Optimizing Dataframe Memory Footprint: Takeaways

Syntax

- Returning an estimate for the amount of memory a dataframe consumes:

```
DataFrame.info()
```

- Retrieving the internal BlockManager object:

```
DataFrame._data
```

- Retrieving the amount of memory the values in a column consume:

```
Series.nbytes
```

- Returning the number of values in a dataframe:

```
DataFrame.size
```

- Returning the true memory footprint of a dataframe:

```
DataFrame.info(memory_usage="deep")
```

- Returning the amount of memory each column consumes:

```
DataFrame.memory_usage(deep=True)
```

- Finding the minimum and maximum values for each integer subtype:

```
import numpy as np
int_types = ["int8", "int16", "int32", "int64"]
for it in int_types:
    print(np.iinfo(it))
```

- Finding the minimum and maximum values for each float subtype:

```
import numpy as np
float_types = ["float16", "float32", "float64", "float128"]
for ft in float_types:
    print(np.finfo(ft))
```

- Converting a column to a specific datatype:

```
Series.astype()
```

- Converting a column to the most space efficient subtype:

```
pd.to_numeric(Series, downcast='integer')
```

- Converting a column to the datetime type:

```
pd.to_datetime(Series)
```

- Converting a column to the category datatype:

```
Series.astype('Category')
```

- Specify the column types when reading in data:

```
import numpy as np
col_types = {"id": np.int32}
df = pd.read_csv('data.csv', dtypes=col_types
```

Concepts

- The BlockManager class is responsible for maintaining the mapping between the row and column indexes and the blocks of values of the same data type.
- Pandas uses the ObjectBlock class to represent blocks containing string columns and the FloatBlock class to represent blocks containing float columns.
- Pandas represents numeric values as NumPy ndarrays, whereas pandas represents string values as Python string objects.
- A kilobyte is equivalent to 1,024 bytes and a megabyte is equivalent to 1,048,576 bytes.
- Many types in pandas have multiple subtypes that can use fewer bytes to represent each value. For example, the `float` type has the `float16`, `float32`, `float64`, and `float128` subtypes. The number portion of a type's name indicates the number of bits that type uses to represent values.
- Subtypes for the most common panda types:

object	bool	float	int	datetime
object	bool	float16	int8	datetime64
		float32	int16	
		float64	int32	
		float128	int64	

- The category datatype uses integer values under the hood to represent the values in a column, rather than the raw values. Categoricals are useful whenever a column contains a limited set of values.

Resources

- Documentation for the BlockManager class
- Documentation for the pd.read_csv() function