Indexing and Filtering Datasets

Chris Achard

@nanohopdev www.nanohop.com

Overview

Indexing and Filtering

- Access just certain rows or columns
- loc and iloc
- Using string methods with loc

Demo

Direct access with square brackets df[]

Powerful filtering with .loc and .iloc

String method .contains

Direct Filtering with Square Brackets

Data Indexing with .loc

Using .iloc to Access Specific Rows or Columns

Filtering Data with str.contains

Review

```
df['id']
```

◀ Access a column as a Series

```
df['id'][1]
```

◀ Access a single row on a column

```
df[1:5]
```

◀ Access a range of rows with a slice

```
df[data['year'] > 1800]
```

◀ Use a basic filter

```
df.loc[ROWS, COLS]
```

- Basic format of .loc

```
df.loc[0:2, :]
```

- Access a slice of rows and all columns

```
df.loc[0:2, [‘title’, ‘artist’]]
```

- Access a slice of rows and specific columns

```
data.loc[data.artist == 'Blake, Robert', :]
```

- Filter on rows, and select all columns

df.iloc[ROWS, COLS]

- Basic format of .iloc

df.loc[0:2, :]

- Slice of rows by integer position end is *exclusive*

df.iloc[[1, 5], [12, 100]]

- Access specific rows and specific columns by position

```
df.col.str.contains(‘serach’)
```

◀ Contains string method

```
df.loc[df.col.str.contains(‘search’), [‘artist’, ‘title’]]
```

◀ Filter data with contains and loc and select certain columns

```
df.loc[df.col.str.contains(’one|two', case=False, regex=True), :]
```

◀ Case insensitive regex search

```
df.col.astype(str).str.contains(‘search’, na=False)
```

◀ Convert column type, and ignore NaN values