Missing Values

This lesson will elaborate on how to deal with data that has missing values.

WE'LL COVER THE FOLLOWING

Missing values

During data collection and entry, it is possible that some values are missed, or data was not available for some entries. Hence, missing data is very common among data science applications.

Pandas makes it very easy to work with missing data. It does not include missing values in all of its different calculations such as sum, mean, etc. by default.

Pandas writes the value `NaN` (Not a Number) when it finds a missing value.

Detecting missing values

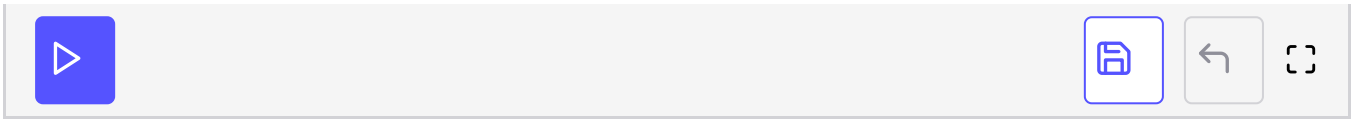
We can detect missing values using the function `isnull`. It returns `True` wherever there is a missing value, and `False`, otherwise.

```
import pandas as pd
df = pd.read_csv('housing.csv')

# check which columns have how many missing values
```

```
print("Missing values in every column : \n" ,df.isnull().sum())
print("\n\n Missing values total : ",df.isnull().sum().sum())


# Display rows that have missing values
missing = df['total_bedrooms'].isnull()
print(df[missing])
```



In **line 5**, we use the function `isnull` and then use `sum` on it. This gives us a list of all columns with the number of missing values in them. From the list, we see that `total_bedrooms` has 207 missing values. If we take another `sum` then we have the total number of missing values in the whole dataset, as done in **line 6**.

In **line 9**, we use `isnull` on `total_bedrooms` column and get a list of `True` and `False`. In the next line, we index using this list and display all the rows that have missing values in the total_bedrooms column.

Passing custom keywords for missing values

The function `isnull` detects `NaN` or empty cells. But what if at the time of data entry, someone wrote a custom value where values were missing, such as `NA`, `missing`, or `N/A`. We can pass a list of these keywords when reading the files to `read_csv` as `missing_values`, and Pandas will perceive these keywords as missing. For instance, if our data has the terms `missing` and `N/A` for missing values, we can write

```
pd.read_csv('filename.csv',missing_values = ['missing','N/A'])
```

Filtering missing values

Pandas has a function `dropna` which filters the rows containing missing values from the dataframe.

```
import pandas as pd
df = pd.read_csv('housing.csv')

print('Shape of original dataframe : ', df.shape)

# filter using dropna
new_df = df.dropna()
print('Shape of filtered dataframe : ', new_df.shape)
```

```
# filter using dropna - drop columns
new_df = df.dropna(axis=1)
print('Shape of filtered dataframe : ', new_df.shape)
```

In line 7, we create a new dataframe which has filtered rows that had missing values. We can see from the output that `new_df` has fewer rows. If we pass `axis=1`, then every column that has a missing value will be dropped. As it can be seen from the output of **line 12**.

`notnull`

As an alternative to dropna(), we can use the notnull() function to remove rows that have missing values in one or more columns.

```
import pandas as pd
df = pd.read_csv('housing.csv')

print('Shape of original dataframe : ', df.shape)
print(df.notnull())
# Display rows that do not have missing values
present = df['total_bedrooms'].notnull()
new_df = df[present]
print('Shape of filtered dataframe : ', new_df.shape)
```

In **line** 7, we use the function `notnull` on the `total_bedrooms` column to get a list of rows that do not have missing values for `total_bedrooms`. In the next line, we filter using this list. As we can see from the output of **line 9**, that the new dataframe has fewer number of rows.

Filling numerical missing values

The strategy behind filling missing values can vary from problem to problem. In some cases, filling in with the mean or median value works very well, while in some cases **interpolation** works better. Interpolation is a mathematical technique of constructing new points from a range of points.

Pandas provides two very easy ways to fill in missing values.

- We can provide any value to the function `fillna` to fill in the missing

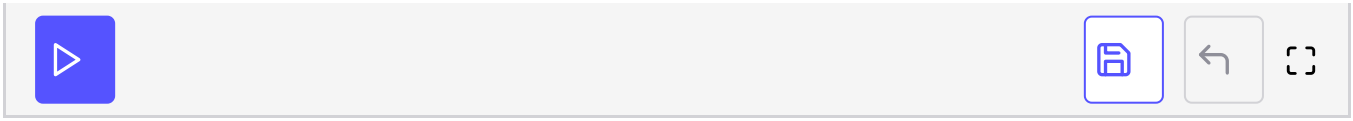
values with.

- We can specify the type of interpolation we want to the `interpolate` function.

```
import pandas as pd
df = pd.read_csv('housing.csv')
print("Total missing values : ",df.isnull().sum().sum())

# fill in values with the median value
median = df['total_bedrooms'].median()
new_df = df.fillna(median)
print("Total missing values : ",new_df.isnull().sum().sum())

new_df = df.interpolate('linear')
print("Total missing values : ",new_df.isnull().sum().sum())
```



We first find the median in **line 6** and then pass it to `fillna` to fill the missing values.

We only provide the name of the interpolation to the `interpolate` function in **line 10**.

Filling non-numerical missing values

For non-numerical missing values, we can simply write `missing` because we cannot use mathematical methods like mean, median, or interpolation to estimate these.

If the variable in question is a very important variable for which we need values, then it may be better to simply disregard the observation.

We have learned how to detect, filter and fill missing values. In the next lesson, we will learn how to deal with duplicates.