In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import datetime as dt
```

In [2]:
```
ca=pd.read_csv(r"C:\Users\berid\OneDrive\Desktop\mydata\hourly_weather\city_attributes.csv")
hum=pd.read_csv(r"C:\Users\berid\OneDrive\Desktop\mydata\hourly_weather\humidity.csv")
pre=pd.read_csv(r"C:\Users\berid\OneDrive\Desktop\mydata\hourly_weather\pressure.csv")
temp=pd.read_csv(r"C:\Users\berid\OneDrive\Desktop\mydata\hourly_weather\temperature.csv")
wed=pd.read_csv(r"C:\Users\berid\OneDrive\Desktop\mydata\hourly_weather\weather_description.csv")
wid=pd.read_csv(r"C:\Users\berid\OneDrive\Desktop\mydata\hourly_weather\wind_direction.csv")
ws=pd.read_csv(r"C:\Users\berid\OneDrive\Desktop\mydata\hourly_weather\wind_speed.csv")
```

Unpivot data to make working with it easier!

In [3]:
```
hum=pd.melt(hum,id_vars="datetime",value_vars=hum.columns[1:]).rename(columns={"value":"hum","variable":"city"})
pre=pd.melt(pre,id_vars="datetime",value_vars=pre.columns[1:]).rename(columns={"value":"pre","variable":"city"})
temp=pd.melt(temp,id_vars="datetime",value_vars=temp.columns[1:]).rename(columns={"value":"temp","variable":"city"})
wed=pd.melt(wed,id_vars="datetime",value_vars=wed.columns[1:]).rename(columns={"value":"wed","variable":"city"})
wis=pd.melt(wid,id_vars="datetime",value_vars=wid.columns[1:]).rename(columns={"value":"wid","variable":"city"})
ws=pd.melt(ws,id_vars="datetime",value_vars=ws.columns[1:]).rename(columns={"value":"ws","variable":"city"})
```

Merge dataframes

In [4]:
```
df=hum.merge(pre,on=["datetime","city"])\
.merge(temp,on=["datetime","city"])\
.merge(wed,on=["datetime","city"])\
.merge(wis,on=["datetime","city"])\
.merge(ws,on=["datetime","city"])

df=df.merge(ca,left_on="city",right_on="City")
```

Clean dataframe of NaN values

In [5]:
```
df=df[df.isna().any(axis=1)==False]
```

Convert datetime col to DateTime format and temperature from k to c

In [6]:
```
df["datetime"]=pd.to_datetime(df["datetime"])
df["temp"]=df["temp"]-273.15
```

In [20]:
```
df["year"]=df["datetime"].dt.year
df["month"]=df["datetime"].dt.month_name()
df["monthnum"]=df["datetime"].dt.month
df["day"]=df["datetime"].dt.day
df["hour"]=df["datetime"].dt.hour
df["date"]=df["datetime"].dt.date
```

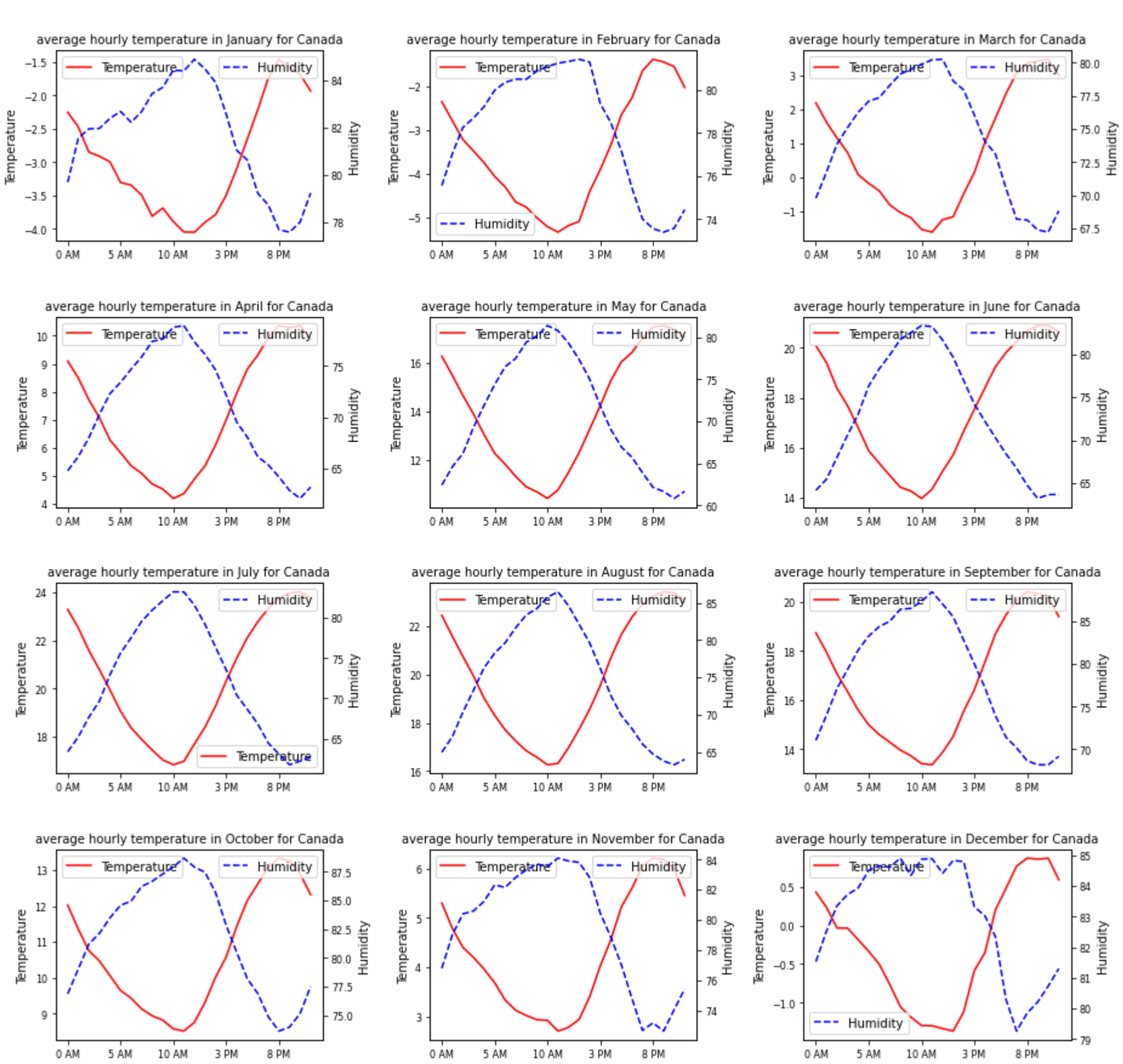
In [8]:
```
df.head()
```

Out[8]:

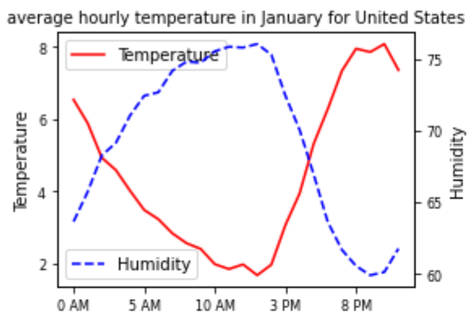
	datetime	city	hum	pre	temp	wed	wid	ws	City	Country	Latitude	Longitude	year	month	monthnum	day	hour
21	2012-10-02 09:00:00	Vancouver	87.0	807.0	11.440217	broken clouds	268.0	0.0	Vancouver	Canada	49.24966	-123.119339	2012	October	10	2	9
22	2012-10-02 10:00:00	Vancouver	88.0	849.0	11.438174	broken clouds	281.0	0.0	Vancouver	Canada	49.24966	-123.119339	2012	October	10	2	10
23	2012-10-02 11:00:00	Vancouver	89.0	890.0	11.436130	broken clouds	295.0	0.0	Vancouver	Canada	49.24966	-123.119339	2012	October	10	2	11
24	2012-10-02 12:00:00	Vancouver	89.0	932.0	11.434087	broken clouds	309.0	0.0	Vancouver	Canada	49.24966	-123.119339	2012	October	10	2	12
25	2012-10-02 13:00:00	Vancouver	90.0	973.0	11.432043	broken clouds	323.0	0.0	Vancouver	Canada	49.24966	-123.119339	2012	October	10	2	13

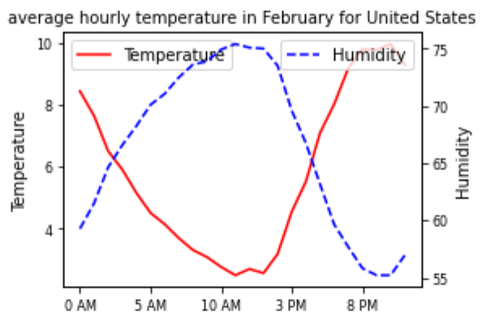
find hottest hour of the day for each month

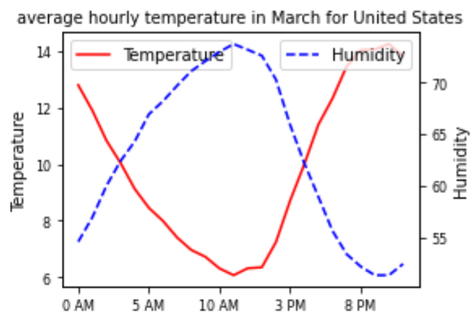
In [9]:
```
for country in df.Country.unique():
    grouped=df[df.Country==country].groupby(["month","hour"]).agg({"hum":"mean","temp":"mean"}).reset_index()


    months=["January","February","March","April","May","June","July","August","September","October","November","December"]
    fig,axes=plt.subplots(4,3,figsize=(15,15))

    for ax,month in zip(axes.ravel(),months):
        filtered=grouped[grouped.month==month].sort_values("hour")
        hours=[str(i)+" AM" if i<=12 else str(i-12)+" PM" for i in filtered.hour]
        filtered["hour"]=hours

        filtered.plot(kind="line",ax=ax,x="hour",y="temp",xlabel="",label="Temperature",color="r",fontsize=8)
        ax.set_ylabel("Temperature")

        ax2=ax.twinx()
        filtered.plot(kind="line",ax=ax2,x="hour",y="hum",xlabel="",label="Humidity",color="b",ls="--",fontsize=8)
        ax2.set_ylabel("Humidity")
        ax2.set_title("average hourly temperature in "+month+" for "+country,size=10)
    plt.legend()
    plt.subplots_adjust(hspace=0.4,wspace=0.4)
    plt.suptitle("Average hourly temperature for "+country,size=20,fontweight="bold")
    plt.show()
```

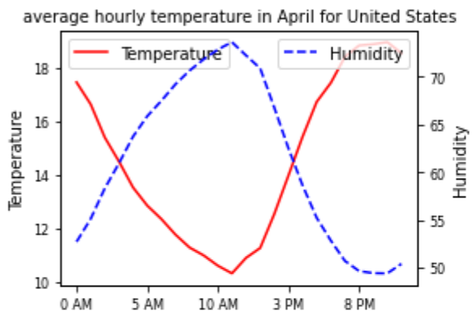


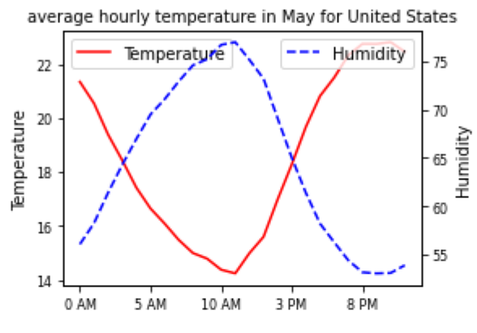
Average hourly temperature for United States

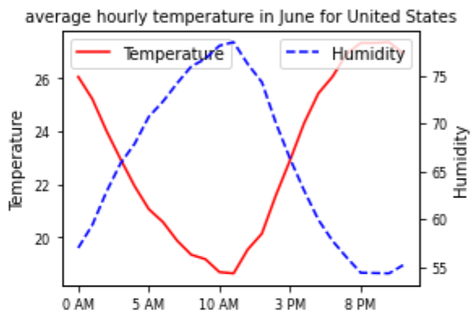


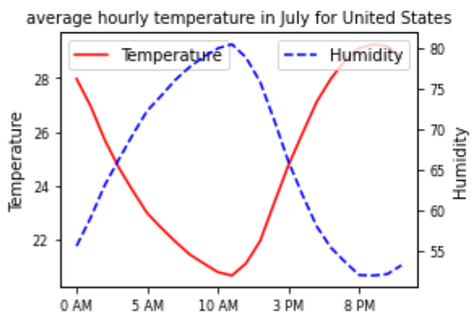


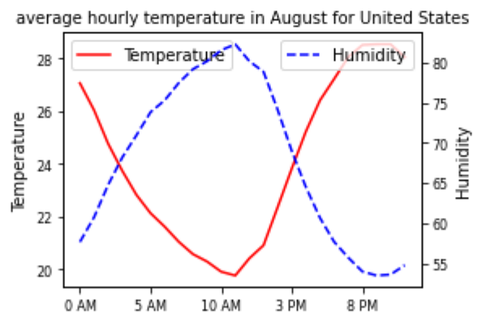


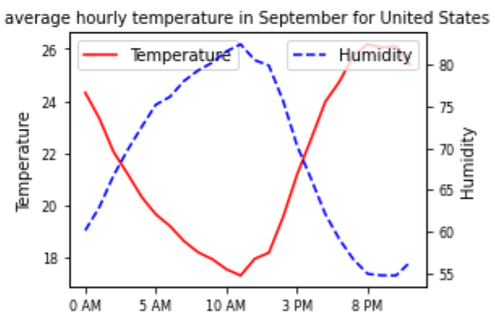


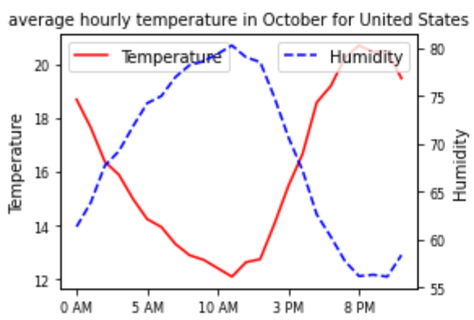


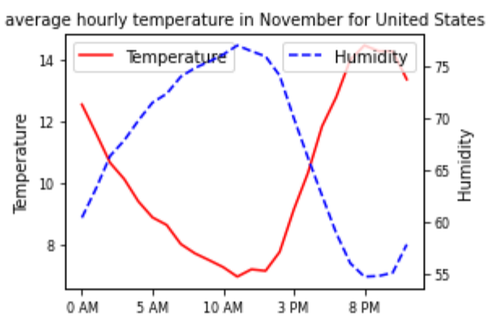


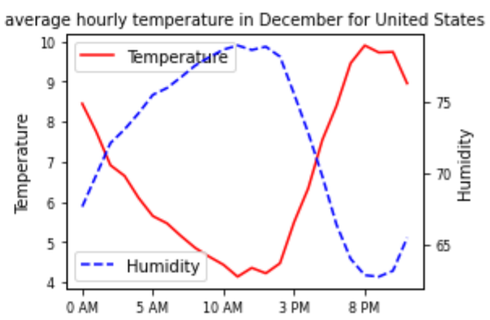


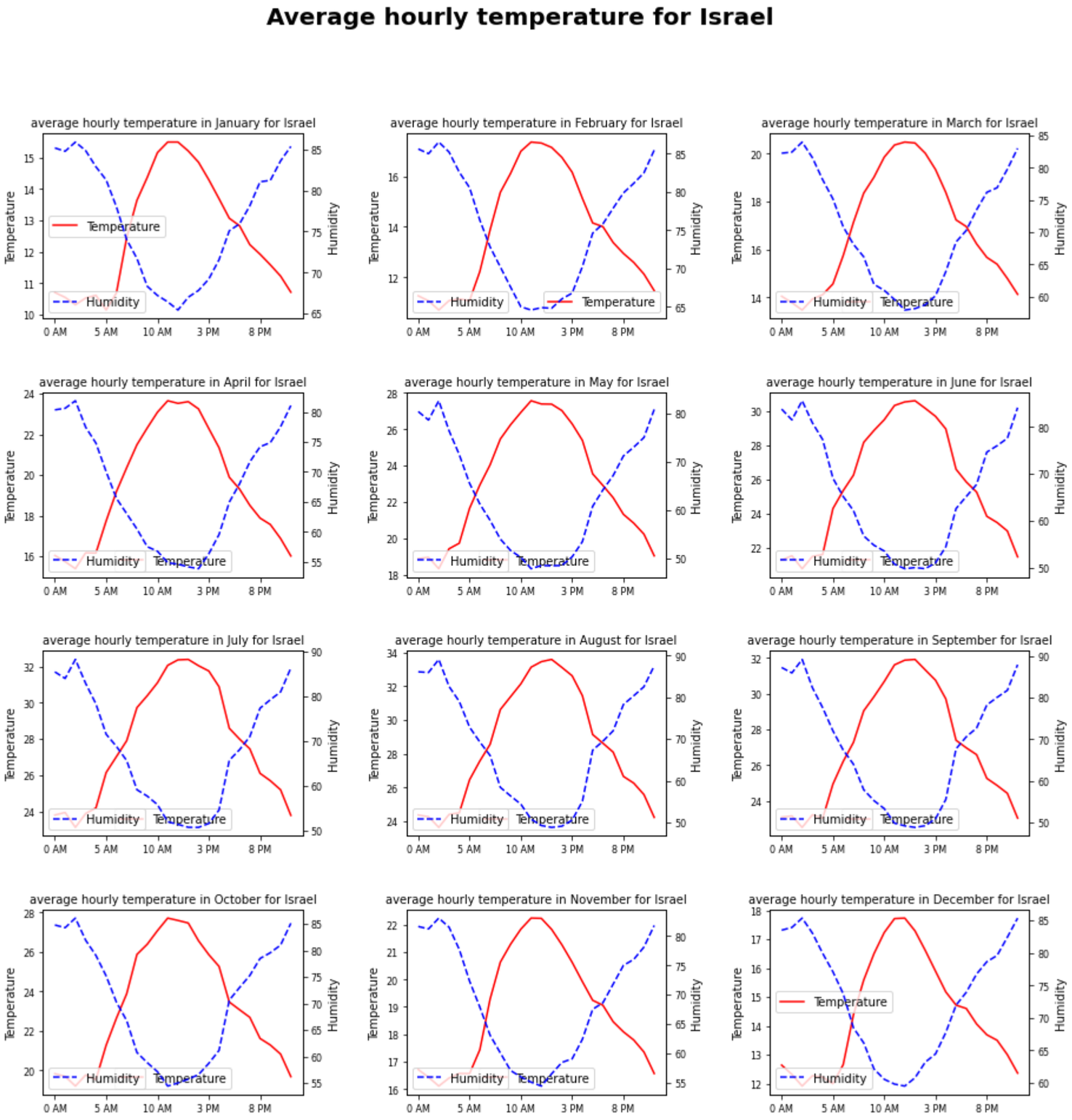








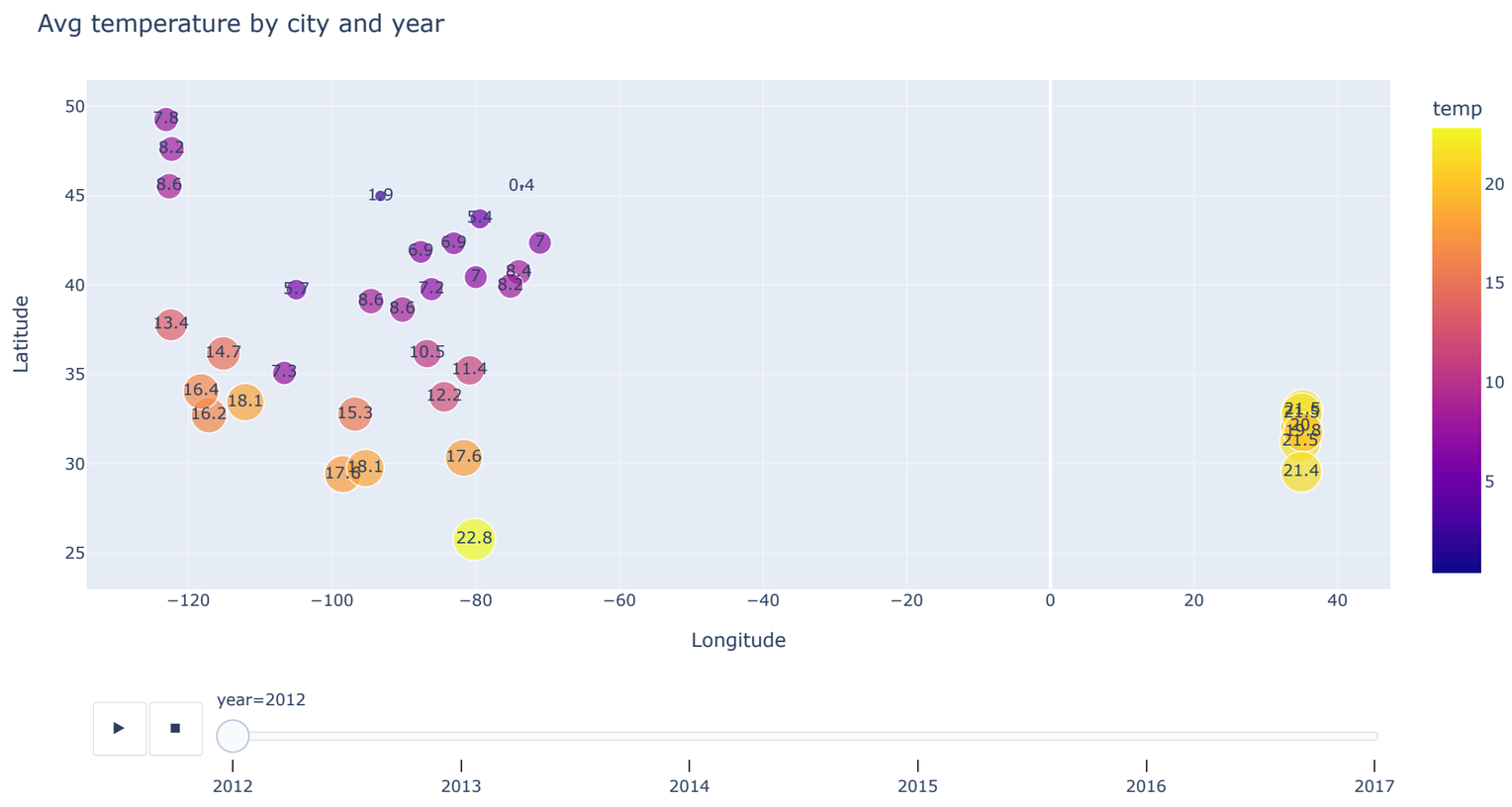




In [10]:
```
# United States and Canada cover large territory, hour of max and min temperatures will be drastically different for each city.
# For Israel, which is a small country, peak hour (~2PM) seems logical
```

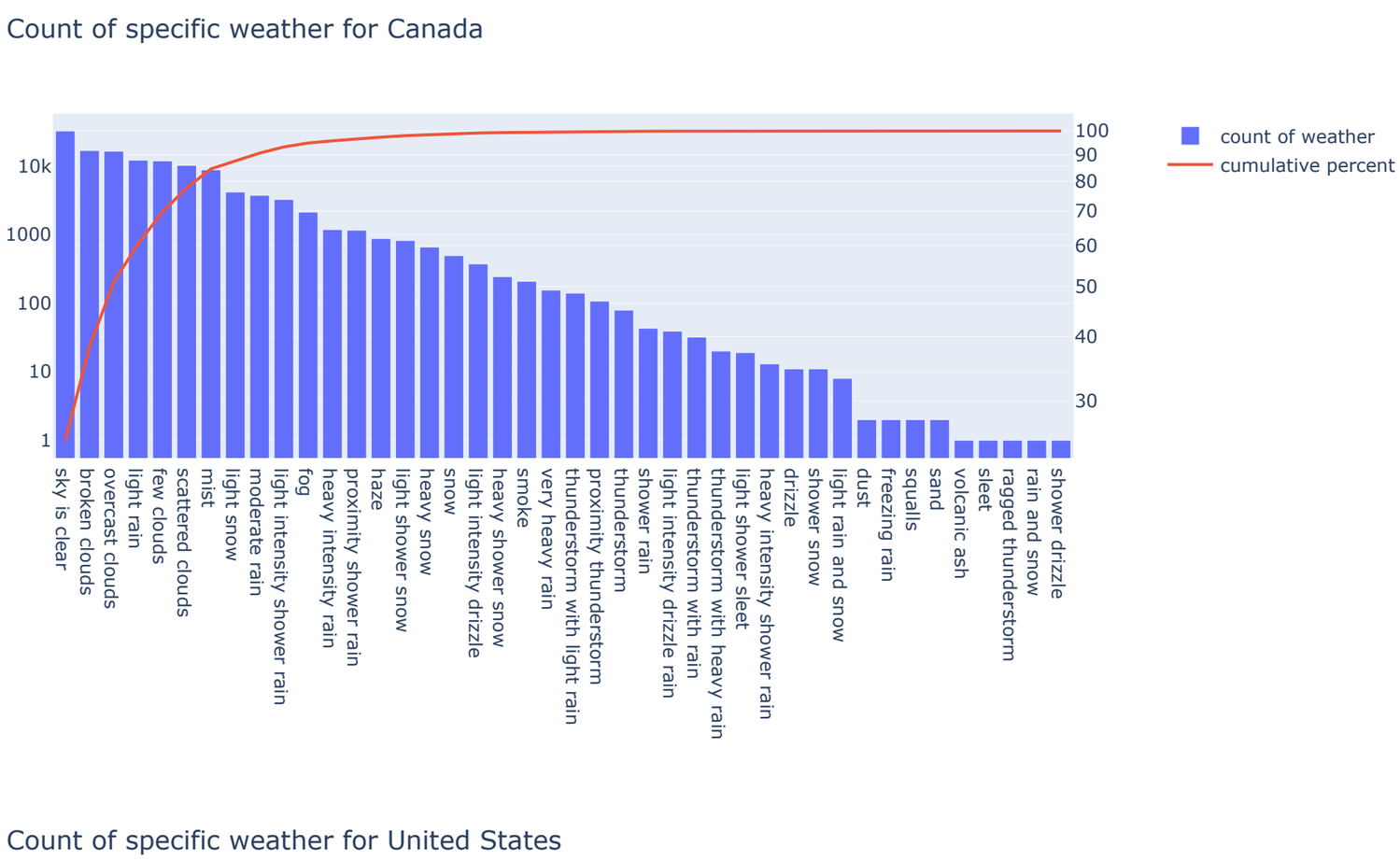
find avg temperature by city and year

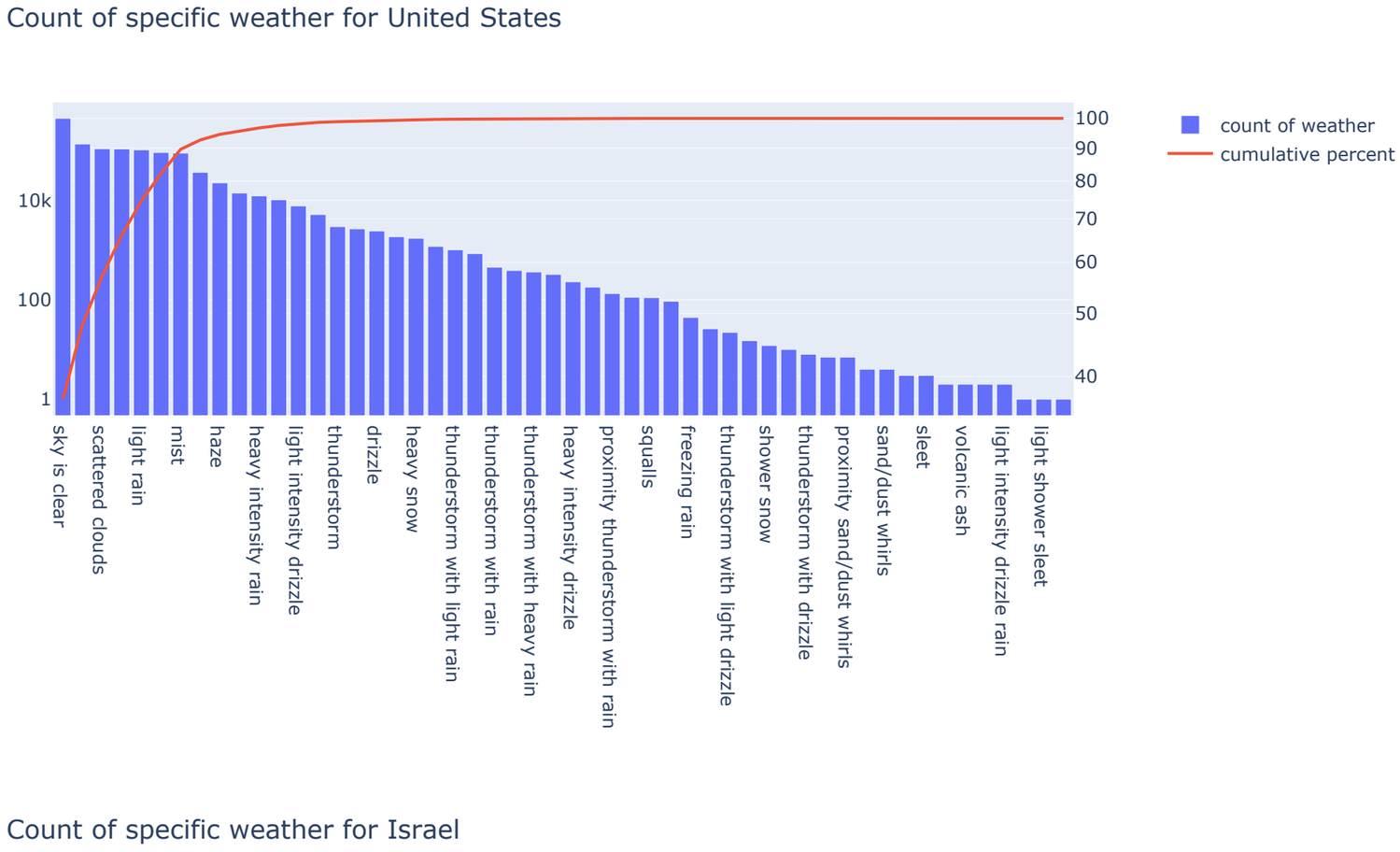
In [11]:
```
grouped=df.groupby(["year","city","Longitude","Latitude"])["temp"].mean().reset_index().sort_values("year")
grouped["temp"]=grouped.temp.round(1)

fig=px.scatter(grouped,x="Longitude",y="Latitude",
               animation_frame="year",
               hover_name="city",
               color="temp",size="temp",
               text="temp")

fig.update_layout(title_text="Avg temperature by city and year", title_font={"size":15},font={"size":10})

fig.show()
```

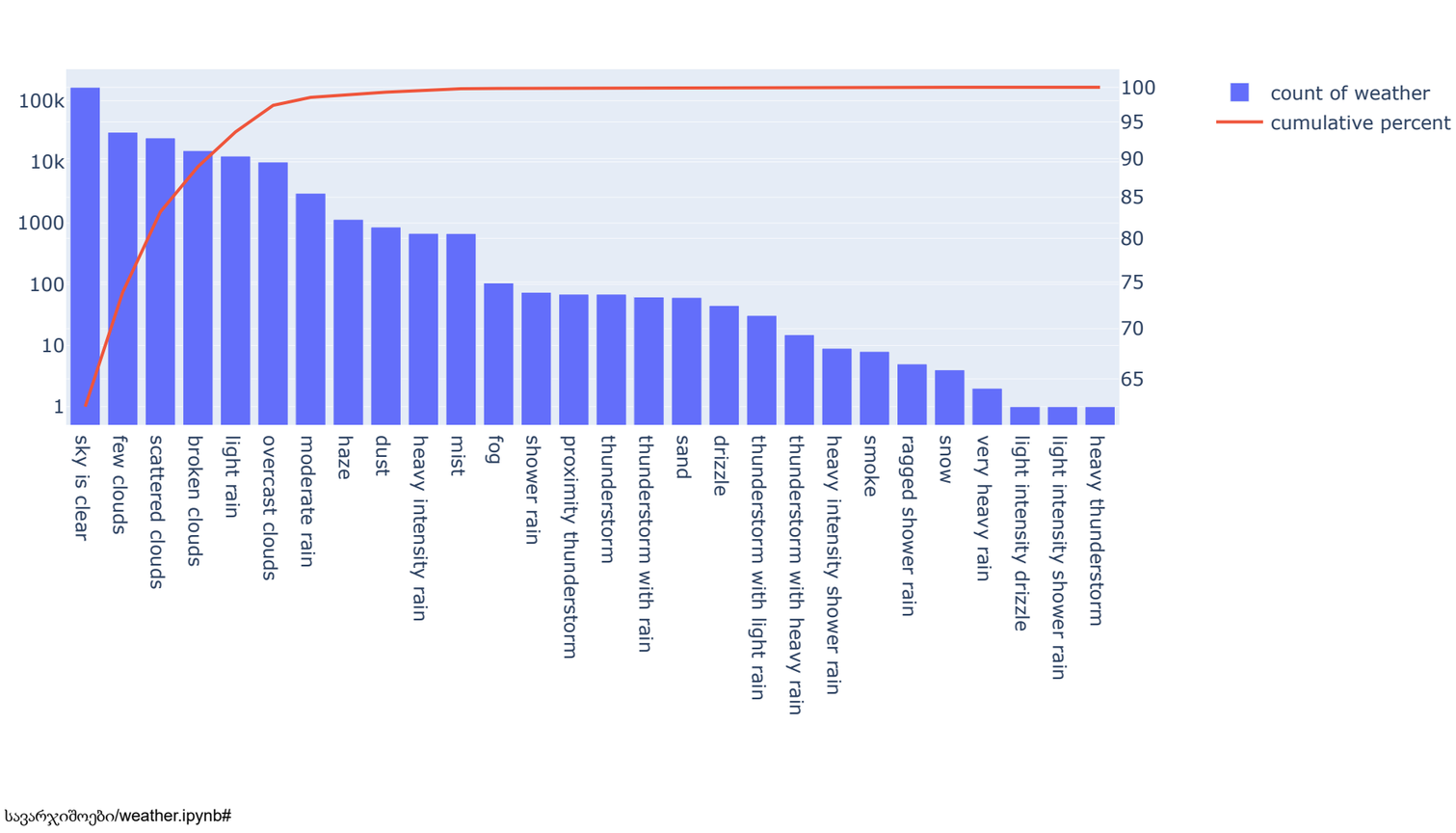


In [12]:
```
countries=df.Country.unique()
for country in countries:

    grouped=df[df.Country==country].wed.value_counts().reset_index().sort_values("wed",ascending=False)
    grouped["cumulative"]=(grouped.wed/grouped.wed.sum()*100).cumsum()

    import plotly.graph_objects as go
    from plotly.subplots import make_subplots

    fig = make_subplots(specs=[[{"secondary_y": True}]])
    fig.add_trace(
        go.Bar(x=grouped["index"],y=grouped["wed"] ,name="count of weather"),
        secondary_y=False)
    fig.update_yaxes(type="log")
    fig.update_layout(title_text="Count of specific weather for "+country)

    fig.add_trace(
        go.Line(x=grouped["index"],y=grouped["cumulative"], name="cumulative percent"),
        secondary_y=True,)
    fig.show()
```

```
C:\Users\berid\AppData\Local\Programs\Python\Python39\lib\site-packages\plotly\graph_objs\_deprecations.py:378: DeprecationWarning:

plotly.graph_objs.Line is deprecated.
Please replace it with one of the following more specific types
  - plotly.graph_objs.scatter.Line
  - plotly.graph_objs.layout.shape.Line
  - etc.
```







Find out How wind direction varies according to the city coordinates

Find cardinal directions

In [13]:
```
wind_direction=[]
for i in df["wid"]:
    if (i>=315 and i<=360) or (i>=0 and i<45):
        wind_direction.append("N")
    elif i>=45 and i<135:
        wind_direction.append("E")
    elif i>=135 and i<225:
        wind_direction.append("S")
    elif i>=225 and i<315:
        wind_direction.append("W")


df["wind_direction"]=wind_direction
```

Find most frequent wind direction for each city

In [14]:
```
city_wind={}
for city in df.City.unique():
    grouped=(df[df.City==city]["wind_direction"].value_counts(normalize=True,sort=True)*100).round().reset_index()
    grouped["City"]=city
    city_wind[city]=grouped

city_wind

wind=pd.DataFrame()
for key,value in city_wind.items():
    wind=pd.concat([wind,value])
```

merge dataframes to get city coordinates and most frequent wind direction for each city

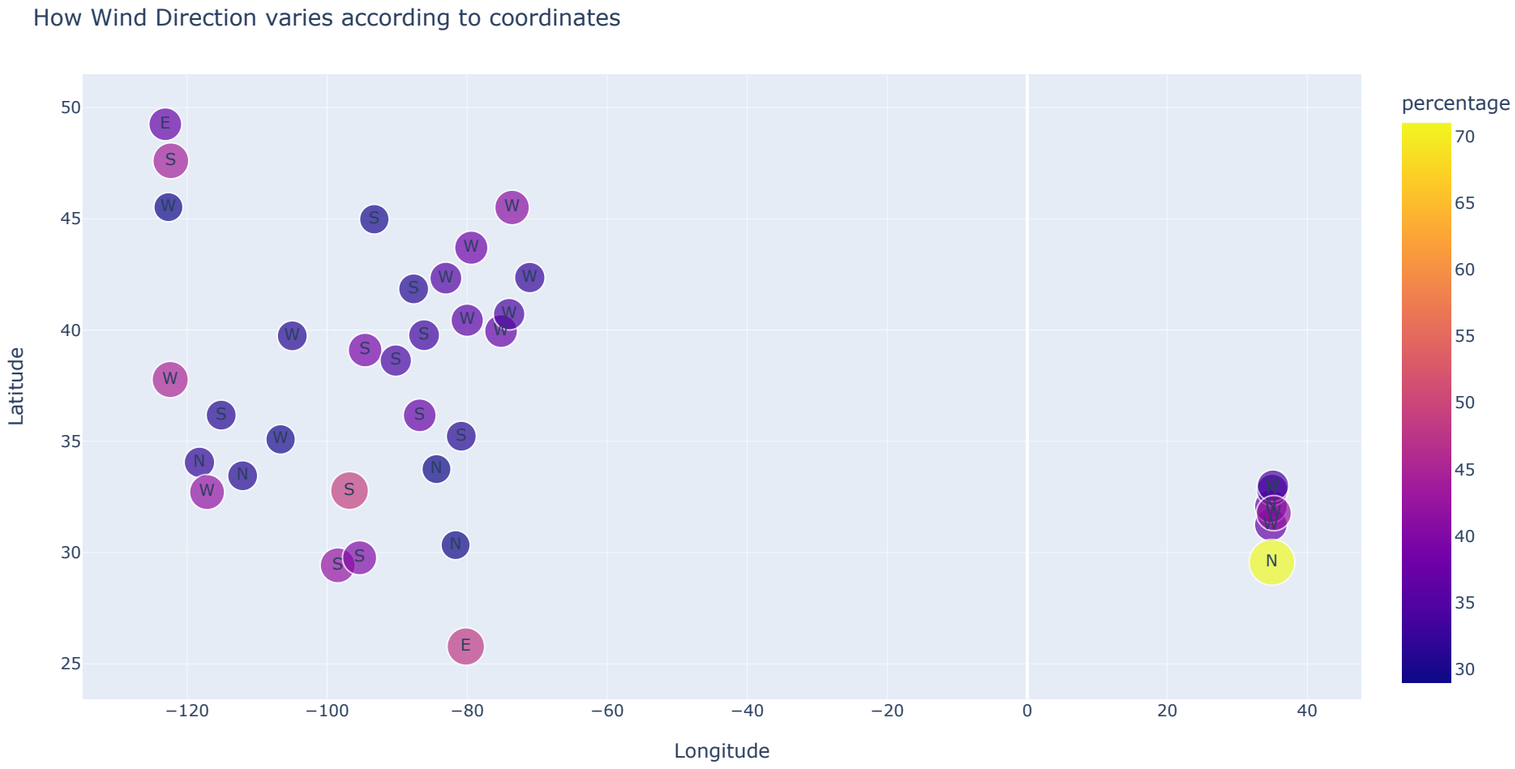
In [15]:
```
coordinates=df[["City","Longitude","Latitude"]].drop_duplicates(keep="first")
merged=coordinates.merge(wind,on="City")
merged=merged.groupby("City").head(1).rename(columns={"index":"WindDirection","wind_direction":"percentage"})
merged.head()
```

Out[15]:

	City	Longitude	Latitude	WindDirection	percentage
0	Vancouver	-123.119339	49.249660	E	37.0
4	Portland	-122.676208	45.523449	W	29.0
8	San Francisco	-122.419418	37.774929	W	45.0
12	Seattle	-122.332069	47.606209	S	44.0
16	Los Angeles	-118.243683	34.052231	N	32.0

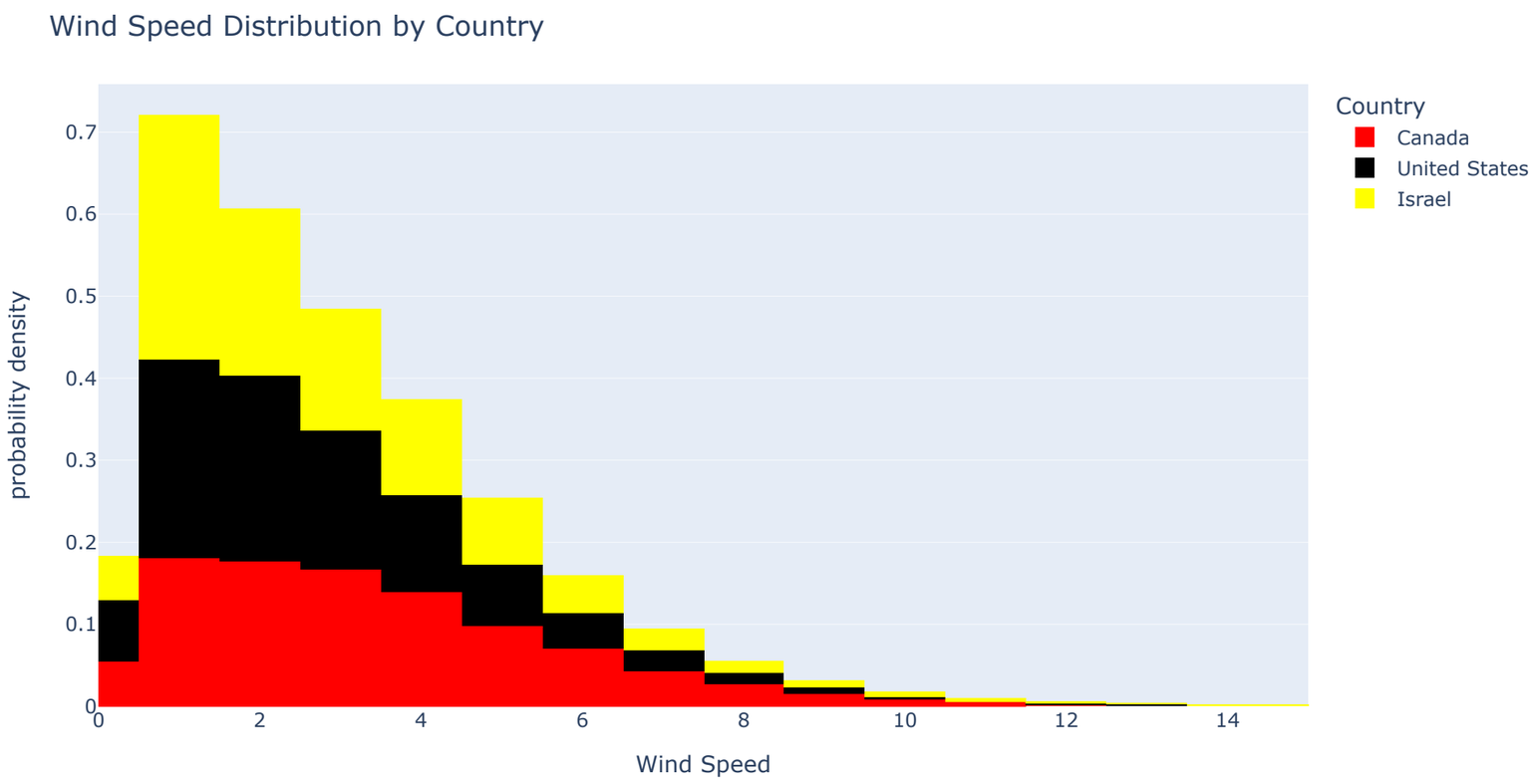
plot data

In [16]:
```
fig=px.scatter(merged,x="Longitude",y="Latitude",
          hover_name="City",
          text="WindDirection",
           size="percentage",
               color="percentage")
fig.update_layout(title_text="How Wind Direction varies according to coordinates", font={"size":10})
```



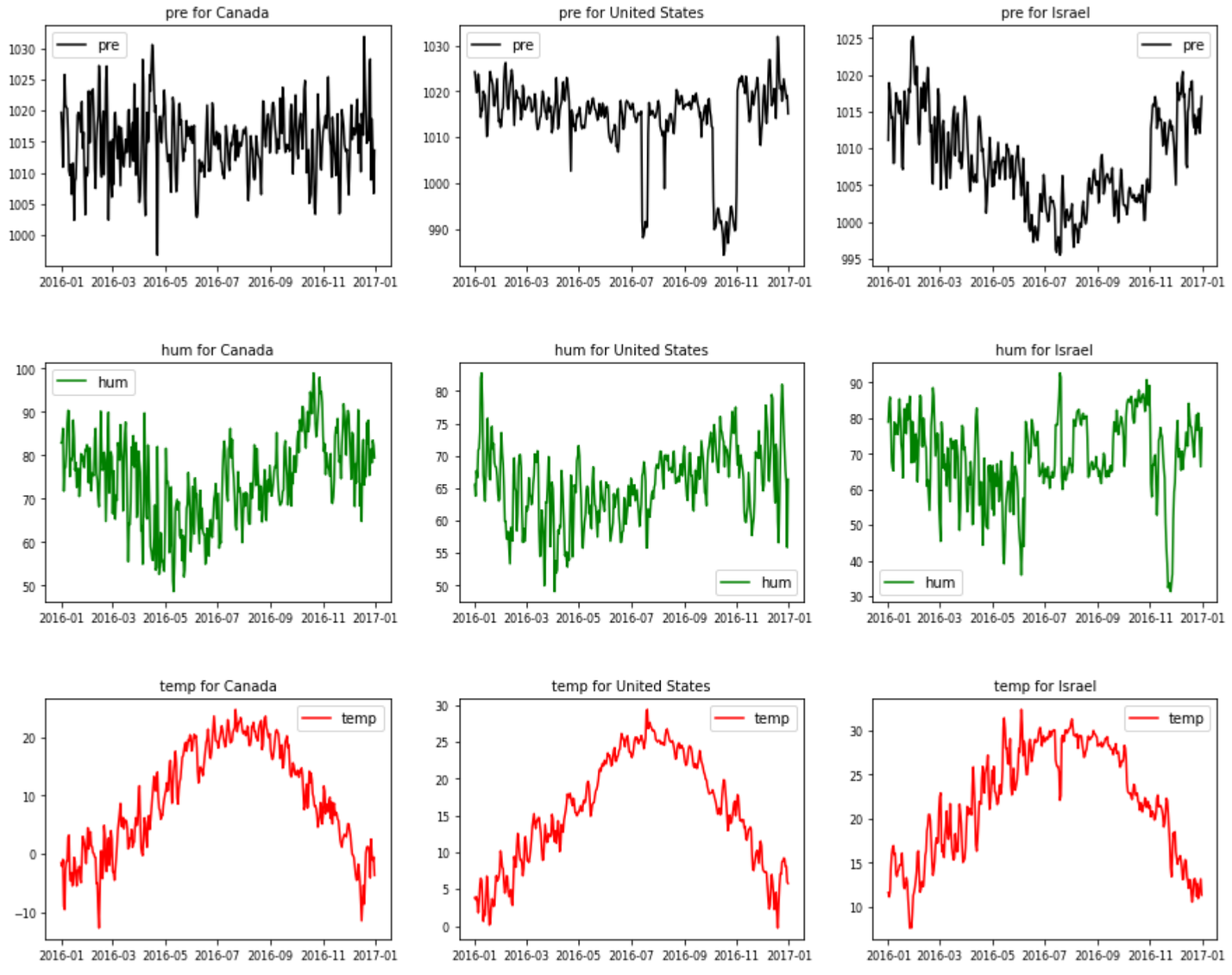
In [17]:
```
# if city is between coordinates {Longitude: [-85°:-70°] , Latitude:[40°:47°] ,then wind direction is West
# if city is between coordinates {Longitude: [-95°:-85°] , Latitude:[35°:45°] ,then wind direction is South
# In USA and Canada only Vancouver and Miami are the cities where wind direction is East
# There are only 4 cities in USA where wind direction is North(percentage is not high though).these cities are between latitute 30° and 34°
# In Israel wind direction is West for most of the city
```

In [18]:
```
fig=px.histogram(df,x="ws",
             histnorm='probability density',
             color="Country",
             range_x=(0,15),
                 color_discrete_sequence=["red","black","yellow"]
            )
fig.update_layout(title_text="Wind Speed Distribution by Country")
fig.update_xaxes(title_text="Wind Speed")


fig.show()
```



Find how pressure, humidity and temperature was changing in 2016 for each country

In [21]:
```
fig,axes=plt.subplots(3,3,figsize=(15,12))
colors=["k","g","r"]
for j, country in enumerate(df.Country.unique()):
    for (i, var),color in zip(enumerate(["pre","hum","temp"]),colors):
        ax = axes[i, j]
        grouped=df[(df.Country==country)&(df.year==2016)].groupby("date")[var].mean().reset_index().sort_values("date")
        grouped.plot(ax=ax,kind="line",x="date",y=var,xlabel="",fontsize=8,color=color)
        ax.set_title(var+" for "+country,size=10)
plt.subplots_adjust(hspace=0.4)
plt.show()
```



In [ ]:
```

```