miditime-New York's tone

June 6, 2020

[1]:
```
import pandas as pd
```

[26]:
```
time=pd.read_csv('data/time_series_data.csv')
others=pd.read_csv('data/other_information.csv')
location=pd.read_csv('data/location.csv')
```

[33]:
```
from miditime.miditime import MIDITime
```

[40]:
```
#try a city (new york):
new_york=time[time['Combined_Key'].str.match('New York')]
```

[52]:
```
#rename to days after out break
data=new_york.iloc[0,:]
data=data[1:]
```

[53]:
```
data.index=list(range(len(data)))
data.index
```

[53]:
```
Int64Index([  0,   1,   2,   3,   4,   5,   6,   7,   8,   9,
            ...
            125, 126, 127, 128, 129, 130, 131, 132, 133, 134],
           dtype='int64', length=135)
```

[54]:
```
data=data.to_dict()
```

[56]:
```
data=[{'beat': d, "overall": data[d]} for d in data.keys()]
```

[91]:
```
# my midi base, 120s for a year
mymidi = MIDITime(120, 'results/new_york_overall.mid', 120, 5, 1)
```

[92]:
```
my_data_timed = [{'beat': mymidi.beat(d['beat']), 'magnitude': d['overall']}␣
↪for d in data]
```

[93]:
```
base=new_york.iloc[0,:][1:]
```

[94]:
```
min_b=min(base)
```

[95]:
```
max_b=max(base)
```

[96]:
```
def mag_to_pitch_tuned(magnitude, low=min_b, high=max_b):
    # Where does this data point sit in the domain of your data? (I.E. the min␣
↪magnitude is 3, the max in 5.6). In this case the optional 'True' means the␣
↪scale is reversed, so the highest value will return the lowest percentage.
```

```
    scale_pct = mymidi.linear_scale_pct(low, high, magnitude)

    # Another option: Linear scale, reverse order
    # scale_pct = mymidi.linear_scale_pct(3, 5.7, magnitude, True)

    # Another option: Logarithmic scale, reverse order
    #scale_pct = mymidi.log_scale_pct(low, high, magnitude, True)

    # Pick a range of notes. This allows you to play in a key.
    c_major = ['C', 'D', 'E', 'F', 'G', 'A', 'B']
    #Find the note that matches your data point

    note = mymidi.scale_to_note(scale_pct, c_major)

    #Translate that note to a MIDI pitch
    midi_pitch = mymidi.note_to_midi_pitch(note)

    return midi_pitch
```

[97]:
```
note_list = []
counter=0
for d in my_data_timed:
    try:
        note_list.append([
            d['beat'],
            mag_to_pitch_tuned(d['magnitude']),
            100,  # velocity
            1  # duration, in beats
        ])
        counter +=1
    except Exception as e:
        print(d['beat'])
        print(counter)
        # some notes may do not match
```

[98]:
```
#finish
mymidi.add_track(note_list)
mymidi.save_midi()
```

```
60 0.0 1 100
60 0.66 1 100
60 1.31 1 100
60 1.97 1 100
60 2.63 1 100
60 3.29 1 100
60 3.94 1 100
60 4.6 1 100
60 5.26 1 100
```

```
60 5.91 1 100
60 6.57 1 100
60 7.23 1 100
60 7.89 1 100
60 8.54 1 100
60 9.2 1 100
60 9.86 1 100
60 10.51 1 100
60 11.17 1 100
60 11.83 1 100
60 12.48 1 100
60 13.14 1 100
60 13.8 1 100
60 14.46 1 100
60 15.11 1 100
60 15.77 1 100
60 16.43 1 100
60 17.08 1 100
60 17.74 1 100
60 18.4 1 100
60 19.06 1 100
60 19.71 1 100
60 20.37 1 100
60 21.03 1 100
60 21.68 1 100
60 22.34 1 100
60 23.0 1 100
60 23.66 1 100
60 24.31 1 100
60 24.97 1 100
60 25.63 1 100
60 26.28 1 100
60 26.94 1 100
60 27.6 1 100
60 28.25 1 100
60 28.91 1 100
60 29.57 1 100
60 30.23 1 100
60 30.88 1 100
60 31.54 1 100
60 32.2 1 100
60 32.85 1 100
60 33.51 1 100
60 34.17 1 100
60 34.83 1 100
60 35.48 1 100
60 36.14 1 100
60 36.8 1 100
```

```
60 37.45 1 100
60 38.11 1 100
60 38.77 1 100
60 39.43 1 100
60 40.08 1 100
60 40.74 1 100
60 41.4 1 100
60 42.05 1 100
60 42.71 1 100
62 43.37 1 100
62 44.02 1 100
62 44.68 1 100
62 45.34 1 100
62 46.0 1 100
62 46.65 1 100
62 47.31 1 100
64 47.97 1 100
64 48.62 1 100
64 49.28 1 100
64 49.94 1 100
64 50.6 1 100
64 51.25 1 100
65 51.91 1 100
65 52.57 1 100
65 53.22 1 100
65 53.88 1 100
65 54.54 1 100
67 55.2 1 100
67 55.85 1 100
67 56.51 1 100
67 57.17 1 100
67 57.82 1 100
67 58.48 1 100
67 59.14 1 100
67 59.79 1 100
67 60.45 1 100
69 61.11 1 100
69 61.77 1 100
69 62.42 1 100
69 63.08 1 100
69 63.74 1 100
69 64.39 1 100
69 65.05 1 100
69 65.71 1 100
69 66.37 1 100
69 67.02 1 100
69 67.68 1 100
71 68.34 1 100
```

```
71 68.99 1 100
71 69.65 1 100
71 70.31 1 100
71 70.97 1 100
71 71.62 1 100
71 72.28 1 100
71 72.94 1 100
71 73.59 1 100
71 74.25 1 100
71 74.91 1 100
71 75.56 1 100
71 76.22 1 100
71 76.88 1 100
71 77.54 1 100
71 78.19 1 100
71 78.85 1 100
71 79.51 1 100
71 80.16 1 100
71 80.82 1 100
71 81.48 1 100
71 82.14 1 100
71 82.79 1 100
71 83.45 1 100
71 84.11 1 100
71 84.76 1 100
71 85.42 1 100
71 86.08 1 100
71 86.74 1 100
71 87.39 1 100
71 88.05 1 100
```

The midi sounds like it is growing higher and higher, so maybe I can use some differences in data to see variantion

[100]:
```
data2=base.diff()
```

[102]:
```
data2=data2.fillna(0)
```

[104]:
```
data2.index=list(range(len(data2)))
data2=data2.to_dict()
data2=[{'beat': d, "overall": data2[d]} for d in data2.keys()]
```

[105]:
```
mymidi2 = MIDITime(120, 'results/new_york_daily.mid', 120, 5, 1)
```

[106]:
```
my_data_timed2 = [{'beat': mymidi2.beat(d['beat']), 'magnitude': d['overall']}␣
↪for d in data2]
```

[108]:
```
min_b2=min(base.diff().fillna(0))
```

[110]:
```
max_b2=max(base.diff().fillna(0))
```

[111]:
```
note_list2 = []
counter=0
for d in my_data_timed2:
    try:
        note_list2.append([
            d['beat'],
            mag_to_pitch_tuned(d['magnitude'], min_b2, max_b2),
            100,  # velocity
            1  # duration, in beats
        ])
        counter +=1
    except Exception as e:
        print(d['beat'])
        print(counter)
        # some notes may do not match
```

[112]:
```
#finish
mymidi2.add_track(note_list2)
mymidi2.save_midi()
```

```
60 0.0 1 100
60 0.66 1 100
60 1.31 1 100
60 1.97 1 100
60 2.63 1 100
60 3.29 1 100
60 3.94 1 100
60 4.6 1 100
60 5.26 1 100
60 5.91 1 100
60 6.57 1 100
60 7.23 1 100
60 7.89 1 100
60 8.54 1 100
60 9.2 1 100
60 9.86 1 100
60 10.51 1 100
60 11.17 1 100
60 11.83 1 100
60 12.48 1 100
60 13.14 1 100
60 13.8 1 100
60 14.46 1 100
60 15.11 1 100
60 15.77 1 100
60 16.43 1 100
60 17.08 1 100
60 17.74 1 100
```

```
60 18.4 1 100
60 19.06 1 100
60 19.71 1 100
60 20.37 1 100
60 21.03 1 100
60 21.68 1 100
60 22.34 1 100
60 23.0 1 100
60 23.66 1 100
60 24.31 1 100
60 24.97 1 100
60 25.63 1 100
60 26.28 1 100
60 26.94 1 100
60 27.6 1 100
60 28.25 1 100
60 28.91 1 100
60 29.57 1 100
60 30.23 1 100
60 30.88 1 100
60 31.54 1 100
60 32.2 1 100
60 32.85 1 100
60 33.51 1 100
60 34.17 1 100
60 34.83 1 100
60 35.48 1 100
60 36.14 1 100
60 36.8 1 100
62 37.45 1 100
62 38.11 1 100
64 38.77 1 100
62 39.43 1 100
64 40.08 1 100
64 40.74 1 100
64 41.4 1 100
65 42.05 1 100
65 42.71 1 100
65 43.37 1 100
65 44.02 1 100
65 44.68 1 100
69 45.34 1 100
65 46.0 1 100
65 46.65 1 100
67 47.31 1 100
69 47.97 1 100
65 48.62 1 100
67 49.28 1 100
```

```
67 49.94 1 100
67 50.6 1 100
67 51.25 1 100
67 51.91 1 100
69 52.57 1 100
67 53.22 1 100
65 53.88 1 100
65 54.54 1 100
71 55.2 1 100
67 55.85 1 100
65 56.51 1 100
65 57.17 1 100
64 57.82 1 100
64 58.48 1 100
64 59.14 1 100
64 59.79 1 100
65 60.45 1 100
67 61.11 1 100
67 61.77 1 100
64 62.42 1 100
62 63.08 1 100
62 63.74 1 100
64 64.39 1 100
64 65.05 1 100
62 65.71 1 100
64 66.37 1 100
62 67.02 1 100
62 67.68 1 100
62 68.34 1 100
62 68.99 1 100
62 69.65 1 100
62 70.31 1 100
62 70.97 1 100
62 71.62 1 100
60 72.28 1 100
60 72.94 1 100
62 73.59 1 100
62 74.25 1 100
62 74.91 1 100
62 75.56 1 100
60 76.22 1 100
60 76.88 1 100
60 77.54 1 100
60 78.19 1 100
62 78.85 1 100
60 79.51 1 100
60 80.16 1 100
60 80.82 1 100
```

```
60 81.48 1 100
60 82.14 1 100
60 82.79 1 100
60 83.45 1 100
60 84.11 1 100
60 84.76 1 100
60 85.42 1 100
60 86.08 1 100
60 86.74 1 100
60 87.39 1 100
60 88.05 1 100
```

[ ]: