DSC540-T301 Data Preparation (2233-1)

Term Project: Final Milestone No.5: Austin TX, Weather in Novemebr

Arash Mahmooudian

Abstract

In this study we have captured weather information for the city of Austin, TX via three different data sources listed below:

- Kaggle Austin weather analysis dataset in CSV format: link: https://www.kaggle.com/datasets/grubenm/austin-weather
- Austin weather through from OpenWeatherMap API in JSON format: link: https://home.openweathermap.org/history_forecast_bulks/new
- Austin weather from Wunderground for the month of Nov-2022 in HTML format: link: https://www.wunderground.com/history/monthly/us/tx/austin/KAUS/date/2022-11

Goal:

Overall aim of this study is to be able to gether data from different data type sources and perform cleaning/transformation and finally store them in a database and perform post-processing with visualized exports.

Steps Taken:

- 1- Download Kaggle dataset through the mentioned link.
- 2- Parse and then export/transform weather information through a web scraping process of the provided link above.
- 3- Hit the openweathermap API and export weather information a from json formatted data.

Finally having all these clean data we filter the weather info for the month of November and store them all in three different tables of SQLite data based, and provide visualized comparative graghs of them.

In [1]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

```
import pandas as pd
import numpy as np
import sqlite3
```

Load already processed datasets

Perform final column name change/transformations

In [2]:
```
df_api = pd.read_csv('Austin_weather_api.csv')
df_kaggle = pd.read_csv('Austin_weather_kaggle_processed.csv')
df_website = pd.read_csv('Austin_weather_html.csv')

print('df_api: ',df_api.shape)
print('df_kaggle: ',df_kaggle.shape)
print('df_html: ',df_website.shape)
```

```
df_api:  (30, 24)
df_kaggle:  (1183, 21)
df_html:  (30, 17)
```

Kaggle dataset clean ups/ trsansformations

df_kaggle is holding data for years 2013 to July 2017, not Nov 2022. For this reason I will extract data for Nov 2014 Which has less missing data (only one day Nov 3th) compared to other yesrs.

Kaggle:

In [3]:
```
# Split 'Data' to year, month and day

df_kaggle['year'] = [str(x).split('/')[2] for x in df_kaggle['Date']]
df_kaggle['day'] = [int(str(x).split('/')[1]) for x in df_kaggle['Date']]
df_kaggle['month'] = [str(x).split('/')[0] for x in df_kaggle['Date']]
#---------------------------------------------------------------------------


#df_kaggle_group = df_kaggle.groupby(['year', 'month'])['Date'].count()

df_kaggle = df_kaggle[(df_kaggle['year'] == '2014') & (df_kaggle['month'] == '11')]  #
df_kaggle['Date'] = df_kaggle['day']                                                 #
df_kaggle.drop(columns=['year','month','day'], inplace = True)                       #
#---------------------------------------------------------------------------

# Add an index of 'k_' to the begining of each attributes
# This because we want to distingush attributes for each datasets.
kaggle_columns = list(df_kaggle.columns)
for i in range(len(kaggle_columns)):
    if (kaggle_columns[i] != 'Date'):
        kaggle_columns[i] = 'k_' + kaggle_columns[i]


kaggle_columns = list(df_kaggle.columns)
for i in range (len(kaggle_columns)):
    kaggle_columns[i] = kaggle_columns[i].lower()

    if (kaggle_columns[i] != 'date'):
        kaggle_columns[i] = 'k_' + kaggle_columns[i]
```

```
    kaggle_columns[i] = kaggle_columns[i].replace('date','Date')
    kaggle_columns[i] = kaggle_columns[i].replace('f','')
    kaggle_columns[i] = kaggle_columns[i].replace('high','_max')
    kaggle_columns[i] = kaggle_columns[i].replace('low','_min')
    kaggle_columns[i] = kaggle_columns[i].replace('avg','_avg')
    kaggle_columns[i] = kaggle_columns[i].replace('percent','')
    kaggle_columns[i] = kaggle_columns[i].replace('sealevel','')
    kaggle_columns[i] = kaggle_columns[i].replace('inches','')
    kaggle_columns[i] = kaggle_columns[i].replace('miles','')
    kaggle_columns[i] = kaggle_columns[i].replace('mph','')
    kaggle_columns[i] = kaggle_columns[i].replace('mph','')
    kaggle_columns[i] = kaggle_columns[i].replace('dewpoint','dew_point')

df_kaggle.columns = kaggle_columns
df_kaggle['Date'] = df_kaggle['Date'].astype(int)
df_kaggle.head()
```

Out[3]:

	Date	k_temp_max	k_temp_avg	k_temp_min	k_dew_point_max	k_dew_point_avg	k_dew_point_n
281	1	68	58	47	43	39	
282	2	75	62	48	52	45	
283	4	75	67	58	69	63	
284	5	59	57	55	58	56	
285	6	63	59	55	56	51	

5 rows × 21 columns

OpenWeathermap called 'api' dataset for simplicity

In [4]:

```
# Apply some column name changes
# Add 'a_' index to beginning of each attributes, and common column as for other datas

api_columns = list(df_api.columns)
for i in range (len(api_columns)):

    if (api_columns[i] != 'date'):
        api_columns[i] = 'a_' + api_columns[i]

    api_columns[i] = api_columns[i].lower()
    api_columns[i] = api_columns[i].replace('mean', 'avg')
    api_columns[i] = api_columns[i].replace('main_', '')
    api_columns[i] = api_columns[i].replace('.1', '')
    api_columns[i] = api_columns[i].replace('date', 'Date')

df_api.columns = api_columns
#-------------------------------------------------------------------

# Split 'Data' to year, month and day
df_api['year'] = [str(x).split('-')[0] for x in df_api['Date']]
df_api['day'] = [int(str(x).split('-')[2]) for x in df_api['Date']]
df_api['month'] = [str(x).split('-')[1] for x in df_api['Date']]
df_api['Date'] = df_api['day']                                    # All datasets wit
```

```
df_api.drop(columns=['year','month','day'], inplace = True)         # Drops unnecessar
#-----------------------------------------------------------------

# Metric conversion pressure to PSI and Visibility to Miles
for index, row in df_api.iterrows():
    pressure = row['a_pressure_min']
    psi = (pressure * 0.0145) *  2.03602
    df_api.loc[index, ['a_pressure_min']] = psi

    pressure = row['a_pressure_avg']
    psi = (pressure * 0.0145) *  2.03602
    df_api.loc[index, ['a_pressure_avg']] = psi

    pressure = row['a_pressure_max']
    psi = (pressure * 0.0145) *  2.03602
    df_api.loc[index, ['a_pressure_max']] = psi


    visibility = row['a_visibility_avg']
    df_api.loc[index, ['a_visibility_avg']] = visibility/1609

df_api.head()
```

Out[4]:

	Date	a_visibility_avg	a_snow_avg	a_wind_speed_min	a_wind_speed_avg	a_wind_speed_max	a_win
0	1	6.206132	0.0	0.00	4.679167	8.99	
1	2	5.920447	0.0	0.00	3.819583	7.00	
2	3	5.837088	0.0	3.44	10.071667	19.57	
3	4	6.063290	0.0	8.05	15.317500	21.85	
4	5	6.122773	0.0	0.00	5.559167	12.66	

5 rows × 24 columns

Website driven data

In [5]:

```
# Apply some column name changes
# Add 'w_' index to beginning of each attributes, and common column as for other datas

df_website_columns = list(df_website.columns)

for i in range (len(df_website_columns)):

    if (df_website_columns[i] != 'Time'):
        df_website_columns[i] = 'w_' + df_website_columns[i]

    df_website_columns[i] = df_website_columns[i].lower()
    df_website_columns[i] = df_website_columns[i].replace(' (°f)','')
    df_website_columns[i] = df_website_columns[i].replace(' (%)','')
    df_website_columns[i] = df_website_columns[i].replace(' (mph)','')
    df_website_columns[i] = df_website_columns[i].replace(' (in)','')
    df_website_columns[i] = df_website_columns[i].replace(' ','_')
    df_website_columns[i] = df_website_columns[i].replace('temperature','temp')
    df_website_columns[i] = df_website_columns[i].replace('time','Date')
```

```
df_website.columns = df_website_columns
#-----------------------------------------------------------------

# Split 'Data' to year, month and day

df_website['month'] = [str(x).split('/')[1]  for x in df_website['Date']]
df_website['day'] = [int(str(x).split('/')[2])   for x in df_website['Date']]
df_website['Date'] = df_website['day']                                    # All
df_website.drop(columns=['month','day'], inplace = True)                  # Drop
df_website.head()
```

Out[5]:

	Date	w_temp_max	w_temp_avg	w_temp_min	w_dew_point_max	w_dew_point_avg	w_dew_point_r
0	2022	66	60.8	55	62	59.5	
1	2022	76	63.6	54	68	60.4	
2	2022	84	74.6	69	70	68.7	
3	2022	85	73.5	59	75	69.0	
4	2022	71	59.4	50	55	49.5	

CREATING DATABASE

In [6]:

```
with sqlite3.connect('austin_weather.db') as conn:
    cursor = conn.cursor()
    cursor.execute("""CREATE TABLE IF NOT EXISTS api (Date int, a_visibility_avg float
                                                      a_wind_speed_min float, a_wind_spe
                                                      a_wind_deg_min float, wind_deg_avg
                                                      a_temp_min float, a_temp_avg float
                                                      a_feels_like_min float, a_feels_li
                                                      a_pressure_min float, a_pressure_a
                                                      a_humidity_min float, a_humidity_a
                                                      a_dew_point_min float, a_dew_point
                                                      PRIMARY KEY (Date))""")

    cursor.execute("""CREATE TABLE IF NOT EXISTS kaggle (Date int, k_temp_max float, k
                                                      k_dew_point_max float, k_dew_point
                                                      k_humidity_max float, k_humidity_a
                                                      k_pressure_max float, k_pressure_a
                                                      k_visibility_max float, k_visibili
                                                      k_wind_max float, k_wind_avg float
                                                      k_events text,
                                                      PRIMARY KEY (Date))""")

    cursor.execute("""CREATE TABLE IF NOT EXISTS website (Date int,
                                                    w_temp_max float, w_temp_avg float, w_
                                                    w_dew_point_max float, w_dew_point_avg
                                                    w_humidity_max float, w_humidity_avg f
                                                    w_wind_speed_max float, w_wind_speed_a
                                                    w_pressure_max float, w_pressure_avg f
                                                    w_precipitation float,
                                                    PRIMARY KEY (Date))""")

    #-------------------------------------------------------------------------------

    # Import datasets into the created database tables
```

```
    df_api.to_sql('api',conn,if_exists='replace',index=False)
    df_kaggle.to_sql('kaggle',conn,if_exists='replace',index=False)
    df_website.to_sql('website',conn,if_exists='replace',index=False)

    conn.commit()
```

Read data from DataBase

In [7]:
```
with sqlite3.connect('austin_weather.db') as conn:
    cursor = conn.cursor()

    # Perform LEFT JOIN on all datasets based on the common key 'Date'
    df_combined = pd.read_sql_query("""SELECT *
                                       FROM
                                       (SELECT api.*, website.* FROM api LEFT JOIN we
                                       ON api.Date = website.Date) t1 LEFT JOIN kaggl
                                       ON t1.Date = t2.Date""", conn)

df_combined.head()
```

Out[7]:

	Date	a_visibility_avg	a_snow_avg	a_wind_speed_min	a_wind_speed_avg	a_wind_speed_max	a_win
0	1	6.206132	0.0	0.00	4.679167	8.99	
1	2	5.920447	0.0	0.00	3.819583	7.00	
2	3	5.837088	0.0	3.44	10.071667	19.57	
3	4	6.063290	0.0	8.05	15.317500	21.85	
4	5	6.122773	0.0	0.00	5.559167	12.66	

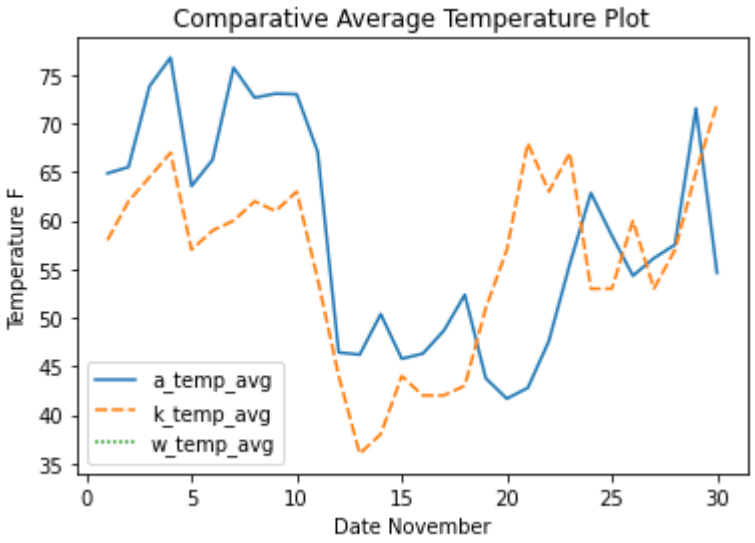
5 rows × 62 columns

Visualizations

Comparative line graphs for the merged dataset having data from all three datasets

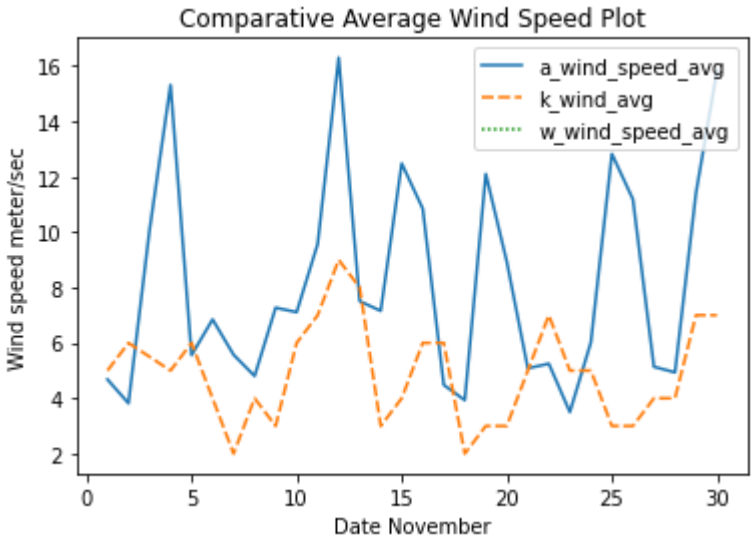
In [8]:
```
# Plot the average tempreture all datasets against each other
# w_temp_avg,  a_temp_avg, k_temp_avg
# 'w' stands for data pulled from website
# 'a' stands for data pulled via api
# 'k' stands for data pulled from kaggle dataset


df_temp = df_combined[['Date','w_temp_avg', 'a_temp_avg', 'k_temp_avg']]
df_temp = df_temp.T.groupby(level=0).first().T
df_temp.set_index('Date', inplace= True)

ax = sns.lineplot(data=df_temp)
ax.set(xlabel='Date November', ylabel='Temperature F', title='Comparative Average Temp
ax.legend(loc='lower left')
```

Out[8]:
```
<matplotlib.legend.Legend at 0x198a49f71c0>
```



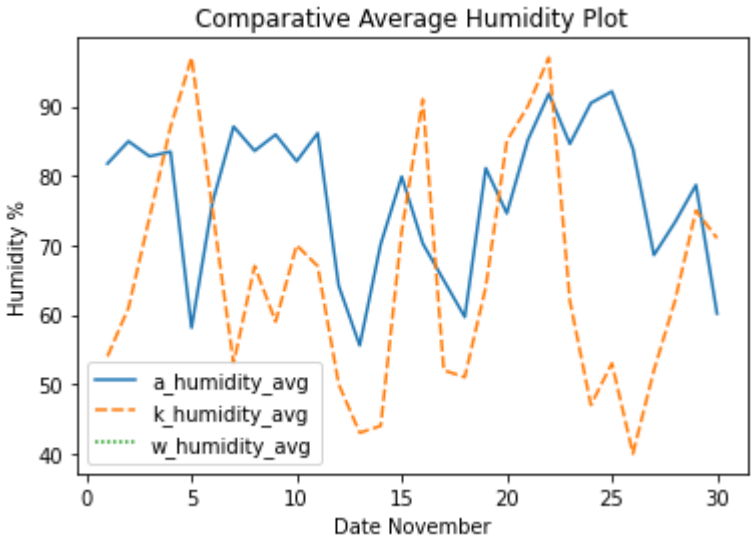
In [9]:
```
df_wind = df_combined[['Date','w_wind_speed_avg', 'a_wind_speed_avg', 'k_wind_avg']]
df_wind = df_wind.T.groupby(level=0).first().T
df_wind.set_index('Date', inplace= True)
ax = sns.lineplot(data=df_wind)
ax.set(xlabel='Date November', ylabel='Wind speed meter/sec', title='Comparative Avera
ax.legend(loc='upper right')
```

Out[9]:
```
<matplotlib.legend.Legend at 0x198a4b14c40>
```



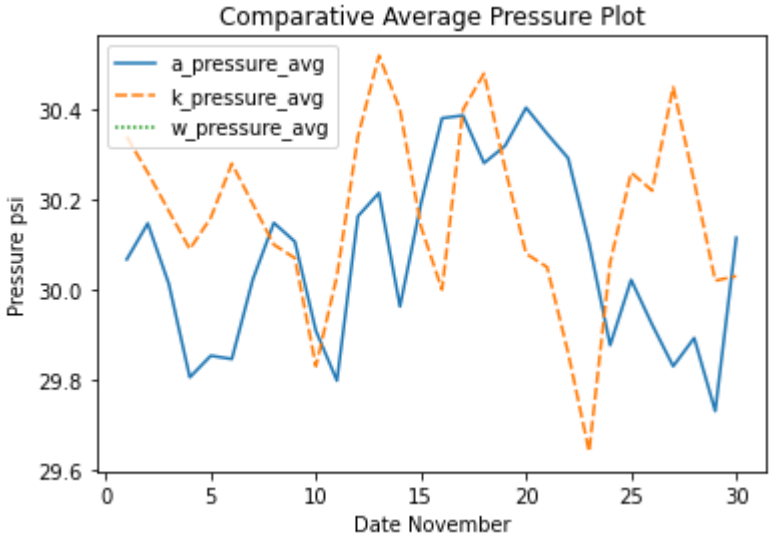
In [10]:
```
df_humidity = df_combined[['Date','w_humidity_avg', 'a_humidity_avg', 'k_humidity_avg'
df_humidity = df_humidity.T.groupby(level=0).first().T
df_humidity.set_index('Date', inplace= True)
ax = sns.lineplot(data=df_humidity)
ax.set(xlabel='Date November', ylabel='Humidity %', title='Comparative Average Humidit
ax.legend(loc='lower left')
```

Out[10]:
```
<matplotlib.legend.Legend at 0x198a4b722b0>
```



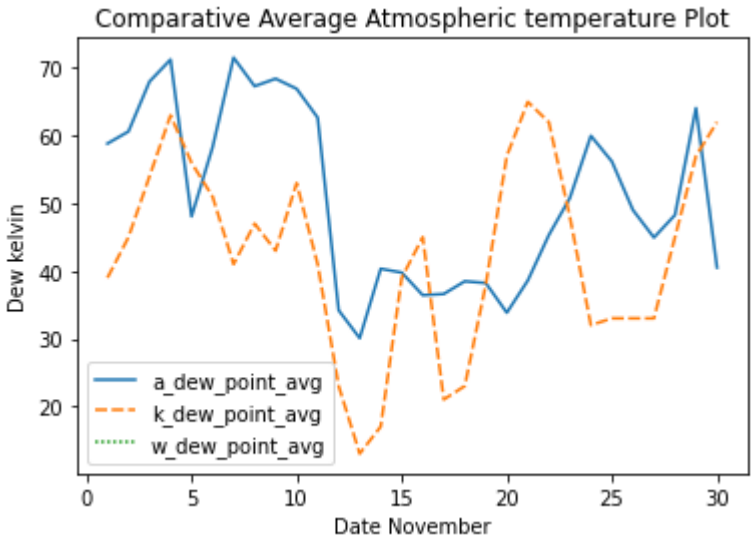
In [11]:
```
df_psi = df_combined[['Date','w_pressure_avg', 'a_pressure_avg', 'k_pressure_avg']]
df_psi = df_psi.T.groupby(level=0).first().T
df_psi.set_index('Date', inplace= True)

ax = sns.lineplot(data=df_psi)
ax.set(xlabel='Date November', ylabel='Pressure psi', title='Comparative Average Press
ax.legend(loc='upper left')
```

Out[11]:
```
<matplotlib.legend.Legend at 0x198a4c587f0>
```



In [12]:
```
df_dew = df_combined[['Date','w_dew_point_avg', 'a_dew_point_avg', 'k_dew_point_avg']]
df_dew = df_dew.T.groupby(level=0).first().T
df_dew.set_index('Date', inplace= True)

ax = sns.lineplot(data=df_dew)

ax.set(xlabel='Date November', ylabel='Dew kelvin', title='Comparative Average Atmosph
ax.legend(loc='lower left')
```

Out[12]:
```
<matplotlib.legend.Legend at 0x198a4ce9430>
```

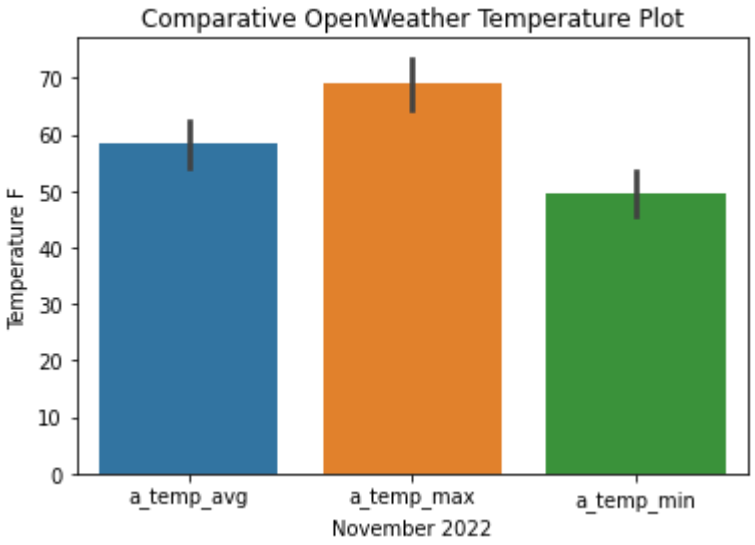


Comparative bar graphs for each data set

In [13]:
```
with sqlite3.connect('austin_weather.db') as conn:
    cursor = conn.cursor()

    # Perform LEFT JOIN on all datasets based on the common key 'Date'
    df_api_db = pd.read_sql_query("""SELECT * FROM api""", conn)
    df_website_db = pd.read_sql_query("""SELECT * FROM website""", conn)
    df_kaggle_db = pd.read_sql_query("""SELECT * FROM kaggle""", conn)
```

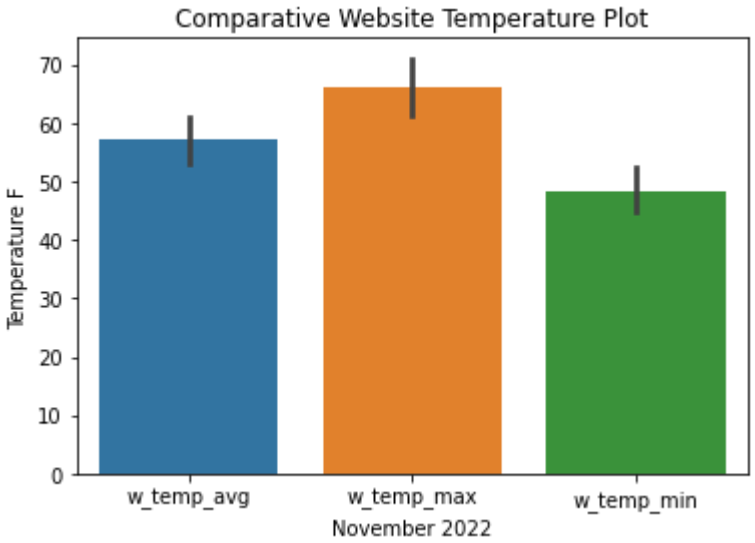
In [14]:
```
df_temp = df_api_db[['Date','a_temp_min', 'a_temp_avg', 'a_temp_max']]
df_temp = df_temp.T.groupby(level=0).first().T
df_temp.set_index('Date', inplace= True)

ax = sns.barplot(data=df_temp)
ax.set(xlabel='November 2022', ylabel='Temperature F', title='Comparative OpenWeather
```

Out[14]:
```
[Text(0.5, 0, 'November 2022'),
 Text(0, 0.5, 'Temperature F'),
 Text(0.5, 1.0, 'Comparative OpenWeather Temperature Plot')]
```



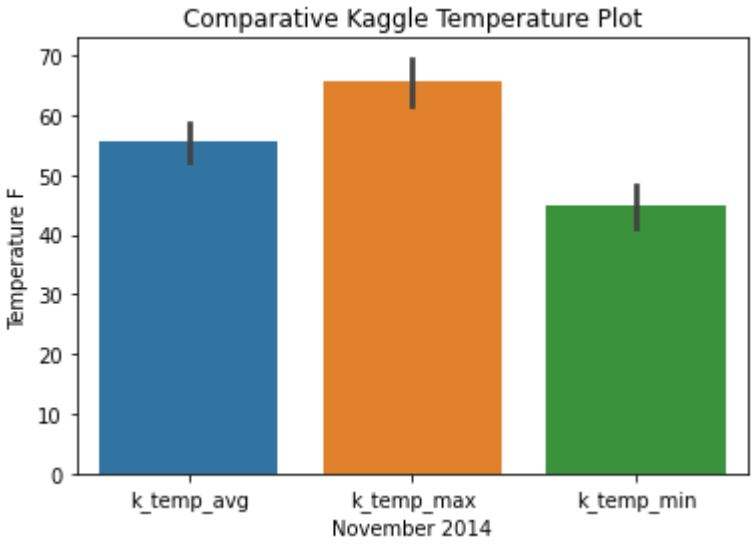
In [15]:
```
df_temp = df_website_db[['Date','w_temp_min', 'w_temp_avg', 'w_temp_max']]
df_temp = df_temp.T.groupby(level=0).first().T
df_temp.set_index('Date', inplace= True)

ax = sns.barplot(data=df_temp)
ax.set(xlabel='November 2022', ylabel='Temperature F', title='Comparative Website Temp
```

Out[15]:
```
[Text(0.5, 0, 'November 2022'),
 Text(0, 0.5, 'Temperature F'),
 Text(0.5, 1.0, 'Comparative Website Temperature Plot')]
```



In [16]:
```
df_temp = df_kaggle_db[['Date','k_temp_min', 'k_temp_avg', 'k_temp_max']]
df_temp = df_temp.T.groupby(level=0).first().T
df_temp.set_index('Date', inplace= True)

ax = sns.barplot(data=df_temp)
ax.set(xlabel='November 2014', ylabel='Temperature F', title='Comparative Kaggle Tempe
```

Out[16]:
```
[Text(0.5, 0, 'November 2014'),
 Text(0, 0.5, 'Temperature F'),
 Text(0.5, 1.0, 'Comparative Kaggle Temperature Plot')]
```



Datasets

In [17]:
```
df_api_db
```

Out[17]:

	Date	a_visibility_avg	a_snow_avg	a_wind_speed_min	a_wind_speed_avg	a_wind_speed_max	a_wi
0	1	6.206132	0.0	0.00	4.679167	8.99	
1	2	5.920447	0.0	0.00	3.819583	7.00	
2	3	5.837088	0.0	3.44	10.071667	19.57	
3	4	6.063290	0.0	8.05	15.317500	21.85	
4	5	6.122773	0.0	0.00	5.559167	12.66	
5	6	6.215040	0.0	0.00	6.849583	19.57	
6	7	6.096048	0.0	0.00	5.571667	11.50	
7	8	5.956080	0.0	0.00	4.803750	10.00	
8	9	6.215040	0.0	3.44	7.270417	16.11	
9	10	6.104982	0.0	3.44	7.111250	12.66	
10	11	6.215040	0.0	1.99	9.574583	23.02	
11	12	6.215040	0.0	11.50	16.305833	21.85	
12	13	3.884400	0.0	4.00	7.507083	10.36	
13	14	5.637637	0.0	5.01	7.158333	10.36	
14	15	6.215040	0.0	9.22	12.482500	17.27	
15	16	6.215040	0.0	7.00	10.850000	18.01	
16	17	6.215040	0.0	1.01	4.482917	10.36	
17	18	6.215040	0.0	0.00	3.931667	8.05	
18	19	6.164466	0.0	1.99	12.100833	19.57	
19	20	6.215040	0.0	4.61	8.933750	17.27	
20	21	6.063290	0.0	1.99	5.083333	9.22	
21	22	4.968070	0.0	1.99	5.249167	8.01	
22	23	6.146649	0.0	0.00	3.501667	8.01	
23	24	3.528796	0.0	1.01	6.020417	16.11	
24	25	6.122773	0.0	5.75	12.827917	20.71	
25	26	5.232935	0.0	4.61	11.187917	18.41	
26	27	6.215040	0.0	0.00	5.137083	12.66	
27	28	6.215040	0.0	0.00	4.935833	12.66	
28	29	6.021597	0.0	5.75	11.412917	20.71	
29	30	6.215040	0.0	3.00	15.830000	24.16	

30 rows × 24 columns

In [18]:
```
df_website_db
```

Out[18]:

	Date	w_temp_max	w_temp_avg	w_temp_min	w_dew_point_max	w_dew_point_avg	w_dew_point
0	2022	66	60.8	55	62	59.5	
1	2022	76	63.6	54	68	60.4	
2	2022	84	74.6	69	70	68.7	
3	2022	85	73.5	59	75	69.0	
4	2022	71	59.4	50	55	49.5	
5	2022	83	66.5	48	73	62.2	
6	2022	84	74.9	68	75	71.3	
7	2022	78	69.1	60	71	66.6	
8	2022	81	71.2	64	69	67.0	
9	2022	81	71.5	64	69	66.4	
10	2022	79	60.2	43	69	55.6	
11	2022	58	45.2	34	32	29.5	
12	2022	58	45.1	29	35	27.5	
13	2022	52	50.0	48	50	45.1	
14	2022	50	43.7	40	43	36.0	
15	2022	55	45.9	40	37	33.9	
16	2022	58	49.4	40	41	35.5	
17	2022	55	50.6	48	45	40.1	
18	2022	47	40.4	37	41	36.6	
19	2022	47	42.4	39	39	32.6	
20	2022	45	43.6	41	44	41.0	
21	2022	54	49.6	44	51	47.0	
22	2022	65	58.8	53	61	52.7	
23	2022	67	62.5	61	65	61.7	
24	2022	61	57.5	55	60	56.3	
25	2022	60	51.7	41	57	45.8	
26	2022	71	54.8	43	47	43.5	
27	2022	75	58.9	37	65	51.9	
28	2022	82	69.8	52	69	64.5	
29	2022	58	47.5	38	46	29.9	

In [19]:
```
print(df_kaggle_db)
```

	Date	k_temp_max	k_temp_avg	k_temp_min	k_dew_point_max
0	1	68	58	47	43
1	2	75	62	48	52
2	4	75	67	58	69
3	5	59	57	55	58
4	6	63	59	55	56
5	7	71	60	48	49
6	8	75	62	48	52
7	9	74	61	47	45
8	10	76	63	49	61
9	11	65	54	42	62
10	12	51	44	36	29
11	13	41	36	31	17
12	14	45	38	30	23
13	15	48	44	40	47
14	16	50	42	34	49
15	17	53	42	30	27
16	18	56	43	30	27
17	19	69	51	33	45
18	20	70	57	44	63
19	21	74	68	62	69
20	22	67	63	58	64
21	23	81	67	53	57
22	24	66	53	40	39
23	25	67	53	38	38
24	26	77	60	42	37
25	27	64	53	42	37
26	28	72	57	42	54
27	29	73	65	56	61
28	30	79	72	64	63

	k_dew_point_avg	k_dew_point_min	k_humidity_max	k_humidity_avg
0	39	37	71	54
1	45	39	80	61
2	63	57	100	87
3	56	54	100	97
4	51	46	93	75
5	41	34	80	53
6	47	41	93	67
7	43	40	86	59
8	53	43	93	70
9	41	29	93	67
10	23	16	64	50
11	13	9	58	43
12	17	12	58	44
13	39	23	100	72
14	45	28	100	91
15	21	14	82	52
16	23	19	78	51
17	38	23	86	64
18	57	43	100	85
19	65	62	100	90
20	62	56	100	97
21	48	39	100	62
22	32	19	76	47
23	33	25	85	53
24	33	27	62	40
25	33	27	76	52
26	45	35	82	62
27	57	52	93	75

```
28              62              59              87              71

    k_humidity_min  ...  k_pressure_avg  k_pressure_min  k_visibility_max  \
0               37  ...           30.34           30.28                10
1               41  ...           30.26           30.18                10
2               73  ...           30.09           30.01                10
3               93  ...           30.16           30.08                10
4               56  ...           30.28           30.22                10
5               26  ...           30.19           30.06                10
6               41  ...           30.10           30.04                10
7               31  ...           30.07           29.92                10
8               46  ...           29.83           29.74                10
9               40  ...           30.03           29.78                10
10              36  ...           30.34           30.25                10
11              27  ...           30.52           30.46                10
12              30  ...           30.40           30.30                10
13              43  ...           30.14           30.03                10
14              82  ...           30.00           29.92                10
15              22  ...           30.40           30.25                10
16              24  ...           30.48           30.42                10
17              42  ...           30.27           30.17                10
18              70  ...           30.08           30.02                10
19              79  ...           30.05           29.99                10
20              93  ...           29.86           29.70                10
21              23  ...           29.64           29.51                10
22              18  ...           30.06           29.88                10
23              20  ...           30.26           30.20                10
24              18  ...           30.22           30.14                10
25              27  ...           30.45           30.37                10
26              41  ...           30.24           30.09                10
27              57  ...           30.02           29.95                10
28              54  ...           30.03           29.97                10

    k_visibility_avg  k_visibility_min  k_wind_max  k_wind_avg  k_windgust  \
0                 10                10           9           5          14
1                 10                10          15           6          25
2                  8                 2          17           5          28
3                  6                 2          13           6          20
4                 10                 5          10           4          17
5                 10                10           7           2          11
6                 10                10          15           4          25
7                 10                10           8           3          13
8                 10                10          17           6          31
9                 10                 9          17           7          32
10                10                10          16           9          27
11                10                10          17           8          28
12                10                10           8           3          11
13                 5                 1          10           4          15
14                 4                 0          17           6          28
15                10                10          16           6          26
16                10                 3          10           2          15
17                10                10          13           3          22
18                 8                 0          12           3          16
19                 6                 0          13           5          20
20                 6                 0          20           7          28
21                10                10          16           5          27
22                10                10          15           5          28
23                10                10           9           3          16
24                10                10          10           3          17
25                10                10          10           4          16
```

```
26                10                10          14           4          24
27                10                10          16           7          31
28                10                10          15           7          28

    k_precipitationsum                   k_events
0                 0.00
1                 0.00
2                 0.56                       Rain
3                 1.51                       Rain
4                 0.04                       Rain
5                 0.00
6                 0.00
7                 0.00
8                 0.00
9                 0.00
10                0.00
11                0.00
12                0.00
13                0.02                       Rain
14                0.01                  Fog , Rain
15                0.00
16                0.00
17                0.00
18                0.01                  Fog , Rain
19                0.33                  Fog , Rain
20                3.33  Fog , Rain , Thunderstorm
21                0.00
22                0.00
23                0.00
24                0.00
25                0.00
26                0.00
27                0.00
28                0.00

[29 rows x 21 columns]
```

In [20]:
```
print(df_combined)
```

	Date	a_visibility_avg	a_snow_avg	a_wind_speed_min	a_wind_speed_avg
0	1	6.206132	0.0	0.00	4.679167
1	2	5.920447	0.0	0.00	3.819583
2	3	5.837088	0.0	3.44	10.071667
3	4	6.063290	0.0	8.05	15.317500
4	5	6.122773	0.0	0.00	5.559167
5	6	6.215040	0.0	0.00	6.849583
6	7	6.096048	0.0	0.00	5.571667
7	8	5.956080	0.0	0.00	4.803750
8	9	6.215040	0.0	3.44	7.270417
9	10	6.104982	0.0	3.44	7.111250
10	11	6.215040	0.0	1.99	9.574583
11	12	6.215040	0.0	11.50	16.305833
12	13	3.884400	0.0	4.00	7.507083
13	14	5.637637	0.0	5.01	7.158333
14	15	6.215040	0.0	9.22	12.482500
15	16	6.215040	0.0	7.00	10.850000
16	17	6.215040	0.0	1.01	4.482917
17	18	6.215040	0.0	0.00	3.931667
18	19	6.164466	0.0	1.99	12.100833
19	20	6.215040	0.0	4.61	8.933750
20	21	6.063290	0.0	1.99	5.083333
21	22	4.968070	0.0	1.99	5.249167
22	23	6.146649	0.0	0.00	3.501667
23	24	3.528796	0.0	1.01	6.020417
24	25	6.122773	0.0	5.75	12.827917
25	26	5.232935	0.0	4.61	11.187917
26	27	6.215040	0.0	0.00	5.137083
27	28	6.215040	0.0	0.00	4.935833
28	29	6.021597	0.0	5.75	11.412917
29	30	6.215040	0.0	3.00	15.830000

	a_wind_speed_max	a_wind_deg_min	a_wind_deg_avg	a_wind_deg_max
0	8.99	0	128.458333	360
1	7.00	0	95.708333	221
2	19.57	0	156.250000	190
3	21.85	144	168.083333	190
4	12.66	0	171.291667	357
5	19.57	0	164.250000	230
6	11.50	0	119.083333	200
7	10.00	0	92.916667	180
8	16.11	0	112.875000	180
9	12.66	0	123.750000	200
10	23.02	95	213.916667	350
11	21.85	10	268.166667	360
12	10.36	6	58.916667	350
13	10.36	0	139.833333	360
14	17.27	10	236.833333	360
15	18.01	10	34.833333	58
16	10.36	0	189.583333	350
17	8.05	0	84.416667	180
18	19.57	0	86.833333	360
19	17.27	18	243.083333	360
20	9.22	10	187.500000	360
21	8.01	0	94.666667	360
22	8.01	0	95.708333	330
23	16.11	0	120.750000	360
24	20.71	10	99.291667	360
25	18.41	10	207.458333	320
26	12.66	0	220.041667	350

27	12.66	0	171.916667	346
28	20.71	140	189.583333	240
29	24.16	10	158.625000	360

	a_temp_min	...	k_pressure_avg	k_pressure_min	k_visibility_max
0	55.42	...	30.34	30.28	10.0
1	58.28	...	30.26	30.18	10.0
2	68.58	...	NaN	NaN	NaN
3	71.67	...	30.09	30.01	10.0
4	50.02	...	30.16	30.08	10.0
5	48.22	...	30.28	30.22	10.0
6	69.03	...	30.19	30.06	10.0
7	62.58	...	30.10	30.04	10.0
8	66.45	...	30.07	29.92	10.0
9	66.22	...	29.83	29.74	10.0
10	45.39	...	30.03	29.78	10.0
11	38.28	...	30.34	30.25	10.0
12	32.02	...	30.52	30.46	10.0
13	45.32	...	30.40	30.30	10.0
14	37.38	...	30.14	30.03	10.0
15	39.58	...	30.00	29.92	10.0
16	39.22	...	30.40	30.25	10.0
17	47.57	...	30.48	30.42	10.0
18	36.28	...	30.27	30.17	10.0
19	37.38	...	30.08	30.02	10.0
20	39.38	...	30.05	29.99	10.0
21	43.29	...	29.86	29.70	10.0
22	50.58	...	29.64	29.51	10.0
23	59.58	...	30.06	29.88	10.0
24	54.39	...	30.26	30.20	10.0
25	47.39	...	30.22	30.14	10.0
26	42.82	...	30.45	30.37	10.0
27	37.42	...	30.24	30.09	10.0
28	62.02	...	30.02	29.95	10.0
29	39.38	...	30.03	29.97	10.0

	k_visibility_avg	k_visibility_min	k_wind_max	k_wind_avg	k_windgust
0	10.0	10.0	9.0	5.0	14.0
1	10.0	10.0	15.0	6.0	25.0
2	NaN	NaN	NaN	NaN	NaN
3	8.0	2.0	17.0	5.0	28.0
4	6.0	2.0	13.0	6.0	20.0
5	10.0	5.0	10.0	4.0	17.0
6	10.0	10.0	7.0	2.0	11.0
7	10.0	10.0	15.0	4.0	25.0
8	10.0	10.0	8.0	3.0	13.0
9	10.0	10.0	17.0	6.0	31.0
10	10.0	9.0	17.0	7.0	32.0
11	10.0	10.0	16.0	9.0	27.0
12	10.0	10.0	17.0	8.0	28.0
13	10.0	10.0	8.0	3.0	11.0
14	5.0	1.0	10.0	4.0	15.0
15	4.0	0.0	17.0	6.0	28.0
16	10.0	10.0	16.0	6.0	26.0
17	10.0	3.0	10.0	2.0	15.0
18	10.0	10.0	13.0	3.0	22.0
19	8.0	0.0	12.0	3.0	16.0
20	6.0	0.0	13.0	5.0	20.0
21	6.0	0.0	20.0	7.0	28.0
22	10.0	10.0	16.0	5.0	27.0

```
23              10.0              10.0        15.0         5.0        28.0
24              10.0              10.0         9.0         3.0        16.0
25              10.0              10.0        10.0         3.0        17.0
26              10.0              10.0        10.0         4.0        16.0
27              10.0              10.0        14.0         4.0        24.0
28              10.0              10.0        16.0         7.0        31.0
29              10.0              10.0        15.0         7.0        28.0

    k_precipitationsum                  k_events
0                 0.00
1                 0.00
2                  NaN                      None
3                 0.56                      Rain
4                 1.51                      Rain
5                 0.04                      Rain
6                 0.00
7                 0.00
8                 0.00
9                 0.00
10                0.00
11                0.00
12                0.00
13                0.00
14                0.02                      Rain
15                0.01                 Fog , Rain
16                0.00
17                0.00
18                0.00
19                0.01                 Fog , Rain
20                0.33                 Fog , Rain
21                3.33  Fog , Rain , Thunderstorm
22                0.00
23                0.00
24                0.00
25                0.00
26                0.00
27                0.00
28                0.00
29                0.00

[30 rows x 62 columns]
```

Austin Weather Analysis

Introduction

In this study, we have used three different weather datasets for Austin weather analysis. These datasets were gathered from three different sources and formats (CSV, JSON, and HTML table).

Data Description:

First: Kaggle Austin weather analysis dataset in CSV format: This dataset is in CSV format and contains data for every date from 2013-12-21 to 2017-07-31. It has 1319 rows and 21 columns. For our analysis, we only used the Month of November in 2014 because it had fewer missing

values as compared to other years. The following are attributes found in this dataset: link: https://www.kaggle.com/datasets/grubenm/austin-weather

Second: Austin weather captured from OpenWeatherMap in JSON format. It contains data from 2013-12-01 to 2022-12-08 with 82540 rows and 26 columns. link: https://home.openweathermap.org/history_forecast_bulks/new

Third: Austin weather from the Wunderground website for the month of Nov-2022 in HTML format. It consists of 19 different variables. For our analysis, we have used daily resolution for the month of November 2022. Totally it consists of 19 different attributes with 30 rows. For this purpose, we have used the BeautifulSoup library to be able to perform web scraping and convert the HTML format table into a pandas data frame. link: https://www.wunderground.com/history/monthly/us/tx/austin/KAUS/date/2022-11

Relationships

These datasets are connected to each other using the city name (Austin) and date (Month Nov). It is also worth mentioning that the JSON format file has been downloaded from OpenWeatherMap through the bulk history feature. There was no specific API to get these historical data through an API, however, we have an open API to get real-time/historical weather information from this website.

Data Clean-up

After data cleaning (handling missing data/transformation/header name change) of these datasets the final products were stored in an SQLite database for further analysis (Joining through a common key of date). In the end, we pulled these data out from the SQLite database and used a python visualization library called "Seaborn: graphical representation of weather information have plotted to provide visual insights.

Lessen learned

Data from different sources might be coming in different formats and metrics, for this, we need to be equipped well with the coding tools (pandas, scipy, numpy, ...) in order to be able to handle them appropriately. Web scraping was a more challenging part of data wrangling as the HTML tables have no specific standard format. Finally, after saving these datasets in an SQLite database I observed that the volume of storage has been reduced as compared to having them all in CSV formats.

Ethical Implications

In order to minimize the physical and emotional harm following steps have been taken:

- Reference links are provided in case one might need to validate the results.
- The transformation has been clearly outlined in each milestone and efforts spent to minimize the error rate and reflect reality.

- Formulas for conversion have already been validated.

Conclusion

Insights provided through the line and bar graphs show that there is a strong correlation between data coming from OpenWeatherMap and from the website as both are referring to the same year and month Nov 2022, however, the difference between them and Kaggle is because Kaggle data represents weather info for Nov 2014. This in turn shows that Nov 2014 was colder than Nov 2022 in this city. On the other hand, we see that metrics: Temperature, Wind speed, Humidity, and Atmospheric Temperature in 2014 Austin has a lower value compared to 2022, except for Pressure.

Finally, overall Data wrangling (Gathering, cleaning, transformation, saving,...) is almost 90% of a data science task to prepare an accurate and reliable data source for the final analysis. The steps must be documented well so one can understand how data was gathered and transformed.

In [ ]: