Course introduction

WORKING WITH CATEGORICAL DATA IN PYTHON

Kasey Jones
Research Data Scientist

What does it mean to be "categorical"?

Categorical

- Finite number of groups (or categories)
- These categories are usually fixed or known (eye color, hair color, etc.)
- Known as qualitative data

Numerical

- Known as quantitative data
- Expressed using a numerical value
- Is usually a measurement (height, weight, IQ, etc.)

Ordinal vs. nominal variables

Ordinal

- Categorical variables that have a natural order

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
1	2	3	4	5

Nominal

- Categorical variables that cannot be placed into a natural order

Blue   Green   Red   Yellow   Purple

Our first dataset

```
adult.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32561 entries, 0 to 32560
Data columns (total 15 columns):
 #   Column           Non-Null Count  Dtype
 --  ------           --------------  -----
 0   Age              32561 non-null  int64
 1   Workclass        32561 non-null  object
 2   fnlgwt           32561 non-null  int64
 3   Education        32561 non-null  object
 4   Education Num    32561 non-null  int64
 5   Marital Status   32561 non-null  object
...
```

[1] https://www.kaggle.com/uciml/adult-census-income

Using describe

```
adult["Marital Status"].describe()
```

```
count                   32561
unique                      7
top        Married-civ-spouse
freq                    14976
Name: Marital Status, dtype: object
```

Using value counts

```
adult["Marital Status"].value_counts()
```

```
 Married-civ-spouse       14976
 Never-married            10683
 Divorced                  4443
 Separated                 1025
 Widowed                    993
 Married-spouse-absent      418
 Married-AF-spouse           23
Name: Marital Status, dtype: int64
```

Using value counts with normalize

```
adult["Marital Status"].value_counts(normalize=True)
```

```
 Married-civ-spouse       0.459937
 Never-married            0.328092
 Divorced                 0.136452
 Separated                0.031479
 Widowed                  0.030497
 Married-spouse-absent    0.012837
 Married-AF-spouse        0.000706
Name: Marital Status, dtype: float64
```

Knowledge check

WORKING WITH CATEGORICAL DATA IN PYTHON

Categorical data in pandas

WORKING WITH CATEGORICAL DATA IN PYTHON

Kasey Jones
Research Data Scientist

dtypes: object

```
adult = pd.read_csv("data/adult.csv")
adult.dtypes
```

```
Age                  int64
Workclass           object
fnlgwt               int64
Education           object
Education Num        int64
Marital Status      object
Occupation          object
Relationship        object
...
```

dtypes: categorical

Default dtype:

```
adult["Marital Status"].dtype
```

```
dtype('O')
```

Set as categorical:

```
adult["Marital Status"] = adult["Marital Status"].astype("category")
```

```
adult["Marital Status"].dtype
```

```
CategoricalDtype(categories=[' Divorced', ' Married-AF-spouse',
' Married-civ-spouse', ' Married-spouse-absent', ' Never-married',
' Separated', ' Widowed'], ordered=False)
```

Creating a categorical Series

```
my_data = ["A", "A", "C", "B", "C", "A"]
```

```
my_series1 = pd.Series(my_data, dtype="category")
print(my_series1)
```

```
0    A
1    A
2    C
...
dtype: category
Categories (3, object): [A, B, C]
```

Creating a categorical Series

```python
my_data = ["A", "A", "C", "B", "C", "A"]
my_series2 = pd.Categorical(my_data, categories=["C", "B", "A"], ordered=True)
my_series2
```

```
[A, A, C, B, C, A]
Categories (3, object): [C < B < A]
```

Why do we use categorical: memory

Memory saver:

```
adult = pd.read_csv("data/adult.csv")
adult["Marital Status"].nbytes
```

```
260488
```

```
adult["Marital Status"] = adult["Marital Status"].astype("category")
adult["Marital Status"].nbytes
```

```
32617
```

Specify dtypes when reading data

1. Create a dictionary:

```
adult_dtypes = {"Marital Status": "category"}
```

2. Set the `dtype` parameter:

```
adult = pd.read_csv("data/adult.csv", dtype=adult_dtypes)
```

3. Check the `dtype`:

```
adult["Marital Status"].dtype
```

```
CategoricalDtype(categories=[' Divorced', ' Married-AF-spouse',
                             ..., ' Widowed'], ordered=False)
```

pandas category practice

WORKING WITH CATEGORICAL DATA IN PYTHON

Grouping data by category in pandas

WORKING WITH CATEGORICAL DATA IN PYTHON

Kasey Jones
Research Data Scientist

The basics of .groupby(): splitting data

```
adult = pd.read_csv("data/adult.csv")
adult1 = adult[adult["Above/Below 50k"] == " <=50K"]
adult2 = adult[adult["Above/Below 50k"] == " >50K"]
```

is replaced by

```
groupby_object = adult.groupby(by=["Above/Below 50k"])
```

The basics of .groupby(): apply a function

```
groupby_object = adult.groupby(by=["Above/Below 50k"])
```

Apply a function:

```
groupby_object.mean()
```

```
                       Age         fnlgwt  Education Num  Capital Gain ...
Above/Below 50k
 <=50K           36.783738   190340.86517       9.595065    148.752468 ...
 >50K            44.249841   188005.00000      11.611657   4006.142456 ...
```

One liner:

```
adult.groupby(by=["Above/Below 50k"]).mean()
```

Specifying columns

Option 1: only runs `.sum()` on two columns.

```
adult.groupby(by=["Above/Below 50k"])['Age', 'Education Num'].sum()
```

```
                     Age  Education Num
Above/Below 50k
 <=50K            909294         237190
 >50K             346963          91047
```

Option 2: runs `.sum()` on all numeric columns and then subsets.

```
adult.groupby(by=["Above/Below 50k"]).sum()[['Age', 'Education Num']]
```

Option 1 is preferred - especially when using large datasets

Groupby multiple columns

```
adult.groupby(by=["Above/Below 50k", "Marital Status"]).size()
```

```
Above/Below 50k  Marital Status
 <=50K             Divorced                  3980
                   Married-AF-spouse           13
                   Married-civ-spouse        8284
                   Married-spouse-absent      384
                   Never-married            10192
                   Separated                  959
                   Widowed                    908
 >50K              Divorced                   463
                   Married-AF-spouse           10 <--- Only 10 records
                   Married-civ-spouse        6692
    ...
```

Practice using .groupby()