Microsoft: DAT210x Programming with Python for Data Science

Bookmarks

- Start Here
- 1. The Big Picture
- **2. Data And Features**

Lecture: Features Premiere
Quiz

Lecture: Determining Features
Quiz

Lecture: Manipulating Data
Quiz

Lecture: Feature Representation
Quiz

Lecture: Wrangling Data
Quiz

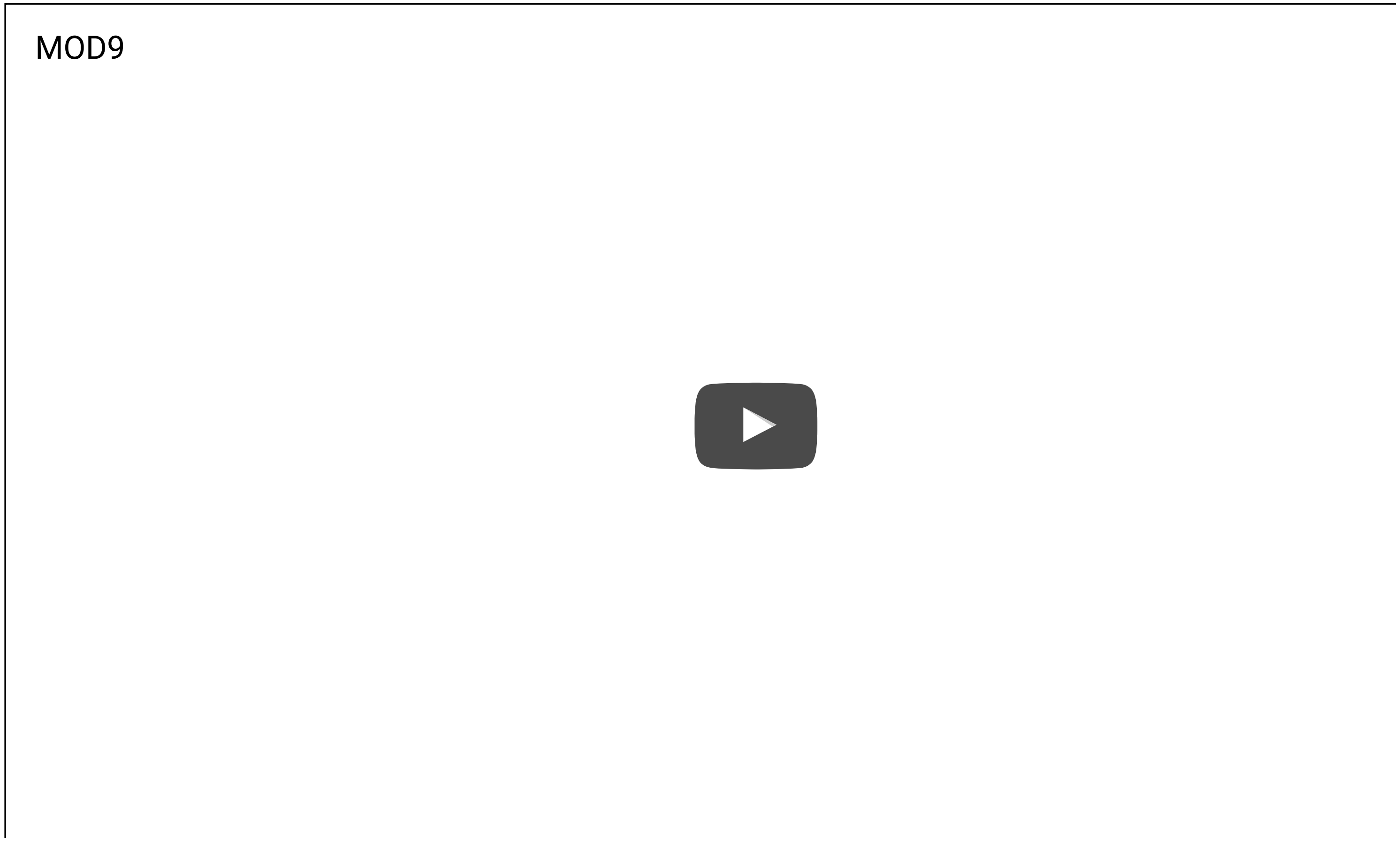
Lab: Data and Features
Lab

Dive Deeper

2. Data And Features > Lecture: Feature Representation > Video

Bookmark

Feature Representation



3. Exploring Data

4. Transforming Data

5. Data Modeling

▶ 0:00 / 6:39 1.0x

Download video Download transcript .srt

Your features need to be represented as quantitative (preferably numeric) attributes of the thing you're sampling. They may be real world values, such as the actual readings from a sensor or other discernible, physical properties. Alternatively, your features can also be calculated derivatives, such as the presence of certain edges and curves in an image, or lack thereof.

If your data comes to you in a nicely formed, numeric and tabular format, then that's one less thing for you to worry about. But this will seldom be the case, and you will often encounter data in textual or other unstructured forms. Luckily, there are a few techniques that can be applied in these scenarios, and your full understanding of your problem should open up even additional data encoding paths for you.

Textual Categorical-Features

If you have a categorical feature, the way to represent it in your dataset depends on if it's ordinal or nominal. For ordinal features, map the order as increasing integers in a single numeric feature. Any entries not found in your designated categories list will be mapped to -1:

```
>>> ordered_satisfaction = ['Very Unhappy', 'Unhappy', 'Neutral', 'Happy', 'Very
Happy']
>>> df = pd.DataFrame({'satisfaction':['Mad', 'Happy', 'Unhappy', 'Neutral']})
>>> df.satisfaction = df.satisfaction.astype("category",
  ordered=True,
  categories=ordered_satisfaction
).cat.codes

>>> df
   satisfaction
0            -1
1             3
2             1
3             2
```

On the other hand, if your feature is nominal (and thus there is no obvious numeric ordering), then you have two options. The first is you can encoded it similar as you did above. This would be a fast-and-dirty approach. While you're just getting accustomed to your dataset and taking it for its first run through your data analysis pipeline, this method might be best:

```
>>> df = pd.DataFrame({'vertebrates':[
...  'Bird',
...  'Bird',
...  'Mammal',
...  'Fish',
...  'Amphibian',
...  'Reptile',
...  'Mammal',
... ]})

# Method 1)
>>> df['vertebrates'] = df.vertebrates.astype("category").cat.codes

>>> df
  vertebrates  vertebrates
0        Bird            1
1        Bird            1
2      Mammal            3
3        Fish            2
4   Amphibian            0
5     Reptile            4
6      Mammal            3
```

Notice how this time, `ordered=True` was not pass in, nor was a specific ordering list. Because of this, Pandas encodes your nominal entries in alphabetical order. This approach is fine for getting your feet wet, but the issue it has is that it still introduces some kind of ordering to a categorical list of items that has none. This may or may not cause problems for you in the future. If you aren't getting the results you hoped for, or even if you *are* getting the results you desired but would like to further increase their accuracy, then a more precise encoding approach would be to separate the distinct values out into individual boolean features:

```
# Method 2)
>>> df = pd.get_dummies(df,columns=['vertebrates'])

>>> df
   vertebrates_Amphibian  vertebrates_Bird  vertebrates_Fish  \
0                    0.0               1.0               0.0
1                    0.0               1.0               0.0
2                    0.0               0.0               0.0
3                    0.0               0.0               1.0
4                    1.0               0.0               0.0
5                    0.0               0.0               0.0
6                    0.0               0.0               0.0

   vertebrates_Mammal  vertebrates_Reptile
0                 0.0                  0.0
1                 0.0                  0.0
2                 1.0                  0.0
3                 0.0                  0.0
4                 0.0                  0.0
5                 0.0                  1.0
6                 1.0                  0.0
```

Pandas `.get_dummies()` method allows you to completely replace a single, nominal feature with multiple boolean indicator features. This method is quite powerful and has many configurable options, including the ability to return a SparseDataFrame, and other prefixing options. It's benefit is that no erroneous ordering is introduced into your dataset.