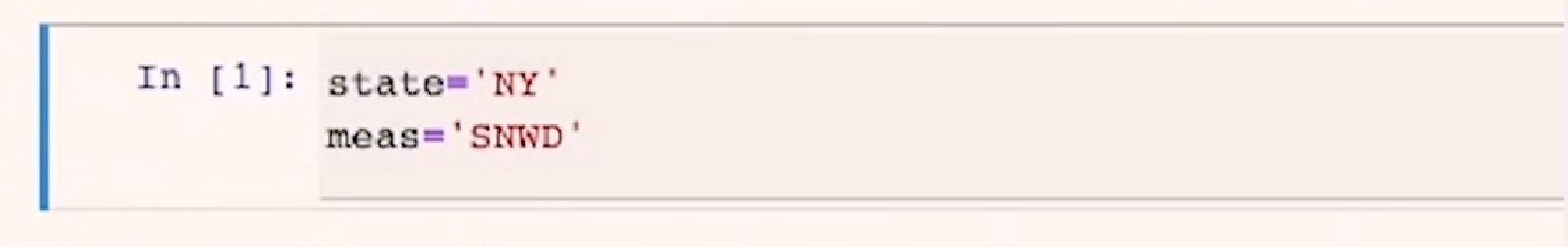
9.3 Visualizing Coefficients On a Map

DSC 232R, Class 9: PCA for Weather Data

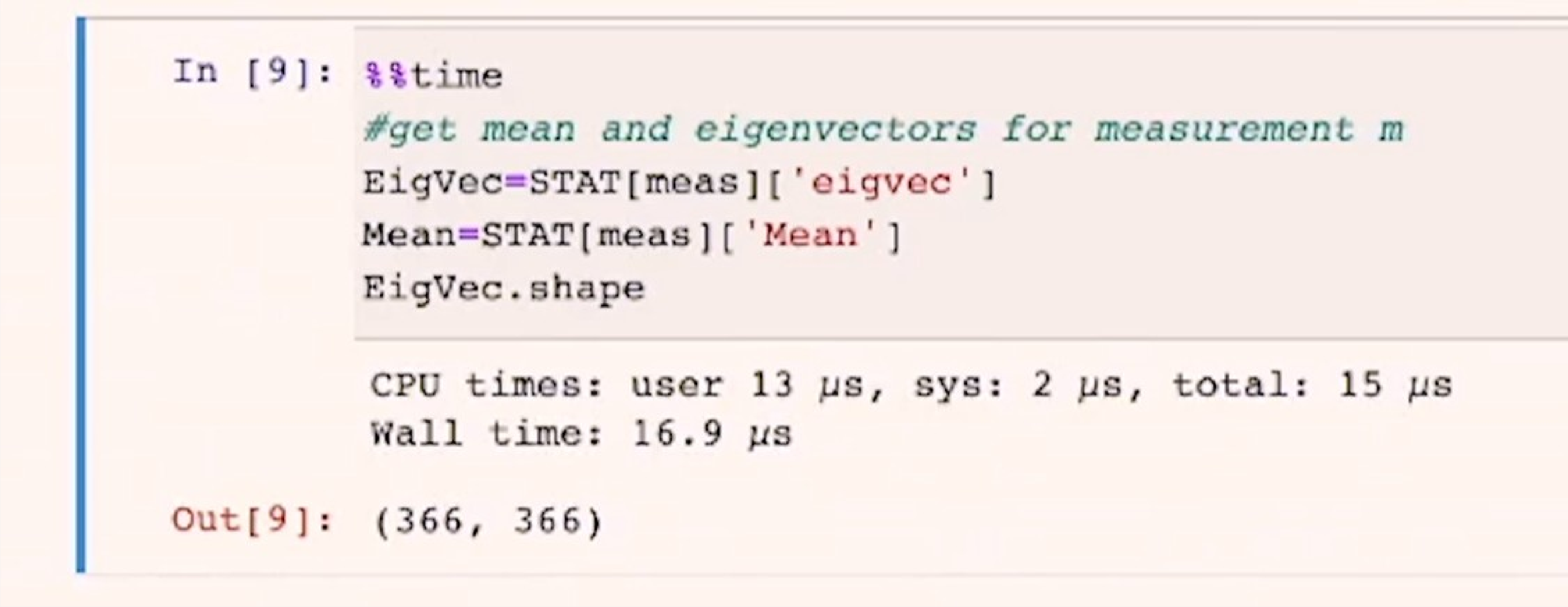
Visualizing Coefficients on a Map

Visualizing Coefficients on a Map



```
In [1]: state='NY'
        meas='SNWD'
```

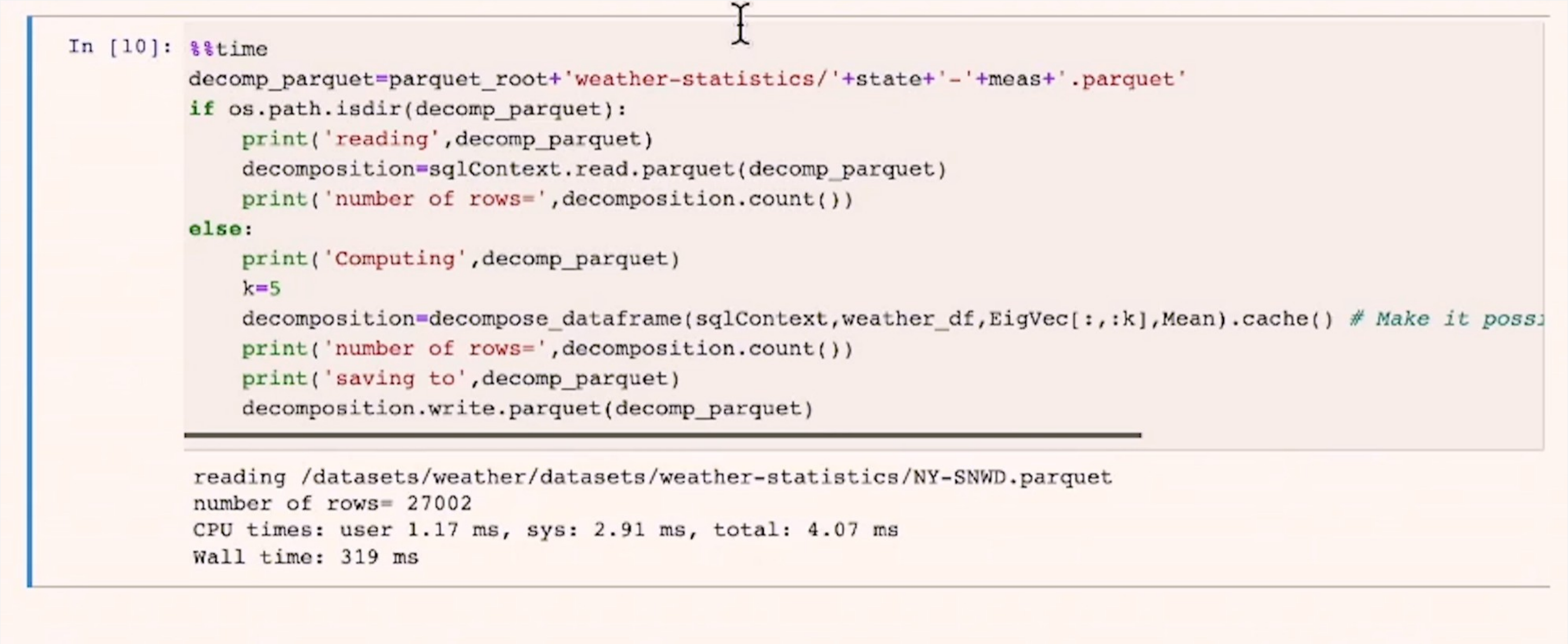
Compute Spectral Decomposition



```
In [9]: %%time
        #get mean and eigenvectors for measurement m
        EigVec=STAT[meas]['eigvec']
        Mean=STAT[meas]['Mean']
        EigVec.shape
```

```
CPU times: user 13 µs, sys: 2 µs, total: 15 µs
Wall time: 16.9 µs
```

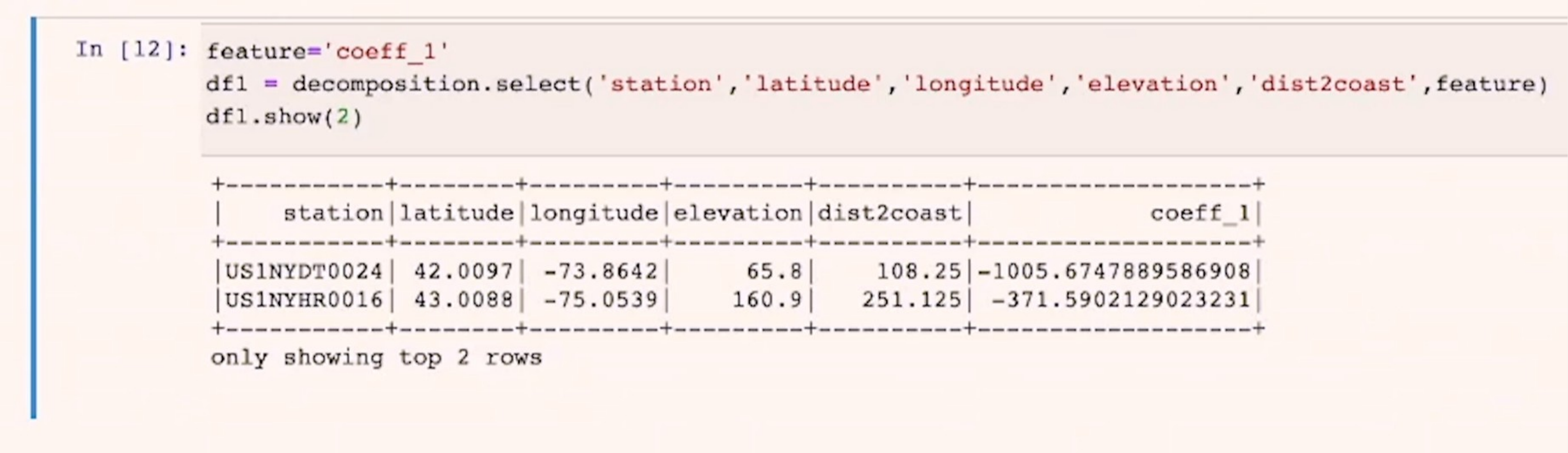
```
Out[9]: (366, 366)
```



```
%%time
decomp_parquet=parquet_root+'weather-statistics/'+state+'-'+meas+'.parquet'
if os.path.isdir(decomp_parquet):
    print('reading',decomp_parquet)
    decomposition=sqlContext.read.parquet(decomp_parquet)
    print('number of rows=',decomposition.count())
else:
    print('Computing',decomp_parquet)
    k=5
    decomposition=decompose_dataframe(sqlContext,weather_df,EigVec[:,:k],Mean).cache() # Make it poss
    print('number of rows=',decomposition.count())
    print('saving to',decomp_parquet)
    decomposition.write.parquet(decomp_parquet)
```

```
reading /datasets/weather/datasets/weather-statistics/NY-SNWD.parquet
number of rows= 27002
CPU times: user 1.17 ms, sys: 2.91 ms, total: 4.07 ms
Wall time: 319 ms
```

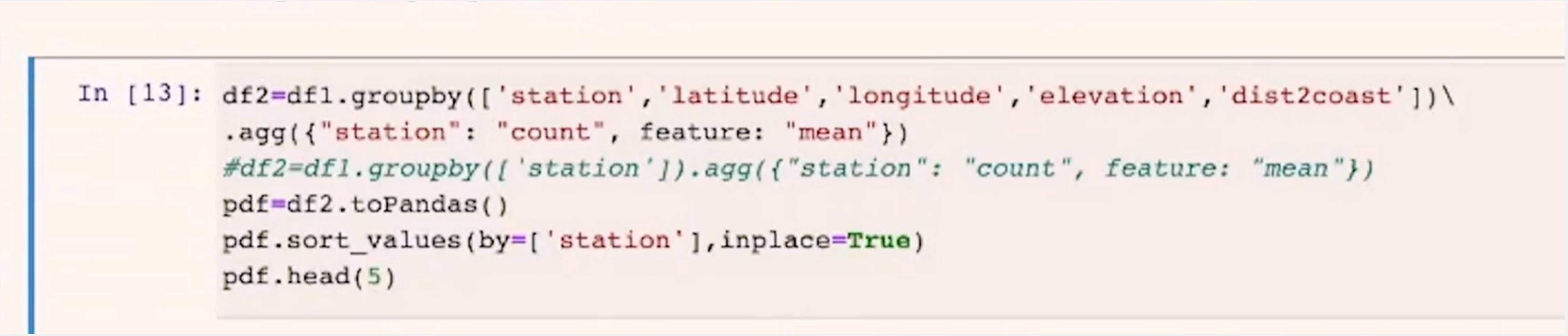
Compute the count and average of `coeff_1` for each station



In [12]:
```
feature='coeff_1'
df1 = decomposition.select('station','latitude','longitude','elevation','dist2coast',feature)
df1.show(2)
```

```
+-----------+--------+---------+---------+----------+------------------+
|    station|latitude|longitude|elevation|dist2coast|           coeff_1|
+-----------+--------+---------+---------+----------+------------------+
|US1NYDT0024| 42.0097| -73.8642|     65.8|    108.25|-1005.6747889586908|
|US1NYHR0016| 43.0088| -75.0539|    160.9|   251.125| -371.5902129023231|
+-----------+--------+---------+---------+----------+------------------+
only showing top 2 rows
```

Cont.

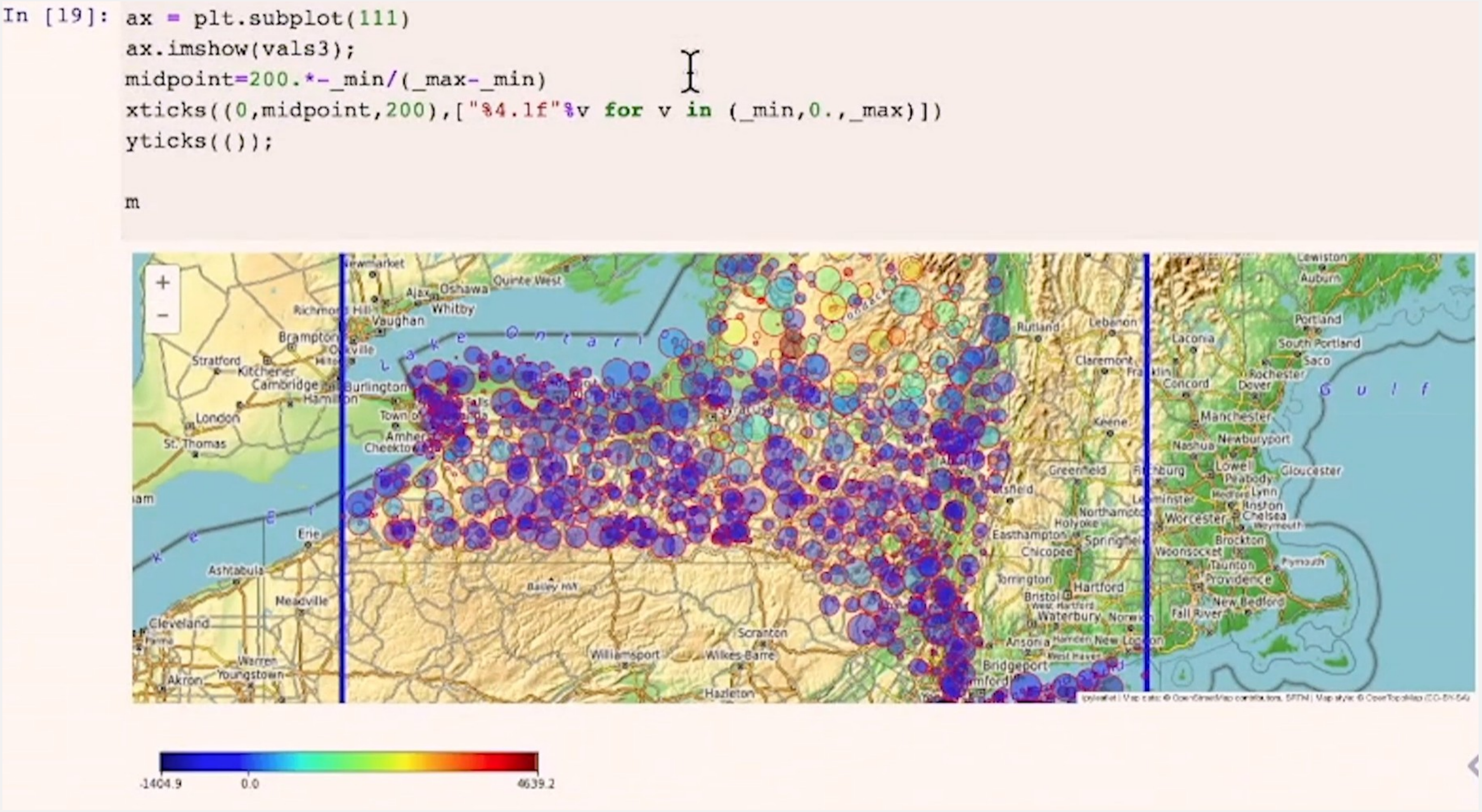


In [13]:

```
df2=df1.groupby(['station','latitude','longitude','elevation','dist2coast'])\
.agg({"station": "count", feature: "mean"})
#df2=df1.groupby(['station']).agg({"station": "count", feature: "mean"})
pdf=df2.toPandas()
pdf.sort_values(by=['station'],inplace=True)
pdf.head(5)
```

Map

- Each circle is centered at a station
- The area of the circle corresponds to the number of years SNWD was recorded at the station
- The color fill of the circle corresponds to the value of `avg(coeff_1)` defined by color-bar

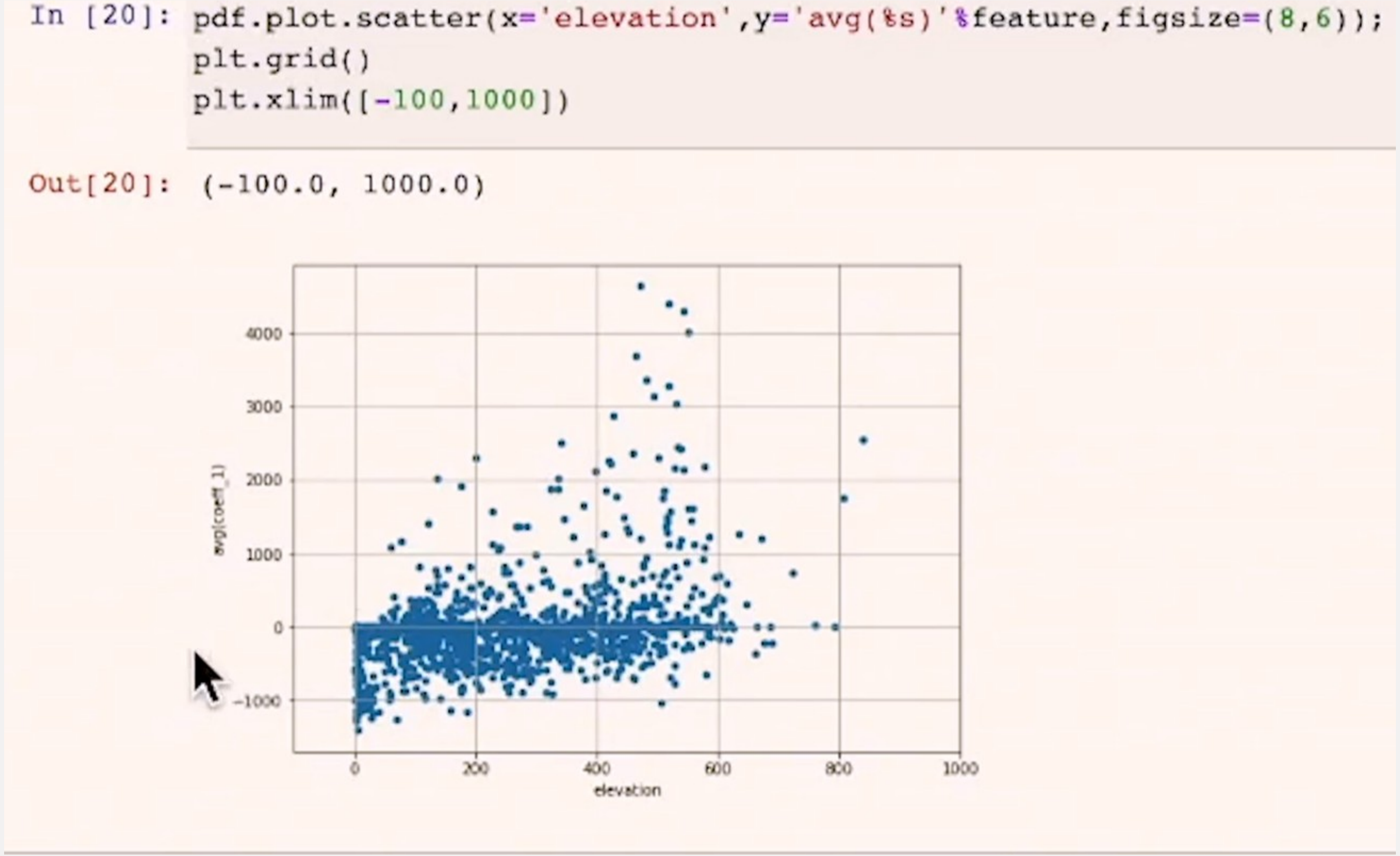


```
In [19]: ax = plt.subplot(111)
         ax.imshow(vals3);
         midpoint=200.*-_min/(_max-_min)
         xticks((0,midpoint,200),["%4.1f"%v for v in (_min,0.,_max)])
         yticks(());

         m
```

Is coef_1 related to elevation?

Is coef_1 related to elevation?

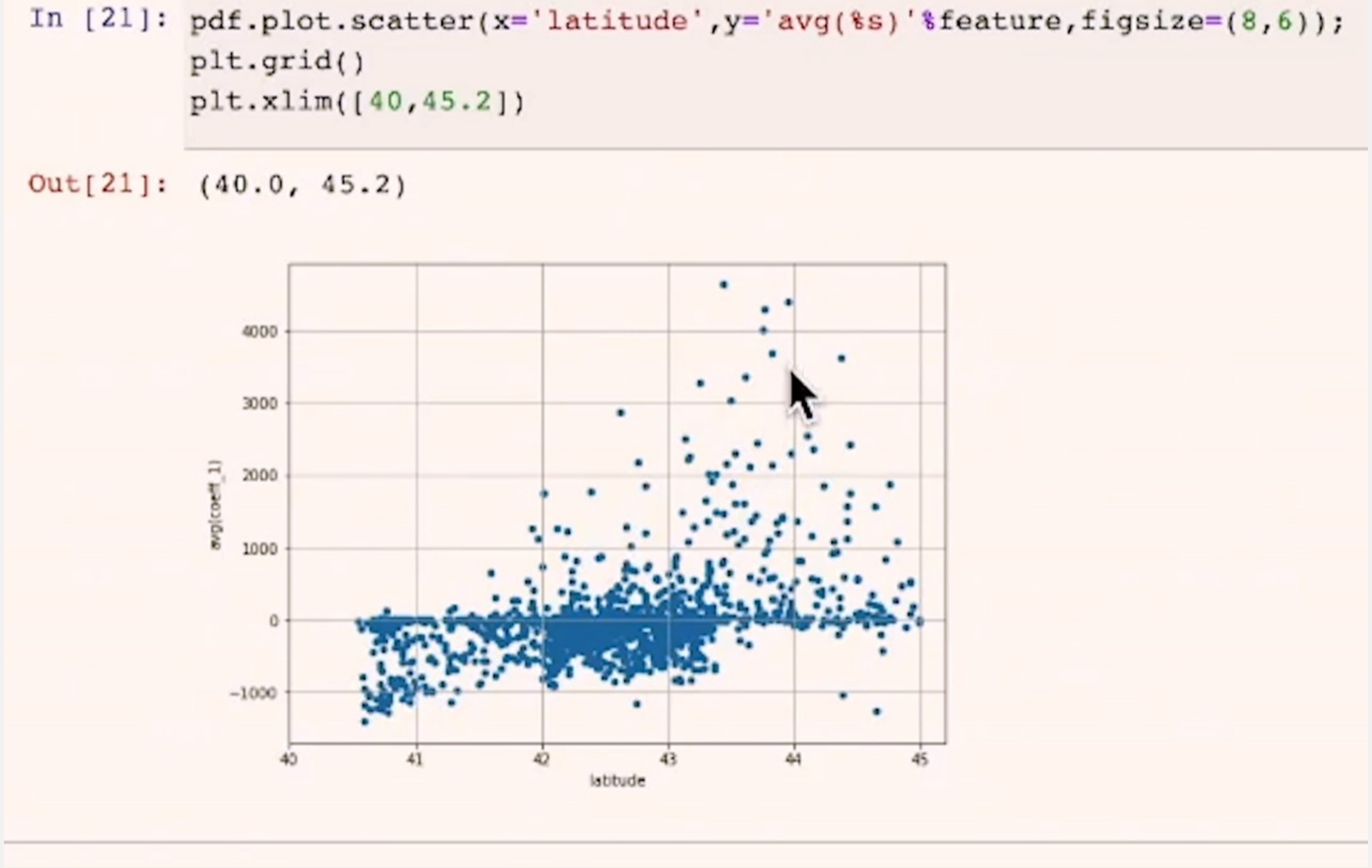


```
In [20]: pdf.plot.scatter(x='elevation',y='avg(%s)'%feature,figsize=(8,6));
         plt.grid()
         plt.xlim([-100,1000])

Out[20]: (-100.0, 1000.0)
```

Is coef_1 related to latitude?

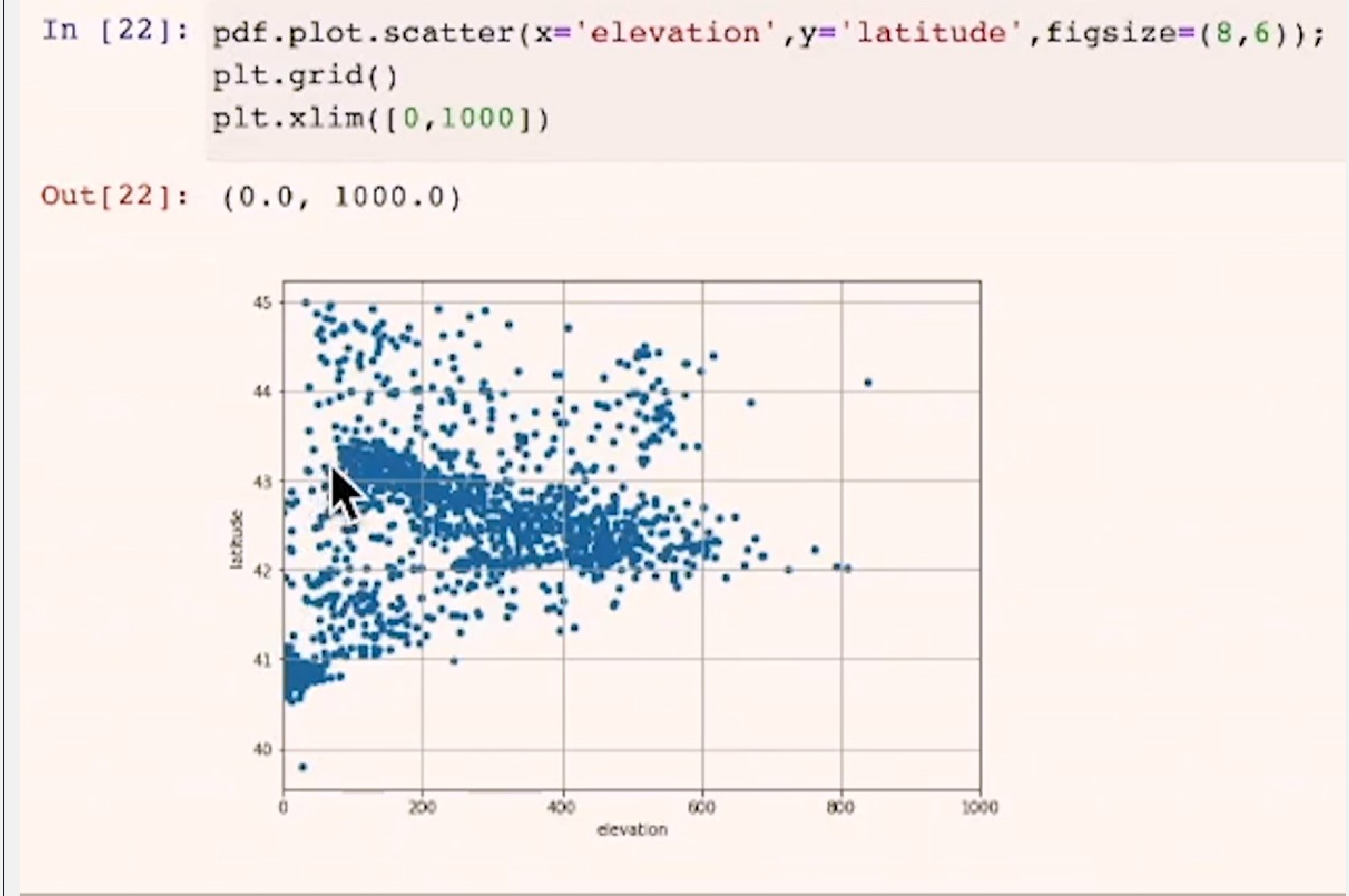
Is coef_1 related to latitude?



```
In [21]: pdf.plot.scatter(x='latitude',y='avg(%s)'%feature,figsize=(8,6));
         plt.grid()
         plt.xlim([40,45.2])
```

```
Out[21]: (40.0, 45.2)
```

Is latitude related to elevation?



```
In [22]: pdf.plot.scatter(x='elevation',y='latitude',figsize=(8,6));
         plt.grid()
         plt.xlim([0,1000])

Out[22]: (0.0, 1000.0)
```

Summary

- We saw how to use `ipyLeaflet` to present data on tops of maps
- We saw that in NY state, most of the snow accumulation is in the Adirondacks
- Snow accumulation increase with elevation, but the relationship is weak: locations with elevation 400-600 meters have widley varying accumulations of snow