23 great Pandas codes for Data Scientists

GEORGE SEIF

Here are 23 Pandas codes for Data Scientists to help better understand your data!

Basic Dataset Information

(1) Read in a CSV dataset

```
pd.DataFrame.from_csv("csv_file")
```

```
pd.read_csv("csv_file")
```

(2) Read in an Excel dataset

```
pd.read_excel("excel_file")
```

(3) Write your data frame directly to csv

Comma separated and without the indices

```
df.to_csv("data.csv", sep=",", index=False)
```

(4) Basic dataset feature info

```
df.info()
```

(5) Basic dataset statistics

```
print(df.describe())
```

(6) Print data frame in a table

```
print(tabulate(print_table, headers=headers))
```

where “print_table” is a list of lists and “headers” is a list of the string headers

(7) List the column names

```
df.columns
```

Basic Data Handling

(8) Drop missing data

```
df.dropna(axis=0, how='any')
```

Returns object with labels on given axis omitted where alternately any or all of the data are missing

(9) Replace missing data

```
df.replace(to_replace=None, value=None)
```

replaces values given in “to_replace” with “value”.

(10) Check for NANs

```
pd.isnull(object)
```

Detect missing values (NaN in numeric arrays, None/NaN in object arrays)

(11) Drop a feature

```
df.drop('feature_variable_name', axis=1)
```

axis is either 0 for rows, 1 for columns

(12) Convert object type to float

```
pd.to_numeric(df["feature_name"], errors='coerce')
```

Convert object types to numeric to be able to perform computations (in case they are string)

(13) Convert data frame to numpy array

```
df.as_matrix()
```

(14) Get first “n” rows of a data frame

```
df.head(n)
```

(15) Get data by feature name

```
df.loc[feature_name]
```

Operating on data frames

(16) Apply a function to a data frame

This one will multiple all values in the “height” column of the data frame by 2

```
df["height"].apply(lambda height: 2 * height)
```

```
def multiply(x):
    return x * 2
```

```
df["height"].apply(multiply)
```

(17) Renaming a column

Here we will rename the 3rd column of the data frame to be called “size”

```
df.rename(columns = {df.columns[2]:'size'}, inplace=True)
```

(18) Get the unique entries of a column

Here we will get the unique entries of the column “name”

```
df["name"].unique()
```

(19) Accessing sub-data frames

Here we’ll grab a selection of the columns, “name” and “size” from the data frame

```
new_df = df[["name", "size"]]
```

(20) Summary information about your data

```
# Sum of values in a data frame
df.sum()
```

```
# Lowest value of a data frame
df.min()
```

```
# Highest value
df.max()
```

```
# Index of the lowest value
df.idxmin()
```

```
# Index of the highest value
df.idxmax()
```

```
# Statistical summary of the data frame, with quartiles, median,
df.describe()
```

```
# Average values
df.mean()
```

```
# Median values
df.median()
```

```
# Correlation between columns
df.corr()
```

```
# To get these values for only one column, just select it like t
df["size"].median()
```

(21) Sorting your data

```
df.sort_values(ascending = False)
```

(22) Boolean indexing

Here we'll filter our data column named "size" to show only values equal to 5

```
df[df["size"] == 5]
```

(23) Selecting values

Let's select the first row of the "size" column

```
df.loc([0], ['size'])
```