Categorical Data

This lesson discusses what is categorical data and how Pandas provide support dealing with it.

WE'LL COVER THE FOLLOWING

Introduction to categorical data

Sometimes you get categorical data which are variables with a limited and usually fixed number of values. For example, **male** and **female**. Machine learning algorithms need numbers to work, so how do you deal with these? We will discuss two ways:

- Label encoding
- One-hot encoding a.k.a. dummy variables.

Dealing with categorical data

Label encoding

Label encoding works by converting the unique values to a numeric representation. For example, if we have two categories **male** and **female**, we can categorize them as numbers:

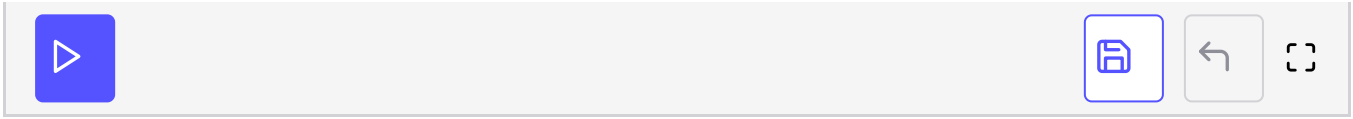
- male as **0**
- female **1**

Pandas provides an easy way to do this by using the `category` type.

```
import pandas as pd

# Create series with male and female values

non_categorical_series = pd.Series(['male', 'female', 'male', 'female'])
# Convert the text series to a categorical series
categorical_series = non_categorical_series.astype('category')
# Print the numeric codes for each value
print(categorical_series.cat.codes)
# Print the category names
print(categorical_series.cat.categories)
```



In the code above, we create a Pandas series with text values of either *male* or *female*.

We then cast the column using the `astype()` function and pass the type of `category`. This finds all the unique values in our column and assigns each a unique integer value while still maintaining the string values.

You can get the integer values by adding `.cat.codes` to the end of your category series.

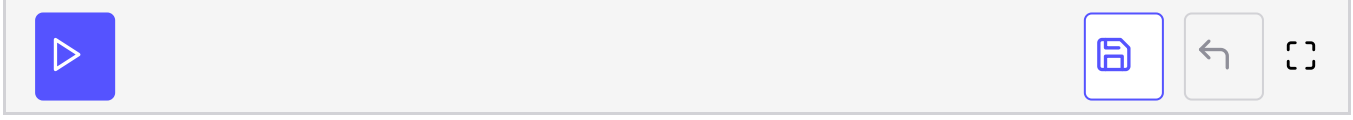
You get the string values by adding `.cat.categories` to the end of your category series.

One-hot encoding

One-hot encoding is similar but creates a new column for each category and fills it with a 1 for each row with that value and zero otherwise.

```
import pandas as pd

# Create series with male and female values
non_categorical_series = pd.Series(['male', 'female', 'male', 'female'])
# Create dummy or one-hot encoded variables
print(pd.get_dummies(non_categorical_series))
```



We see that we just had to use the `get_dummies()` call on our series and it automatically makes new columns for each unique value in our series and fills them with a 1 or 0 as appropriate.

Now that you have a grasp on some ways of cleaning our data, the next lesson

Now that you have a grasp on some ways of cleaning our data, the next lesson brings you a challenge to solve.