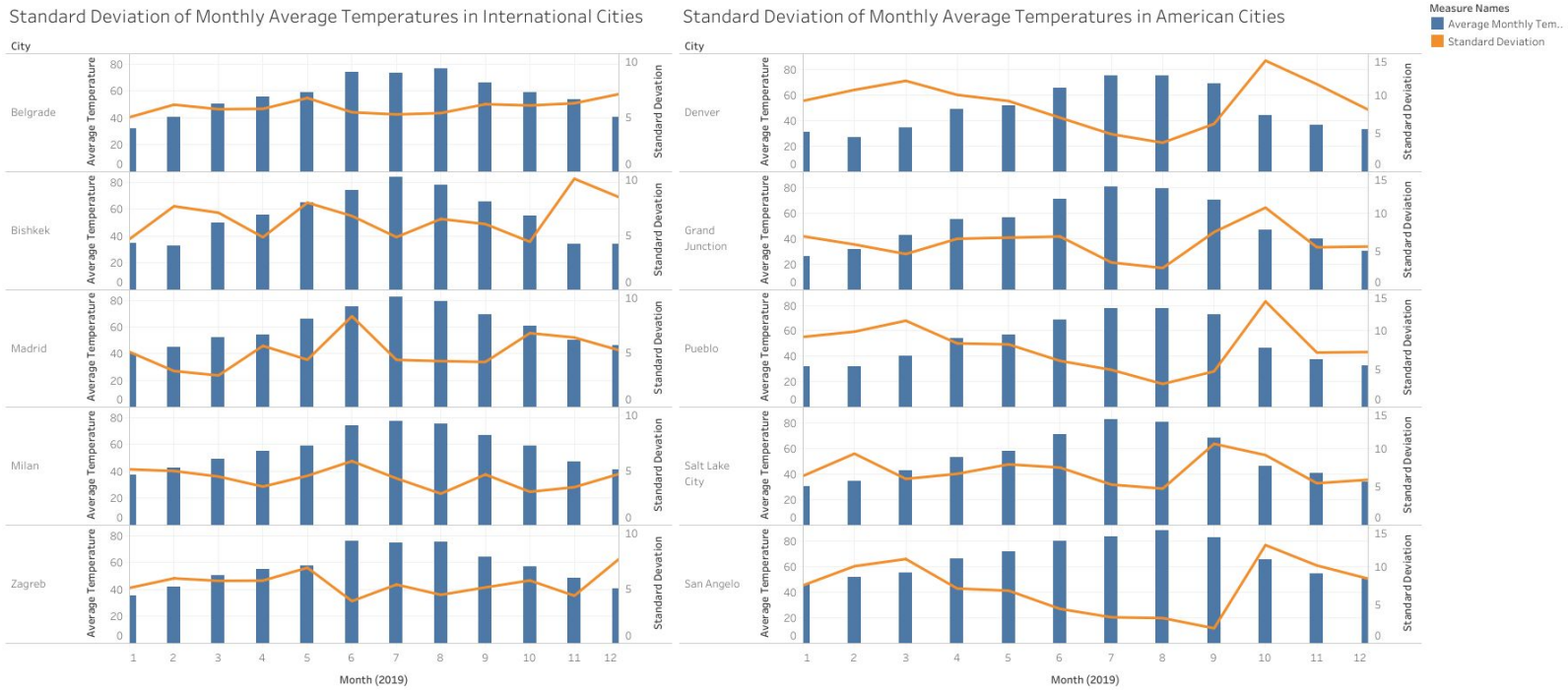
Further Analysis to Narrow the Options

Lots of Options, But Not All Are Feasible

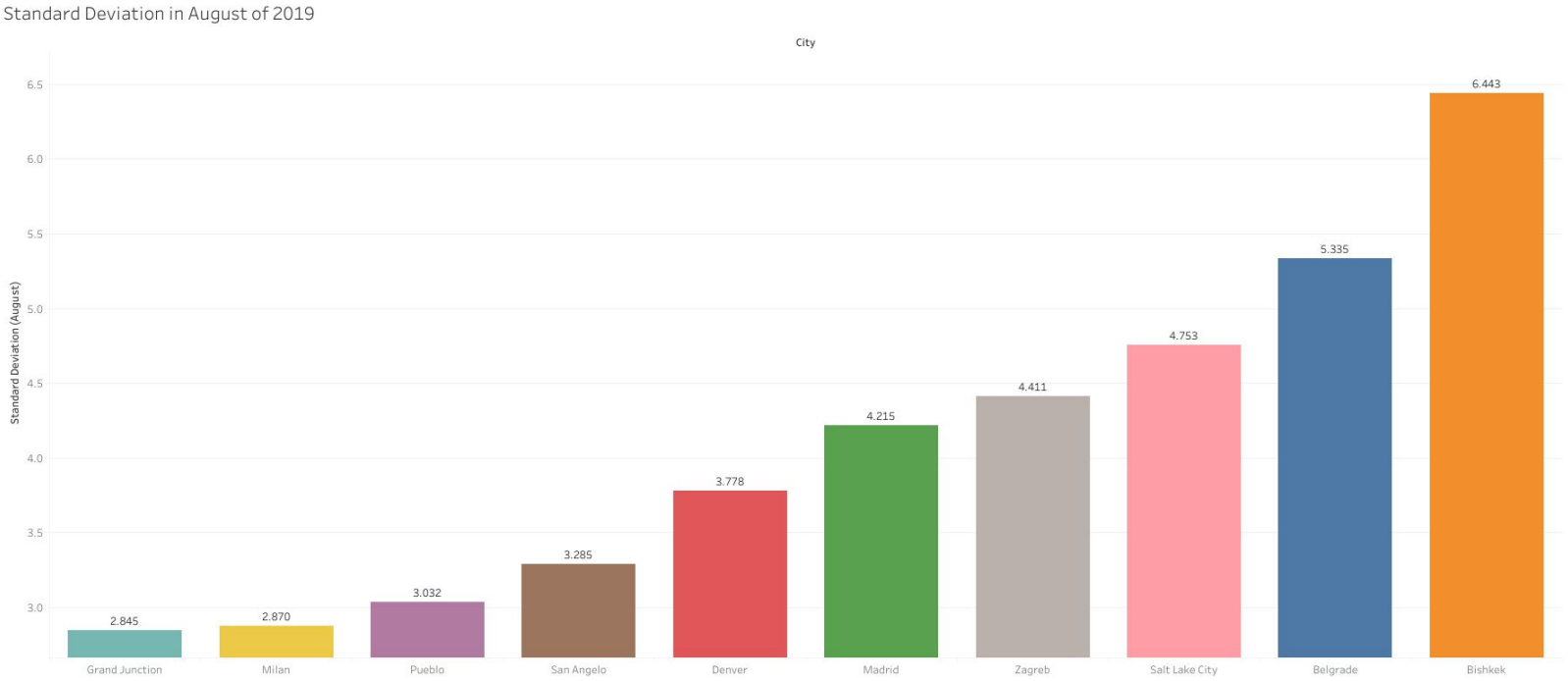
After narrowing our list to ten cities, we realized that expansion to each of these would still be a massive undertaking. To deal with this, we conducted further analysis to find 3 realistic choices for our expansion. We went about this by:

1. Deciding on an additional metric: the standard deviation of the temperature in the summer months
2. Creating combination charts and bar graphs to visualize those metrics
3. Ranking the cities according to those metrics for our final recommendation

Combination Chart



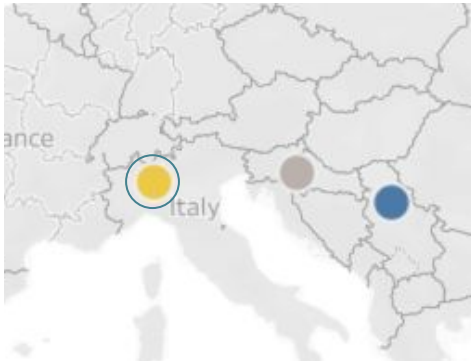
Standard Deviations in Order

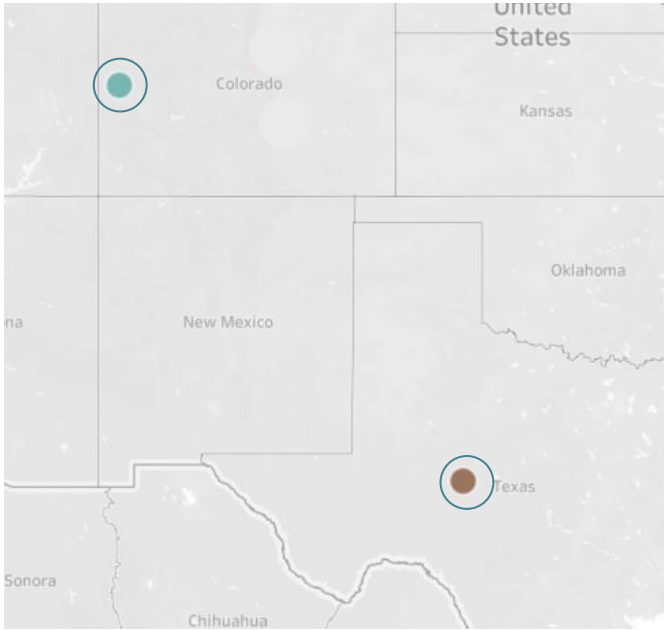


Final Recommendations

The final recommendation based on our temperature analysis is to further explore the market conditions and possible retail partners in 3 key locations:

1. San Angelo, TX
2. Milan, Italy
3. Grand Junction, CO





The following code intends to create a CSV file with data that includes both the monthly average temperatures and the standard deviations of each month.

In [4]:
```
import numpy as np
import pandas as pd

# Read the CSV file into a DataFrame
df = pd.read_csv(r'C:\\Users\\Bryan\\Downloads\\glob_city_temperatures.csv')

# Define data types for columns
data_types = {'City': str, 'Month': int, 'Day': int, 'Year': int, 'AvgTemperature': float}

# Drop unnecessary columns
df = df.drop(['State', 'Region', 'Country'], axis=1)

# Filter the DataFrame for the year 2019
filtered_df = df[(df['Year'] == 2019)]

# Calculate the monthly averages of temperatures for each city
monthly_averages = filtered_df.groupby(['Month','City'])['AvgTemperature'].mean().reset_index()

# Calculate the monthly standard deviations of temperatures for each city
monthly_std = filtered_df.groupby(['Month','City'])['AvgTemperature'].std().reset_index()

# Merge the average and standard deviation DataFrames
combined_df = monthly_averages.merge(monthly_std, on=['Month', 'City'], suffixes=('_avg', '_std'))

# Save the combined DataFrame to a CSV file
combined_df.to_csv('monthly_averages&std.csv', index=False)
```

The following code intends to return the standard deviations in August in ascending order.

The following code intends to return the standard deviations in August in ascending order.

In [15]:
```
import numpy as np
import pandas as pd

# Read the CSV file into a DataFrame
df = pd.read_csv(r'C:\\Users\\Bryan\\Downloads\\glob_city_temperatures.csv')

# Define data types for columns
data_types = {'City': str, 'Month': int, 'Day': int, 'Year': int, 'AvgTemperature': float}

# Drop unnecessary columns
df = df.drop(['State', 'Region', 'Country'], axis=1)

# Filter the DataFrame for the year 2019 and August
filtered_df = df[(df['Year'] == 2019) & (df['Month'].isin([8]))]

# Calculate the monthly averages of temperatures for each city
monthly_averages = filtered_df.groupby(['Month', 'City'])['AvgTemperature'].mean()

# Calculate the monthly standard deviations of temperatures for each city
monthly_std = filtered_df.groupby(['Month', 'City'])['AvgTemperature'].std()

# Sort the standard deviations in ascending order
sorted_std_df = monthly_std.sort_values(ascending=True)

# Print the sorted DataFrame
sorted_std_df
```

Out[15]:
```
Month  City
8      Grand Junction    2.844935
       Milan             2.870405
       Pueblo            3.032243
       San Angelo        3.284506
       Denver            3.777597
       Madrid            4.214532
       Zagreb            4.410776
       Salt Lake City    4.752745
       Belgrade          5.335438
       Bishkek           6.443158
Name: AvgTemperature, dtype: float64
```

The following code was used to create the final dataset we wanted to use. It takes the huge dataset of daily temperatures and narrows the data down to the cities we wanted. It also included a line that checks for empty data values that are marked by 0's or -99's.

In [29]:
```
import pandas as pd

# Specify the file path to the CSV file
file_path = "C:\\Users\\Bryan\\Downloads\\city_temperature.csv"

# Read the CSV file into a DataFrame
df = pd.read_csv(file_path, dtype={'1': str, '2': str, '3': str, '4': str, '5': int, '6': int, '7': int, '8': float},
                 low_memory=False)

# Filter data for specific cities
filtered_df = df[(df['City'].isin(['Grand Junction', 'Pueblo', 'Salt Lake City', 'San Angelo', 'Denver', 'Zagreb', 'Milan',
                                   'Belgrade', 'Madrid', 'Bishkek']))]
#Data cleaning for null values
filtered_df_1 = filtered_df[((filtered_df['Day'] != 0) & (filtered_df['AvgTemperature'] != -99))]

# Specify the file path for the new file
new_file_path = "C:\\Users\\Bryan\\Downloads\\glob_city_temperatures.csv"

# Save the filtered data to a new CSV file
filtered_df_1.to_csv(new_file_path, index=False)
```